driver+

Driving Innovation in Crisis Management for European Resilience

D934.21 – SOLUTION TESTING PROCEDURE

SP93 - SOLUTIONS

MARCH 2018 (M47)

This project has received funding from the European Union's 7th Framework Programme for Research, Technological Development and Demonstration under Grant Agreement (GA) N° #607798

Project information

Project Acronym:	DRIVER+
Project Full Title:	Driving Innovation in Crisis Management for European Resilience
Grant Agreement:	607798
Project Duration:	72 months (May 2014 - April 2020)
Project Technical Coordinator:	TNO
Contact:	coordination@projectdriver.eu

Deliverable information

Deliverable Status:	Final
Deliverable Title:	D934.21 – Solution testing procedure
Deliverable Nature:	Report (R)
Dissemination Level:	Public (PU)
Due Date:	March 2018 (M47)
Submission Date:	11/05/2018
Sub-Project (SP):	SP93 - Solutions
Work Package (WP):	WP934 - DRIVER+ CM Solutions
Deliverable Leader:	TCS
Reviewers:	Gerald Schimak, AIT Adam Widera, WWU Nicola Rupp, WWU Erik Vullings, TNO Francisco Gala, ATOS
File Name:	DRIVER+_D934.21_Solution testing procedure.docx

Revision Table

Issue	Date	Comment	Author
V0.1	26/02/2018	Initial draft, document structure	Laurent Dubost, TCS
V0.2	02/03/2018	Document, Sections, 1, 2, 3, Contribution to Section 4, Solution testing procedure	Laurent Dubost, TCS Ludwig Kastner, FRQ
V0.3	21/03/2018	Document reshaping after remarks from early review. Contribution to Sections 4.3 Test plan and 5.2 Information workflow	Laurent Dubost, TCS Todor Tagarev, Valeri Ratchev, CSDM Antoine Léger, TCS
V0.4	22/03/2018	After reviews by Gerald Schimak, Adam Widera: alignment with D923.11 and D923.21, redistribution of section on Guidelines, and Process, examples added, explanation of concepts reworked. Examples from Trial 1 added. (section 3.3.3).	Laurent Dubost, TCS Hector Naranjo, GMV
V0.5	31/04/2018	Contribution to sections 3.1 and 3.2: Test-bed related integration and testing	Joaquin Marquez Bugella, FRQ
V0.8	12/04/2018	Contribution to Annex 3, examples of User stories and Test cases in PoS,	Christoph Ruggenthaler, AIT
V0.9	13/04/2018	Modification of document after Review board meeting.	Laurent Dubost, TCS, Erik Vullings, TNO
V0.10	17/04/2018	Finalisation of final draft version: consistency check, correction of typos.	Gunnar Schwoch, DLR, Jan Kraemer, DLR, Erik Vullings, TNO, Joaquin Marquez Bugella, FRQ
V0.11	26/04/2018	General review and quality/formatting check	Francisco Gala, ATOS
V0.12	02/05/2018	Adjustment according to general review	Laurent Dubost, TCS
V0.13	08/05/2018	Final check and approval for submission	Tim Stelkens-Kobsch, Quality Manager (DLR)
V0.14	10/05/2018	Final check and approval for submission	Peter Petiet, Project Director (TNO)
V1.0	11/05/2018	Final check and submission to the EC	Francisco Gala (ATOS)

The DRIVER+ project

Current and future challenges due to increasingly severe consequences of natural disasters and terrorist threats require the development and uptake of innovative solutions that are addressing the operational needs of practitioners dealing with Crisis Management. DRIVER+ (Driving Innovation in Crisis Management for European Resilience) is a FP7 Crisis Management demonstration project aiming at improving the way capability development and innovation management is tackled. DRIVER+ has three main objectives:

1. Develop a pan-European Test-bed for Crisis Management capability development:
    - Develop a common guidance methodology and tool (supporting Trials and the gathering of lessons learnt.
    - Develop an infrastructure to create relevant environments, for enabling the trialling of new solutions and to explore and share Crisis Management capabilities.
    - Run Trials in order to assess the value of solutions addressing specific needs using guidance and infrastructure.
    - Ensure the sustainability of the pan-European Test-bed.
2. Develop a well-balanced comprehensive Portfolio of Crisis Management Solutions:
    - Facilitate the usage of the Portfolio of Solutions.
    - Ensure the sustainability of the Portfolio of Solutions.
3. Facilitate a shared understanding of Crisis Management across Europe:
    - Establish a common background.
    - Cooperate with external partners in joint Trials.
    - Disseminate project results.

In order to achieve these objectives, five sub-projects (SPs) have been established. **SP91** ***Project Management*** is devoted to consortium level project management, and it is also in charge of the alignment of DRIVER+ with external initiatives on Crisis Management for the benefit of DRIVER+ and its stakeholders. In DRIVER+, all activities related to Societal Impact Assessment (from the former SP8 and SP9) are part of SP91 as well. **SP92** ***Test-bed*** will deliver a guidance methodology and guidance tool supporting the design, conduct and analysis of Trials and will develop a reference implementation of the Test-bed. It will also create the scenario simulation capability to support execution of the Trials. **SP93** ***Solutions*** will deliver the Portfolio of Solutions which is a database driven web site that documents all the available DRIVER+ solutions, as well as solutions from external organisations. Adapting solutions to fit the needs addressed in Trials will be done in SP93. **SP94** ***Trials*** will organize four series of Trials as well as the final demo. **SP95** ***Impact, Engagement and Sustainability***, is in charge of communication and dissemination, and also addresses issues related to improving sustainability, market aspects of solutions, and standardization.

The DRIVER+ Trials and the Final Demonstration will benefit from the DRIVER+ Test-bed, providing the technological infrastructure, the necessary supporting methodology and adequate support tools to prepare, conduct and evaluate the Trials. All results from the Trials will be stored and made available in the Portfolio of Solutions, being a central platform to present innovative solutions from consortium partners and third parties and to share experiences and best practices with respect to their application. In order to enhance the current European cooperation framework within the Crisis Management domain and to facilitate a shared understanding of Crisis Management across Europe, DRIVER+ will carry out a wide range of activities, whose most important will be to build and structure a dedicated Community of Practice in Crisis Management, thereby connecting and fostering the exchange on lessons learnt and best practices between Crisis Management practitioners as well as technological solution providers.

Executive summary

The solution testing procedure described in this document is intended for the technical stakeholders of the integration process: solution owners, coordinators and developers, as well as Test-bed infrastructure coordinators and Simulator owners and developers.

Its scope is larger than initially described in the DOW, which only discussed testing a solution standalone, as it also covers SP93 and SP94 Trial-integration activities as well. The reason for this is that these activities are of a similar nature, performed mostly by the same partners, and logically connected. Encompassing them in the same procedure will ease the continuity of the integration and testing activities, and will facilitate the coordination between SP93 (Portfolio of CM Solutions) and SP4 (Trials).

This solution testing procedure therefore covers the integration and testing activities of standalone solutions (SP93) and technical set-up of a Trial as a whole (SP94). The goal of this procedure is to make sure that the solutions and the technical set-up are ready at the end of Dry Run 1 to support the Trials execution. This procedure is purely technical, it does not address the actual assessment of the solutions which will be performed during the Trial, but it enables it. The whole procedure is under the joint responsibility of the solution coordinator and the Test-bed infrastructure coordinator, who work in coordination with the other members of the Trial Committee.

This procedure is designed to be supportive, descriptive in nature, and gives recommendations to the solution coordinator who will be able to customise it in agreement with the Trial Committee to consider her/his own familiar methods, the specific constraints of the Trial, the time schedule, the nature of the technical set-up, or the participating solutions themselves. It consists of the following three steps:

- **Step 0:** Standalone solutions – generic integration and testing; During this step, the solution owner defines the main user stories of his/her solution, translates them to test cases, and integrates the solution to the Test-bed reference implementation. This integration is tested against the solution test cases, which define the required inputs from the Test-bed, and the expected outputs of the solution. This step is not Trial specific and is performed by each solution owner, under the coordination of SP93 task leaders as a generic preparation for any Trial.
- **Step 1:** Standalone solutions – Trial specific integration and testing; during this step, individual solutions are adapted and integrated to fulfil the Trial specific requirements. Solution owners contribute to the writing of the Trial specific requirements (defined by Step 2) and write the corresponding test cases for their own solutions. This step is performed by each solution owner under the coordination of solution coordinators (SP94), and leaders of corresponding SP93 tasks.
- **Step 2:** Technical set-up integration and testing; this step aims at designing, integrating and testing the Trial's technical set-up as a whole. The technical set-up consists of a set of solutions, the Test-bed, Simulators and measuring tools. Step 2 is split in two parts: 2.1 defines the technical set-up, generates the Trial specific requirements and defines one or more test scenarios, and (after Step 1 is completed) step 2.2 performs the integration of multiple solutions and tests it against these test scenarios. A test scenario, in this sense, is a scenario that is dedicated to test the main interactions between different solutions as required by the Trial scenario. This step is coordinated jointly by the solution coordinator (SP94) and the Test-bed infrastructure coordinator (SP92) with the support of solution and simulation owners.

Guidelines and examples are given to help stakeholders design and describe the technical set-up or formulate the requirements, test cases and test scenarios needed to structure and document the integration and testing work. The way requirement and test-actions can be documented (in the test plan or in the PoS) is also described. This procedure is based on the experience of the past DRIVER experiments and will benefit from the DRIVER+ Trials. At terms, it may provide a contribution to an updated TGM (1).

A mapping of the current content of the PoS against the Taxonomy of crisis management functions for the classification of solutions (2) is also presented. This mapping is performed using the CM function perspective and the gap perspective and shows that internal solutions are aligned with the DRIVER+ gaps (3).

Table of Content

List of Figures

List of Tables

List of Acronyms

Acronym	Definition
AVRO	Remote procedure call and data serialization framework developed within Apache's Hadoop project (https://avro.apache.org/)
BPMN	Business Process Model and Notation
C3	Command, Control and Coordination (functional area)
CCIM	Crisis Communications and Information Management (functional area)
CIS	Common Information Space where solutions share information
CM	Crisis Management
COP	Common Operational Picture
CSS	Common Simulation Space where Simulators share information
DOW	Description of Work
DR1, DR2	Dry Run 1 and 2, respectively, the two test runs before the actual Trial takes place
EMSI	Emergency Management
GUI	Graphical User Interface
IT	Information Technology
LOC	Local Operational Centre
N/A	Not applicable
OK/NOK/POK	OK: successful, NOK: failed; POK: partially successful
PoS	DRIVER+ Portfolio of Solutions website
ROC	Regional Operational Centre
SMAP	Social Media Analysis Platform
T94x.5	Stands for the solutions utilisation and assessment task of each trial: T943.5, T944.5, T945.5 and T946.5
TC	Test case
TGM	Trial Guidance Methodology
TSU	Technical set-up
UML	Unified Modelling Language
US	User story
WBS	Work Breakdown Structure

1. Introduction

1.1 Identification & intended audience

This document represents the deliverable D934.21 “Solution testing procedure” of the DRIVER+ project.

Its main intended audience are the technical stakeholders of the integration process: solution owners, coordinators and developers, as well as the Test-bed infrastructure coordinator and the simulator owners.

1.2 Scope of the document

The scope of this document exceeds the scope described in the DOW: in the DOW, the proposed scope was limited to individual solution integration (SP93 tasks, encompassing T934.2 and T934.2 activities). In addition to this initial scope, the current scope also includes integrating and testing a single and multiple solutions as part of a Trial (addressed by SP94 tasks: T942 and F94x.5).

The reason for enlarging the initial scope of this document is that these activities (SP93: individual solution integration and testing, and SP94: multiple integration and testing) are of a similar nature, performed mostly by the same partners, and logically connected. Covering individual and multiple solution testing with a single overarching procedure will foster the project-wide understanding of the technical stakeholders, ease the continuity of the integration and testing, and facilitate the coordination between SP93 and SP4, as well as SP92, which is involved with the Test-bed implementation in both SP93 and SP94 integration and testing activities.

In the following, the initial description provided by the DOW concerning the present document as well as the parts of the task description most significant for this work are included. As far as SP94 is concerned, some extracts of the DOW which are of relevance for this procedure are mentioned in section 2.4.

The initial description of this document contained in the DOW (p315) is as follows:

> D934.21: solution testing procedure: This deliverable will explain the DRIVER+ solution integration testing procedure and the overall criteria for assessing the integration test results. It will provide clear guidelines for describing the solution scenarios and the underlying test cases, defining the atomic test actions in the PoS database. In addition, this deliverable will also provide guidelines for assessing the results of the test actions and for assessing the success of the whole solution scenario test. Finally, this deliverable will also provide an initial mapping of the tools and methods that were introduced in the DRIVER experiments to PoS solutions and a taxonomy of solution functions for use in the PoS.

The following part of T934.2 solution adaptation task’s description which can be found in the DOW (p297) is relevant to this document as it describes the links between WP934 and the Trials:

> T934.2: In this task, the technical partners in the DRIVER+ project will perform all the work that is needed in order for the individual tools and methods to work as a CM solution in the DRIVER+ Test-bed context and the work that is needed in order to meet additional requirements of the trial owners on particular solutions (if any). The main work will be directed towards integration in the DRIVER+ Test-bed and integration with other tools to assure they can work together as a CM solution. Feature improvements (if any) will be limited to those explicitly requested by the trial owners.

And so is the description of T934.3 solutions integration tests which can be found in the DOW (p298):

> The main objective of this task is to validate the claim that the DRIVER+ solutions are (technically) ready for use in the trials. This may involve the integration of the DRIVER+ solutions with some locally available legacy systems which need to be used in the trial. This claim will be validated by executing all the Test Cases for each of the Integration Scenarios that have been defined in the PoS in T934.2.

The description of T94x.5 solution utilization and assessment given by the Dow fin a similar manner for each Trial (p315 for Trial 1) is also relevant for this procedure:

> The solution coordinator will be responsible for the whole composition of solutions during the Trial and other events (Dry Run rehearsals). The solution coordinator derives the requirements for the solutions and communicates with selected solutions ensuring their readiness for further progress phases. If needed, the solution coordinator might reject specific solutions.

These integration and testing activities are purely technical: they contribute to the preparation of the Trial but should not be mixed with the assessment of solutions, which will occur during the execution of the Trial and thereafter.

This document contributes to the technical side of the DRIVER+ methodology. It may be a potential input to a later update of the Trial Guidance Methodology.

1.3 Document structure

After the introduction (section 1.1) which presents the document identification, intended audience and structure, an overview of the solution testing procedure is provided (section 1). Its objective (section 2.1) and structure in three steps are explained (section 2.1) as well as its schedule (section 2.2), Test plan (section 2.3) and the roles and responsibilities of its main stakeholders (section 2.4).

After this overview, practical guidelines are given in section 3 on the execution of each step. The guidelines corresponding to each step are presented in a dedicated section. Step 0, which consists in the integration and testing of standalone solutions to the Test-bed standard implementation is discussed in section 3.1. Step 1, which consists in the integration of standalone solutions in the Trial specific context, is discussed in section 3.2. Finally step 2, which deals with the integration and testing of the whole technical set-up, is discussed in sections 3.3 for sub-step 2.1 (technical set-up design) and in section 3.3 regarding step 2.2 for the actual integration and testing of multiple solutions.

For each integration and testing step, the same structure is duplicated: first the way the requirements are generated is exposed, then the generation of data sets, and then the integration and testing.

Section 3 gives additional explanations and practical guidelines for the design and description of the technical set-up (section 3.3), as well as for the writing of integration requirements, user stories, test cases and test scenarios (section 3.5).

The mapping of the solutions currently described in the PoS against the taxonomy of CM functions (2) is presented and discussed (section 4)

The conclusion is drawn in (section 5).

Several Annexes are provided: Annex 1 presents the ten terms of the DRIVER+ terminology which are most relevant for this document. Annex 2 presents the full mapping of the PoS currently available solutions against the taxonomy of CM functions. Annex 3 then presents examples of user stories and corresponding test cases as currently entered in the PoS. Annex 4 presents additional guidelines the design of the technical set-up.

2. Solution integration and testing procedure overview

This section gives an overview of the solution testing procedure as a process in terms of main steps, schedule, test plan, and roles and responsibilities.

This procedure is designed to be supportive, it is mainly descriptive. For each Trial, it can and shall be adapted by the solution coordinator in agreement with the Trial Committee to take into account the specific constraints of the Trial.

2.1 Objective

The solution integration and testing procedure's final objective is to make sure that the technical set-up, which will be used during the Trial, works as required by the Trial Committee. In other words, the testing activities ensure that the technical set-up will, at time of Trial, be able to:

- Enable participants to play the Trial scenario (1).
- Enable the collection of the data needed for the assessment of solutions (during the Trial).
- Be successfully integrated in the Trial test-bed.
- Be successfully deployed in the Trial hosting environment.

This procedure encompasses S93 activities concerning the integration and testing of standalone solutions, and SP94 activities concerning the integration and testing of multiple solutions, heading to the integration and testing of the whole Trial technical set-up.

The procedure ends when Dry Run 1 has been validated.

The word "testing" is used here as a purely technical activity related to software integration. This procedure, which aims at making sure the Trial's technical set-up is ready, is part of the Trial preparation and execution phases. It does not encompass any solution assessment activity, which is performed during the Trial evaluation phase, according to the TGM.

2.1 Structure

The solution integration and testing procedure consists of three main steps represented in Figure 2.1.

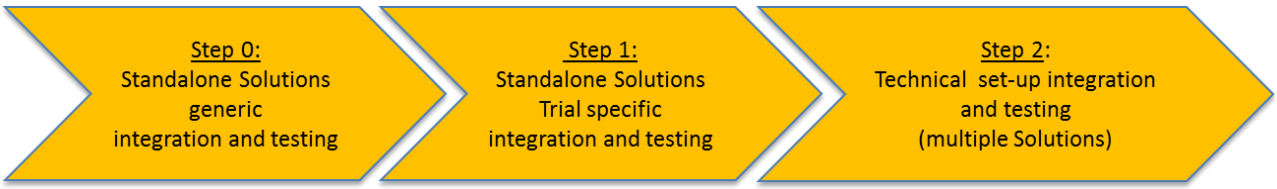

Figure 2.1: Major steps of the solution integration and testing procedure

- **Step 0: Standalone solution – generic integration and testing**. During this step, the standalone solutions are integrated to the Test-bed reference implementation. This step is generic in the sense that is not Trial specific and can be performed as a generic preparation for any Trial. The testing is performed against test cases. Table 2.1 contains a synthetic description of this step.
- **Step 1: Standalone solutions – Trial-specific integration and testing**. During this step, individual solutions are adapted and integrated to fit the Trial specific needs expressed by Trial specific requirements. This step is performed by the solution owners. The testing is performed against test cases. Table 2.2 contains a synthetic description of this step.
- **Step 2: Technical set-up integration and testing**. This step aims at integrating and testing the Trial's technical set-up. It first defines the technical set-up, and then, after standalone integration of solutions has been concluded in Step 1, it performs integration and testing of multiple solutions working collaboratively. It is divided in two sub-steps: first Step 2.1: Technical set-up design,

followed by Step 2.2: Multiple solution integration and testing. This also concludes the solution testing procedure. Table 2.3 contains a synthetic description of this Step.

These three steps are detailed respectively in sections 3.1 (Step 0), section 3.3 (Step 1), and sections 3.3 (Step 2.1) and 3.3 (Step 2.2) for Step 2.

Table 2.1: Step 0 synthetic description

	Step 0 : Standalone solution generic integration and testing
Summary	During this step, the standalone solutions are integrated to the Test-bed reference implementation. This step is generic in the sense that it is not Trial specific and can be performed as a generic preparation for any Trial.
Roles and responsibilities	This step is performed by each solution owner individually, under the coordination of corresponding task leaders (see associated tasks).
Main activities	• Define solution's user stories and corresponding test cases (Main functions and usual data inputs and outputs). • Connect solution to Test-bed standard implementation in testing environment. • Integrate solution (prepare input and output AVRO schemas). • Prepare input and output data. • Run test cases/ assess success. • Correct and run again until success.
Corresponding Test-bed Use Case	Step 0 is supported by the "integration process for a single solution or simulator" use case of the Test-bed (4).
Schedule	Step 0 can start as soon as a Test-bed implementation is available, it does not need a Trial design to be started.
Associated task(s) (in DOW)	T934.2: Solutions adaptation: for definition of solutions "users stories" and corresponding test cases (and documentation in PoS) by solution owners. T934.3 Solutions integration tests: Solution owners integrate their solution to Test-bed standard implementation, passing the test-cases, and documentation of test results.
Validation	Step 0 is validated when all test cases have been passed successfully and documented in PoS.

Table 2.2: Step 1 synthetic description

	Step 1: Standalone solution Trial specific integration
Summary	During this step, individual solutions are adapted and integrated to fit the Trial specific needs expressed by Trial specific requirements. This step is performed by the solution owners. The testing is performed against test cases.
Roles and responsibilities	This step is performed by each solution owner under the coordination of solution coordinators, and leaders of corresponding tasks (see associated tasks).
Main activities	• Collect Trial specific requirements (from Step 2.1) and write corresponding test cases. • Connect to Trial specific implementation in testing environment. • Perform adaptation and integration of solution. • Prepare input and output data.

	Step 1: Standalone solution Trial specific integration
	• Run test cases/ assess success. • Correct and run again until success.
Corresponding Test-bed Use Case	Step 1 is supported by the "integration process for a single solution or simulator" use case of the Test-bed (4).
Schedule	Step 1 can start as soon as the Trial's specific requirements are known.
Associated Task(s) (in DOW)	• T934.2 solution adaptation; for the adaptation of solutions by solution owners. • T934.3 solutions integration tests; for integration to Trial specific Test-bed implementation, and documentation of test results by solutions owners. • T94x.5: Solution utilization and assessment; for the generation of Trial specific requirements. • T924.2: Support the correct test-bed reference implementation and support its use during trials; for the support around Simulators and other Test-bed trial specific components.
Validation	Step 1 is validated when all test cases have been passed successfully and documented in PoS.

Table 2.3: Step 2 synthetic description

	Step 2.1: Technical set-up design
Summary	This step aims at integrating and testing the Trial's technical set-up. The aim of this step is to enable the selected solutions to work collaboratively and later support the Trial. During this step, the technical set-up is defined, and then, after the standalone integration of solutions has been performed in Step 1, the integration of multiple solutions is done and tested. Step 2 is divided into two sub-steps: Step 2.1: Technical set-up design which happens first in time, and Step 2.2: Multiple solutions' integration and testing which closes the solution testing procedure.
Roles and responsibilities	This step is performed jointly by the solution coordinator and the Test-bed infrastructure coordinator in close coordination with selected solutions and simulators or measuring tools owners.
Main activities	<u>Step 2.1: Technical set-up design</u> • Collect Trial's early design (scenario, solutions, participants, collection plan). • Design Trial's technical set-up and test scenarios. • Define Trial's specific requirements (for Step 1 use). <u>Step 2.2: Multiple solution integration and testing</u> • Deploy Trial's specific Test-bed implementation in Trial environment (including Simulators and Measuring tools). • Deploy solutions in Trial's specific Test-bed. • Prepare Test scenario. • Run test scenario.

	Step 2.1: Technical set-up design
	• Correct (interface or solution / simulator / measuring tool if needed). • Run again until success.
Corresponding Test-bed Use Case	Step 2 is supported by the "Set-up and testing of the Test-bed" use case of the Test-bed (5).
Schedule	• Set 2.1 must start immediately after tools selection. • Step 2.2 can start as soon as the Step 1 is completed. Step 2 can be started for a set of solutions and/or simulators as soon as Step 1 has been completed all of them. • Step 2.2 can only be completed when Step 1 has been completed for all solutions / simulators.
Associated Task(s) (in DOW)	• T942.5: Solution coordinators co-design the technical set-up and the coordinate Step 2.2. • T924.2: Test-bed infrastructure coordinator co- design the technical set-up, co-coordinate Step 2.2. • T942.2: Solution owners assist the design of technical set-up and generation of Trial specific requirements; Simulators owners and Test-bed representative of other Test-bed components take part in integration with respect to their component. • T934.3: Solution owners participate to T2.2: multiple solutions.
Validation	• Step 2 is validated when all test-scenarios have been passed successfully.

2.2 Schedule

Figure 2.2 shows the way Steps are organized in time as well as the exchanges which happen between them.

If, at the end of Dry run 1, some test scenarios are not satisfactory, corrective actions need to be decided. Depending on the seriousness of the needed corrections the Trial Committee might decide to hold a second Dry run 1 (Dry run 1 - v2) meeting before Dry run 2.

If after this second Dry run 1, some tests are still not passed, the Trial Committee will decide the way to proceed: for example, a solution might be excluded from the Trial, or the scenario may be adapted to meet the criteria.

After the validation of Dry run 1 the solution testing procedure is ended.

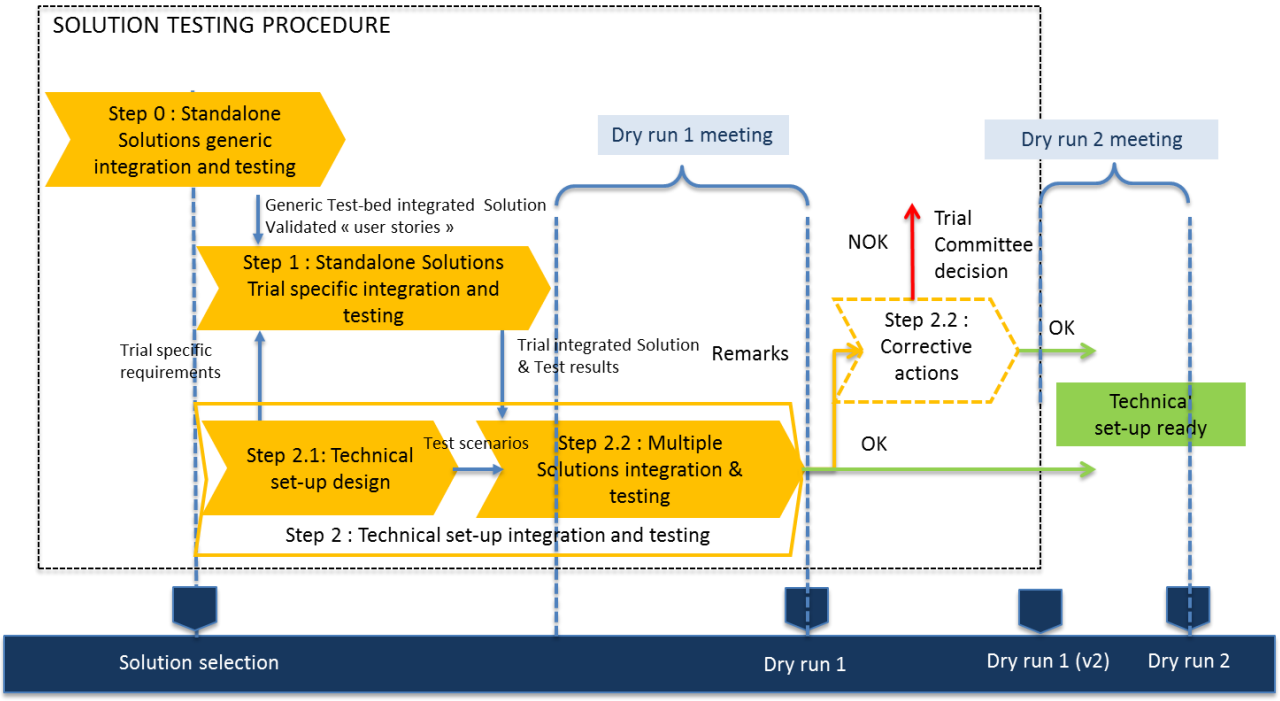

Figure 2.2: Schedule of solution testing procedure

2.3 Test plan

In order to organize the execution of integration tests (sections 3.1.3, 3.2.2, 3.4.2) and capitalize their results, the following Test plan table can be adopted. This table supports the test planning as well as the collection of test results.

Table 2.4: Test plan table

Test ID	Planned test phase	Before DR1 results	Comments	DR1 results	Comments	Trial behavior	Comment
[test ID]	Step 1, Step 2	[NOK], [POK], [OK],	<free text>	[NOK], [POK], [OK],	<free text>	[NOK], [POK], [OK],	<free text>

It is recommended to adopt such a test plan table for each solution, and one for the technical set-up.

The proposed default fields are the following:

- **Test ID**: identifier of the test is usually a set of letters and numbers. For example, the test-ID can be formed by a set of letters referring to the solution, another set referring to the requirement and a number; <Test_SMAP_data_01>.
- **Planned test step**: the periods can be Step 1, “before Dry run1” or Step 2.2, “During Dry run1”. Standalone solution testing shall be performed before Dry Run1 (DR1), multiple solutions testing is usually performed during DR1. Guidelines regarding this planning can be found in section 3.5.5.
- **Results**: the results of the test can take three values: OK if the test succeeded (the expected results described in the test case were observed), not OK (NOK) if the test failed (the expected results described in the test case were not observed), partially OK (POK), if only some results were observed, or the results observed were somehow different from the expected ones. POK and NOK

are both considered as failures, but POK is less severe than NOK. This distinction may be useful to manage the corrective actions.

- **Comments**: free text box where the person in charge of the test is expected to provide any comment. Comments are especially useful when a test failed (NOK or POK). In this case, expected corrective measures, planned dates for a new test are expected to be written here.

2.4 Roles and responsibilities

This section identifies the major actors in the solution testing procedure and describes their specific responsibilities in this process. This description completes the description of these roles which are part of the TGM (1).

The Trial Committee is working as a team; all decisions taken relatively to the solution testing are discussed within the Committee and agreed by the Trial owner.

The persons contributing directly to the solution testing procedure are:

- The solution coordinator.
- The solution owners.
- Test-bed infrastructure coordinator.
- Simulator owner.
- Trial Platform owner (which may be the Trial owner, but not necessarily).

The solution testing procedure defined in this document covers the activities of the following tasks:

- T934.2 solution adaptation; for the adaptation of solutions by solution owners.
- T934.3 solutions integration tests; for integration to Trial specific Test-bed implementation, and documentation of test results by solutions owners.
- T94x.5: solution utilization and assessment; for the generation of Trial specific requirements.
- T924.2: Support the correct test-bed reference implementation and support its use during trials.

It has strong interfaces with:

- T923 Test-bed reference implementation.

Table 2.5 presents, for each step and each stakeholder of the procedure the actions which he or she is responsible for and the task where this effort will take place.

The allocation of activities to tasks that is made in Table 2.5 is based on the DOW. The Trial Committee working as a team, for each specific Trial, this allocation of roles and responsibilities can be amended in agreement with the Trial Committee to take into account the Trial's specific constraints, organisation, or the specific skills of the Trial Committee members.

Table 2.5: Roles and steps

Role	Step 0 : Standalone Solution generic integration and testing	Step 2.1: Technical set-up design	Step 1: Standalone solution Trial specific integration	Step 2.2: Multiple solutions integration and testing
Solution coordinator		Design technical set-up Generates specific Trial requirements.	Facilitates integration process and monitors the status of individual solution testing reported by the solution owners.	Coordinates jointly with Test-bed infrastructure coordinator the final integration of the technical setup and its testing against test scenarios.
Solution owner	Defines solution user stories and associated test-cases. Documents solution in PoS. Performs generic Test-bed integration.	Contributes to set-up design, and generation of Trial specific requirements from own solution point of view.	Performs adaptation of own solution Performs integration of to Trial specific Test-bed. Defines test cases associated to Trial specific requirements, Passes test cases for own solution.	Participates in the integration and testing of own solution.
Test-bed infrastructure coordinator		Participates to technical set-up design with respect to test-bed and Simulators.	Facilitates and monitor the progress of individual solution integration and testing with respect to Test-bed and Simulators.	Participates in the integration and testing with respect to Test-bed and Simulator.
Simulator owner		Contributes to set-up design, and generation of Trial specific requirements from own solution point of view.	Participates to the individual integration and testing of solutions when required by the baseline.	Participates in the integration and testing with respect to own Simulator.
Trial Platform owner			Implements the deployment requirements regarding power, space, hardware and connectivity (not limited to).	Participates to the integration and testing with respect requirements relative to the platform. (e.g.: deployment).

2.5 Limitations

This section discusses the limitations of this procedure.

2.5.1 Tasks which are not part of this procedure

Apart from the corrective actions (section 2.2) which may be taken between Dry Run 1 and Dry Run 2, all other actions are not part of this integration and testing procedure. Although it is not the responsibility of this document to define them, the following activities can be mentioned as not being part of this procedure:

- Implementation of the whole scenario data-sets (and not only the test scenario data).
- Change of GUI interfaces language.
- Change of symbology.
- Run of the Trial scenario with those whole data sets.

Any anomaly which may occur after the validation of Dry Run 1 (while implementing the whole dataset, or while playing the actual scenario with the full dataset), will not be considered as part of this procedure, but as part of the Trial preparation itself.

2.5.2 Safety & security aspects

The usage of solutions in operational conditions might require respecting certain security or safety requirements.

In accordance with the DoW (6) the present procedure “assure(s) that the test cases for all solutions include the safety/security related testing as necessary for their later use in the crisis management context.”

Yet, considering that they are aiming at the “future use” of the solution, these safety and security related requirements will be analysed solution by solution and not in association with a specific Trial.

This analysis will be performed for all internal solutions. The results of this analysis will be included in the D934.31 document. It will contain the following:

- Identification of the main categories of safety and security concerns which are relevant for the considered solutions.
- Mapping of internal solutions against these main concerns.
- The main applicable standards or regulations which may apply in the context of Civil Protection.

3. Guidelines on execution of steps

This section gives practical guidelines to perform the integration process. First the Step 0, 1, 2.1 and 2.2 are detailed. Then additional guidelines are given on lower level concepts such as requirement, user stories, test cases and test scenarios.

3.1 Step 0: Standalone solution generic integration and testing

This section provides practical guidelines for the correct execution of Step 0: Standalone solution generic integration and testing. This integration consists in integrating the solution as a standalone solution to the Test-bed standard implementation.

These guidelines are structured in two:

- Define solutions user stories and test cases and prepare data sets.
- Integrate and test solutions against requirements.

This integration is performed on the Test-bed standard implementation (

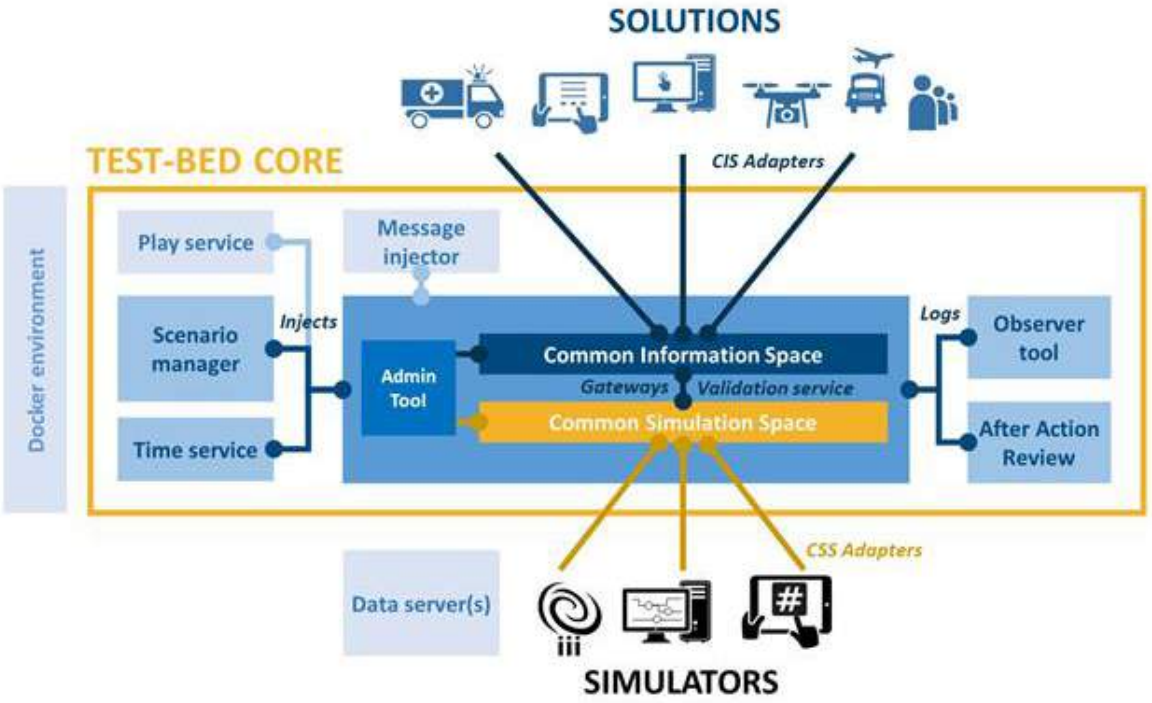

Figure 3.1) which "lies at the heart of the trialling environment of the DRIVER+ project. It provides an open source technical backbone to perform Trials or exercises in a methodical and structured way by offering practitioners a suite of free software tools." (7).

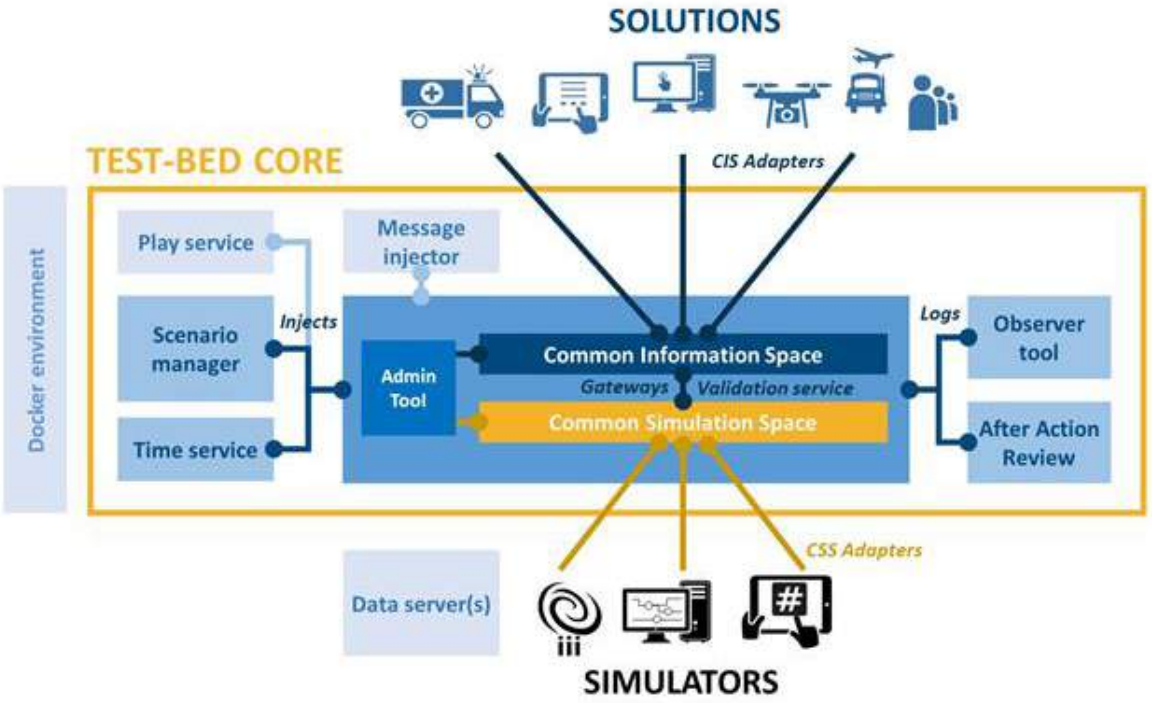

Figure 3.1: All components of the Test-bed reference implementation

3.1.1 Define user stories and test cases

User stories: Each CM solution has its *raison d'être*: some main functions that is performs and make it valuable for Crisis Management. In DRIVER+, these functions are described in user stories, which are requirements written in the user story style.

The solution's user stories describe the main functions of this solution. For practical reasons, it is recommended to focus on certain functions, which are expected to be of interest for the future Trials (a maximum of five user stories per solution is recommended).

Test cases: For each solution a set of test cases must be prepared. The purpose of these test cases is to validate the user stories. A maximum of three test cases per user story is recommended.

Therefore, the sequence of actions to be performed would be:

1. Choose main functions of solution which may have a chance to be of interest for CM Trials.
2. Write corresponding user stories (max. five recommended).
3. Enter descriptions in PoS.
4. Write corresponding test cases for each user story (max. three test cases per user story recommended).

It is recommended to include the input and output aspects in the user stories.

For example:

"As a Firefighter I want the SMAP solution to collect information from social media based on provided lists of keywords, so that I can access additional sources of information to get a better picture of the situation at hand."

More extensive guidelines on how to write user stories can be found in section 3.5.2 and example of actual user stories and corresponding test cases entered in the PoS can be found in Annex 3. Guidelines to writing test-cases can be found in section 3.5.1.

3.1.2 Preparation of test data sets

To be able to perform the test corresponding to the solution's user stories, the following test data sets need to be prepared:

- **Input data**: generate messages which are compliant with the information schema(s) which is/are specified for the inputs solution during Trial (one data set per schema, for each data set: messages containing all data types with some instances of each type)
- **Output data:** generate data sets defined in solution internal format, containing all types of data which need to be sent out by the solution. The output message generated by the solution is checked by the Test-bed schema validation service.
- **Processed data**: data similar to the type to be processed during the Trial.

For example: Scenario is a wildfire and the data types which need to be processed are: Fire start, wind direction, fire units, fire area, refill area for water bomber, plume.

For the testing of a COP tool, which needs to calculate a plume during the Trial, and needs to receive and send out EMSI situation report messages with situation report. The following data set would be generated:

Input data: EMSI situation report messages containing entities from all scenario data types (fire start, wind direction, fire units, fire area, refill area for water bomber, plume). The messages of this data set must be passed through the schema verification service *(5)* of the Test-bed.

Output data: Situation containing entities from all data types (fire start, wind direction, fire units, fire area, refill area for water bomber, plume). The COP will generate the EMSI output message(s) based on this situation data set.

Processed data: if the COP tool is able to process a Plume (i.e.: gas dispersion) from wind direction, wind speed, geolocation of source, 3D digital map, chemical product code, the processing data set will be a set of tuples: wind direction, wind speed, geolocation of source, 3D digital map, and chemical product code.

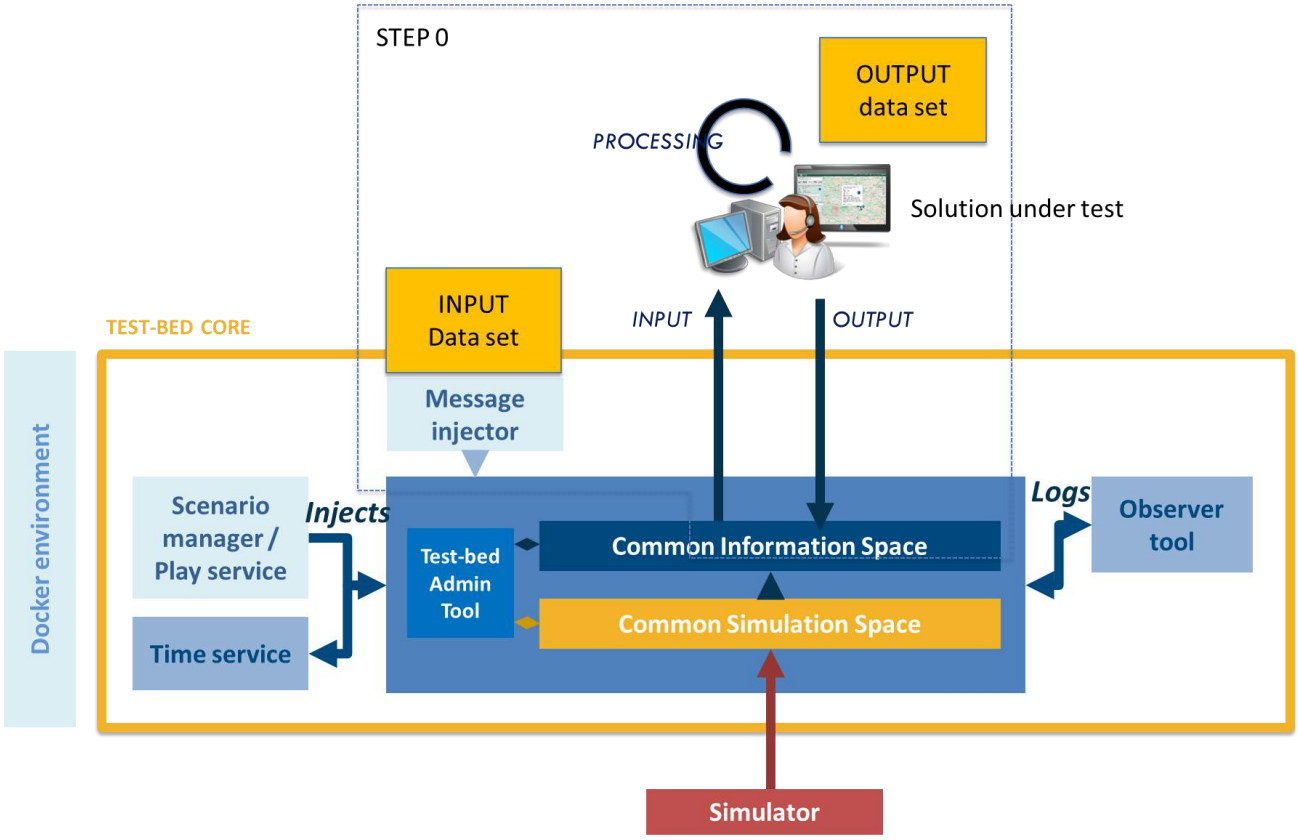

Figure 3.2: Step 0 data sets

3.1.3 Integrate and test solution against user stories

Solutions are integrated into the Test-bed via the **Common Information Space (CIS)**, which "provides an interface to connect the solutions to the Test-bed using the standardized CIS adapters" (5).

The Test-bed integration process is a technical step, which takes place after the selection of the solution. The integration is performed by the solution owner with the support of Task 934.3 leader (FRQ) and is coordinated by the Trial solution coordinator.

The Test-bed can be deployed locally (at the solution owner's premises) so the Technical department of the selected solution owner can start their Test-bed integration.

The detailed step-by-step technical guide of the Integration Process is presented in the reference implementation document (7). This integration process is structured as follows (Figure 3.3):

1. Deploy the Test-bed in the development environment (download it from GitHub (8)).
2. Download the Test-bed adapter (in accordance with the solution's development) from GitHub (8).
3. Define the schemas (AVRO schema(s)) which need to be used by the solution for its information exchange (in accordance to usual use of solution as described in user stories).
4. Make an initial "Hello world" connection test to check the basic Test-bed integration (connection established).
5. Pass the test cases corresponding to the solution's functional requirements (user stories), correct errors, and iterate until the defined Test cases are successfully passed, (correctly structured messages are sent by solution).

A library of AVRO schemas implementing some CM information exchange standards is available in the Test-bed schema registry.[1]

A more detailed and technical view of this process is made available online in a GitHub repository (8), which is constantly updated to reflect the progress of the Test-bed implementation.

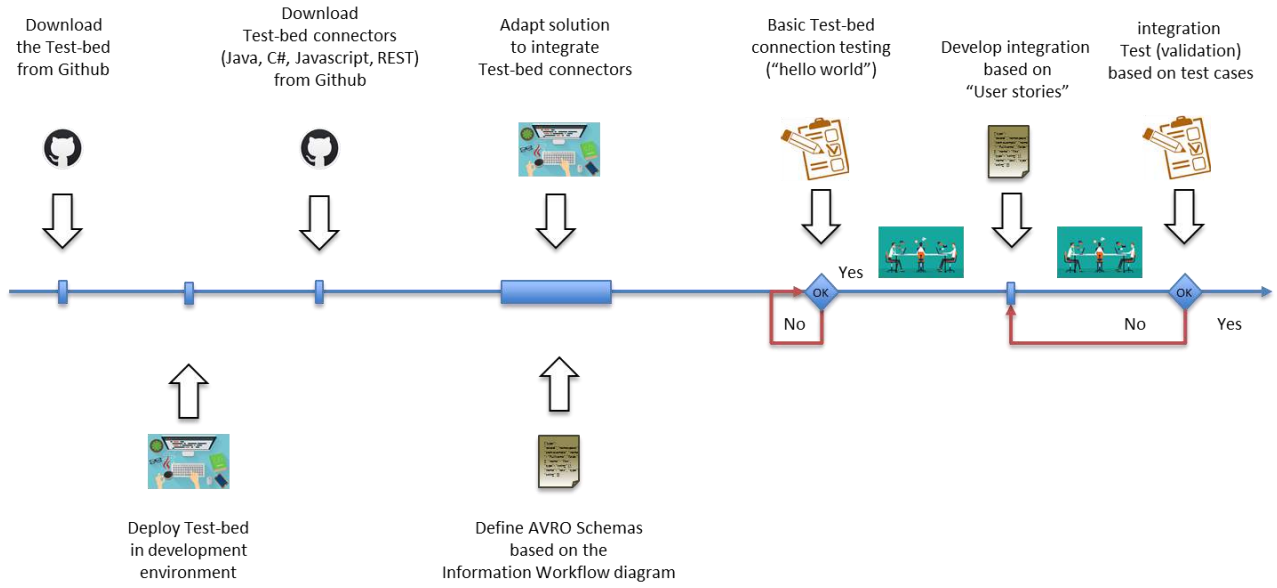

Figure 3.3: Integration of individual solution to the Test-bed

[1] "In the CM domain, several standards exist, such as CAP, EDXL or EMSI. They are represented using XML, a textual representation of a message that is easily readable by computers. A recurring problem with all standards, however, is that they rarely represent all the information you would like to share. This often leads to adding new fields, or, even worse, re-purposing existing fields. Additionally, not every organisation uses it in the same way. For trialling new solutions, the Test-bed needs to be flexible and exact, and that's why the Test-bed does support these standards but converted to the AVRO format." (7)

In order to support the solution owner's integration team solving "technical doubts" during the integration process, several Slack support channels are made available by the Test-bed reference implementation development team (9).

3.2 Step 1: Standalone solution Trial specific integration and testing

This section gives guidelines for the realization of Step 1: Standalone solution Trial specific integration and testing.

During Step 1, individual solutions are adapted and integrated to fit the Trial specific needs expressed by Trial specific requirements (generated by Step 2.1).

At this stage, Step 0 is considered as completed and the integration in the Test-bed generic implementation is done, but Trial specific aspects (mainly related to data schemas) which have not been addressed in Step 0 need to be addressed in this step.

Step 1 deals with what in the Trial is requested and had not been foreseen by solution owner at Step 0 stage: unexpected types of data, unexpected or modified schemas, or additional functions (i.e. adaptation request).

Only test cases which concern a standalone solution are performed in the step. Test cases which require interaction with multiple solutions are tested in Step 2.2.

This step is performed by the solution owners. The testing is performed against test cases.

Trial specific requirements are discussed in section 3.2.1, and corresponding test-cases are discussed in section 3.2.3.

3.2.1 Trial specific Requirements

Trial specific requirements are generated by Step 2.1. The following types of requirements need to be considered (list is not restrictive):

- **Interface requirements**
  - **Scenario data;** Scenario data are Trial specific. At this stage, the types of operational data which are going to be used in the scenario are already known. A solution needs to consider at least some of the scenario data types (e.g. ambulances and victims). The scenario data requirements define which types shall be processed by each solution and under which format it shall be exchanged, or under what appearance it should be displayed (e.g. symbols).
  - **Trial specific Test-bed integration;** these requirements concern the integration of the solutions with aspects of the Test-bed which are Trial specific: mostly the data schemas.
- **Adaptation;** these requirements concern the changes in the solution which are required by the Trial Committee as a condition for its participation in the Trial. It does not include the adaptation needed to interface the Test-bed which is addressed implicitly by the Test-bed integration. The requested adaptation can concern minor functional aspects or the ability to import, export, and/or process certain types of data related to the Trial scenario. These adaptation requirements are discussed and agreed with the solution provider to ensure they are technically feasible and compatible with the available resources and time. These requirements are taken into account by Step 1.
- **Deployment;** these requirements concern the ability to deploy the technical set-up in the hosting environment of the Trial. This may concern aspects like: availability of local network, mobility (portable devices), availability of internet connections (with a given bandwidth), availability of large screens, etc.). These requirements must be fulfilled by the Platform (Task: "Event Logistics and Platform adaptation" of each Trial). These requirements are tested in Step 2.2.
- **Performance;** these requirements concern the expected performance of the solutions or the technical set-up (e.g. related to time response, processing capacity). These requirements can be

tested in Step 1 if they concern a standalone solution, or in Step 2.2 if their concern multiple solutions.

Section 3.5.5 gives examples for each type of requirements (Trial specific and generic) and gives advice regarding the step during which these requirements may be tested. Whenever the testing can be done in several steps (depending on the context) the Test plan will precise the step.

Safety and Security requirements concern the safety and security aspects (e.g. ability to process restricted information). These requirements will not be implemented in the Trial set-up but discussed in D934.31 (10). The way they are going to be addressed is discussed in section 2.5.2.

Some types of requirements can be tested either in Step 1 or Step 2 depending on the context; The step(s) in which each shall be performed can be managed by the solution coordinator via the Test-plan (section 2.3.).

3.2.2 Preparation of test data sets

In order to be able to perform the test, some data sets need to be prepared:

- Input data: generate messages which are compliant with the information schema(s) which is/are specified for the inputs solution during Trial (1 data set per schema, for each data set: messages containing all data types with some instances of each type).
- Output data: generate sets of data in internal format defined by the solution. Containing all types of data which need to be sent out by the solution. The output message generated by the solution is checked by the Test-bed schema validation service.
- Processed data: data similar to the type which is going to be processed during the Trial.

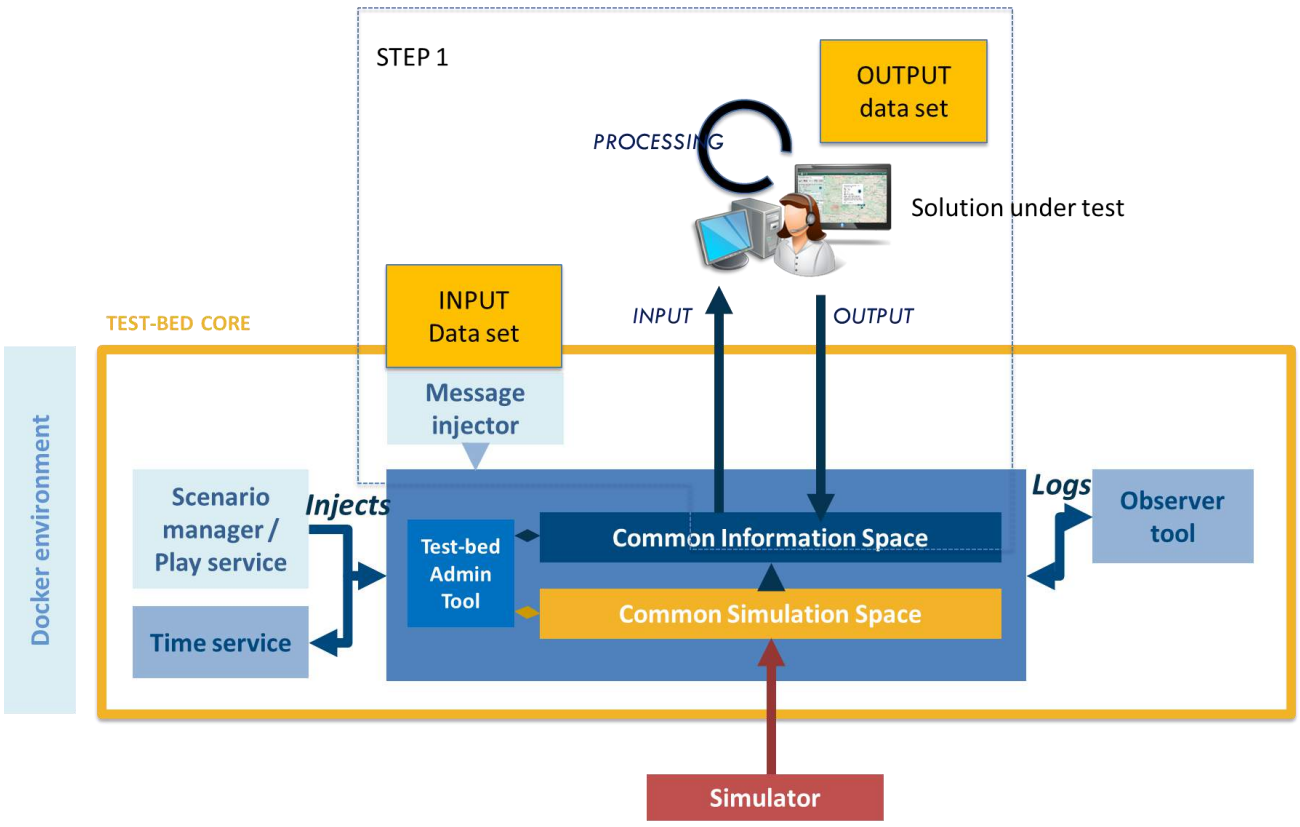

Figure 3.4: Step 1 scope and data sets

3.2.3 Integrate and test solutions against requirements

During Step 1 (Standalone solution Trial specific integration and testing) the testing activity is based on **test cases** which are testing the Trial specific requirements.

This integration process is structured as follows:

1. Define the schemas (AVRO schema(s)) which need to be used by the solution for its information exchange (in accordance with the Information Workflow and the data requirements).
2. Adapt solution (if required).
3. Pass the test cases corresponding to the solution's functional requirements (user stories), correct errors, and iterate until the defined Test cases are successfully passed, (correctly structured messages are sent by solution).

3.3 Step 2.1: Design of technical set-up

This section describes in more detail Step 2.1: Design of technical set-up. This step consists in technical set-up design and produces the Trial specific requirements for Step 1.

3.3.1 Technical set-up

The Trial technical set-up is the set components (solutions, Test-bed, simulators, measuring tools), interacting together in a way that will, at Trial time, enable them to support the execution of the Trial and the data collection.

The technical set-up is characterised by the list of its components, the way they are exchanging information (information workflow), and the way these components are deployed on the infrastructure (physical view) and on the organization (organizational view).

The design of the technical set-up is performed by producing these views. As usual the work on concrete examples (use cases) helps elaborating the views.

As most solutions are not fully mature, and the focus is more on functional characteristics, security and safety requirements are taken into account in a specific manner which is discussed in section 2.5.2.

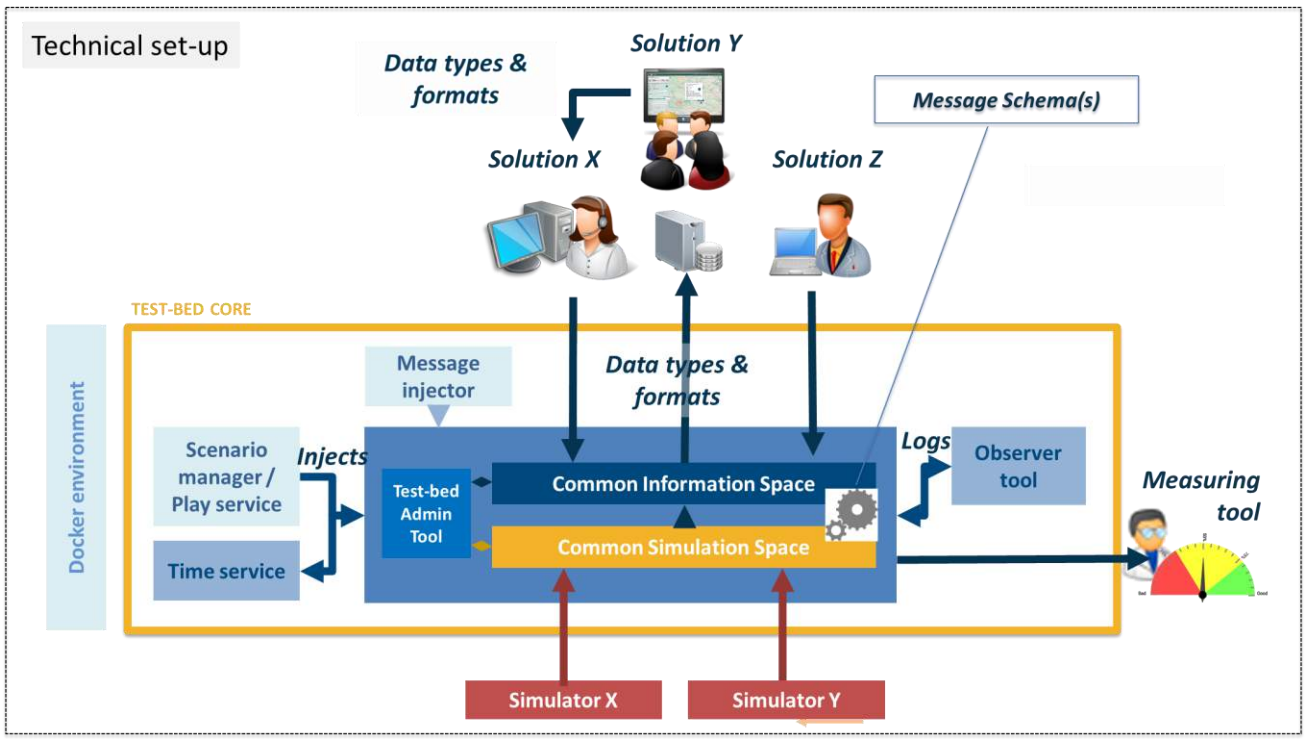

Figure 3.5: Trial technical set-up

As shown in Figure 3.5, although the Test-bed plays a central role in the technical set-up, the technical setup can contain interfaces between solutions, simulator or measuring tools which will not be connected via the Test-bed.

The following aspects are not part of the technical set-up:

- The full set of scenario data (e.g. maps, full set of available resources).
- The part of solutions which do not provide any software but focus on enhanced procedures, processes or organization structures.

Non-IT solutions need to be taken into account in the technical setup (at least in the information workflow) as long as they exchange information with the IT solutions[2]. Otherwise they are of no concern to this technical procedure.

The following sections provide examples of the various possible views.

3.3.2 Information workflow

The information workflow describes the way the various components of the technical set-up (i.e. solution, Test-bed, operational entity, simulator, measuring tools) must exchange information in order to be able to later support the Trial.

The questions to be answered by the Information workflow are:

- What component will this component need to exchange information with?
- What type of data will be exchanged (e.g. tactical situation)?
- What technical protocol will be used for the technical connection (e. g. Test-bed connection).
- What message format (i.e. schema) will be used? (e.g. EMSI).

[2] In any case, non-IT Solutions do have to be described in the PoS and their user stories and associated test cases also need to be documented there.

The kinds of activities represented in the information workflow can be one of the following:

- Operational activities.
- Simulated operational activities.
- Measurement activities.

Only the technical view of the information workflow is relevant to this solution testing procedure.

3.3.3 Example of Trial 1

Figure 3.6 shows an information workflow from Trial 1 preparation. In this figure, the solutions are: Socrates OC, SGSP Drone (legacy) Drones Rapid Mapping, 3Di flooding. XVR is a Simulator.
The information workflow (Figure 3.6) shows that:

- Drones Rapid Mapping and 3Di send map layers update messages through the Test-bed, which are consumed (for display) by Socrates OC server.
- 3Di uses elevation data.
- XVR and 3Di are exchanging Maps.
- XVR is sending the resource positions to Socrates OC server (through the Test-bed) using an EMSI (11) Gateway.
- Socrates OC server is sending information to clients located in Operations Centres.

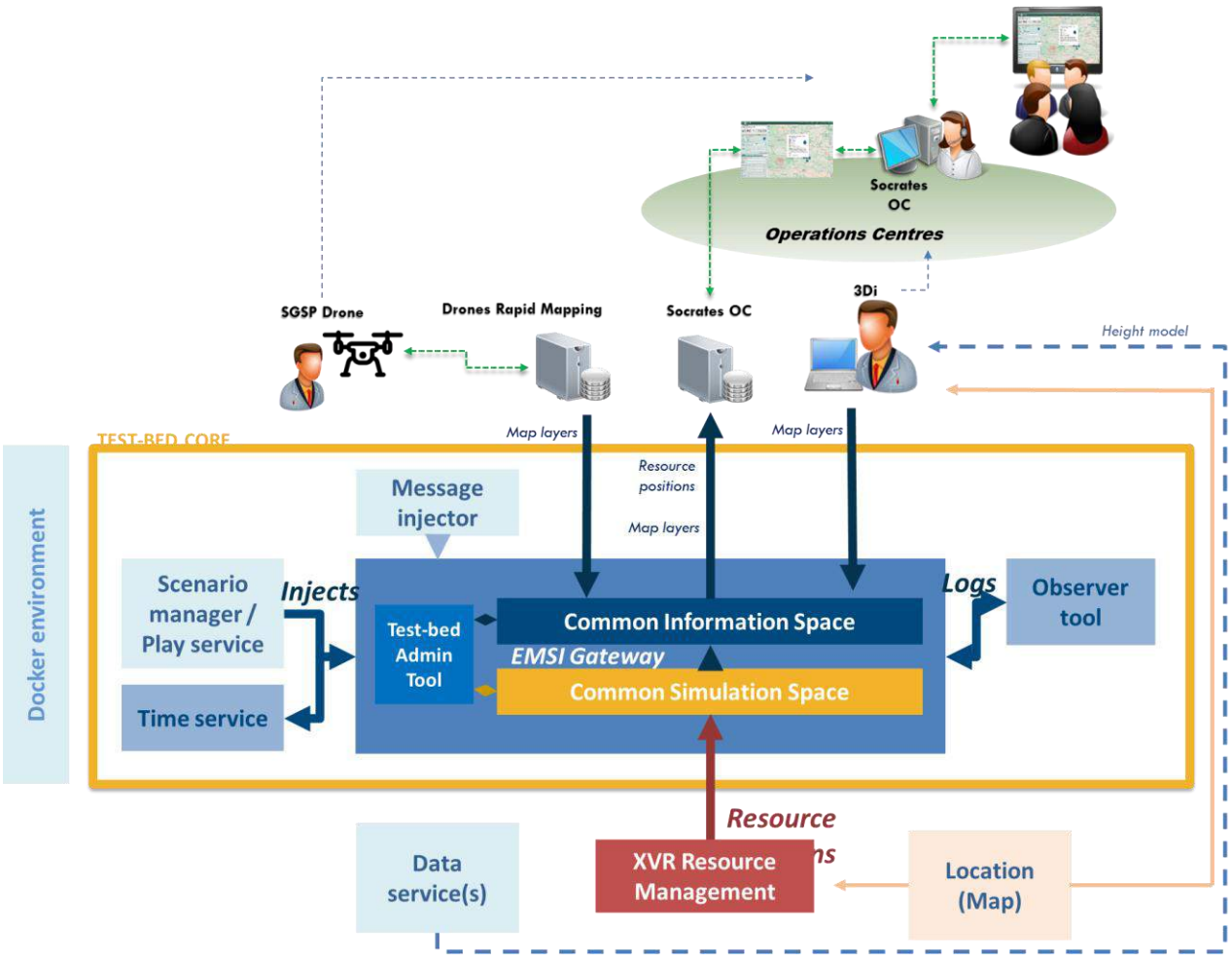

Figure 3.6: Trial 1 Information workflow

3.3.4 Example of Expe41

Table 3.1 shows a more technical representation of the interfaces between the solutions and involved in Experiment41 (12). The solutions are Asphodèle, LUPP, Large Event, Life-X COP, Crisis Wall and the XVR Simulator. This table describes the type or information that is exchanged, the technical protocol used to do so, and the format in which data is exchanged.

In this table, the "Format" column corresponds to the schema to be used for the exchange.

In Experiment 41, the Test-bed standard implementation was not available. This explains why the Test-bed is never mentioned in the protocol column. Whether the interface is performed via the Test-bed or not should be mentioned in the "Protocol" column.

Table 3.1 gives an example of the way these protocol and formats can be defined;

Table 3.1: Interfaces between solutions in EXPE41

Output	Input to	Data type	Protocol	Format
Asphodèle	Large Event	Tactical situation	drag and drop	KML
Asphodèle	Life-X COP	Tactical situation	Mail	KML
Asphodèle	LUPP	Tactical situation	Mail	PNG
LUPP	Large Event	Tactical situation	http REST	EDXL-DE + EMSI
LUPP	Life-X COP	Tactical situation	http REST	EDXL-DE + EMSI
Large Event	Crisis Wall	Alert	http REST	CAP
Large Event	Life-X COP	Base map	WMS	Technical: Large Event is the map server for Life-X COP
Large Event	LUPP	Base map	WMS	
Life-X COP	Crisis Wall	Alert	http REST	CAP
XVR Simulator	Large Event	Maps	Manual	Shape

The Interfaces requirements can be formulated as separate requirements, or just refer to a table similar to Table 3.1.

The fulfilment of the Information workflow is tested in Step 2.2 (section 3.3).

3.4 Step 2.2: Technical set-up integration and testing

This section describes the Step 2.2: technical set-up integration and testing. The integration work and testing done during this step involves multiple solutions and tests so that they are able to work together as requested by requirements. The whole technical set-up is tested (Figure 3.7).

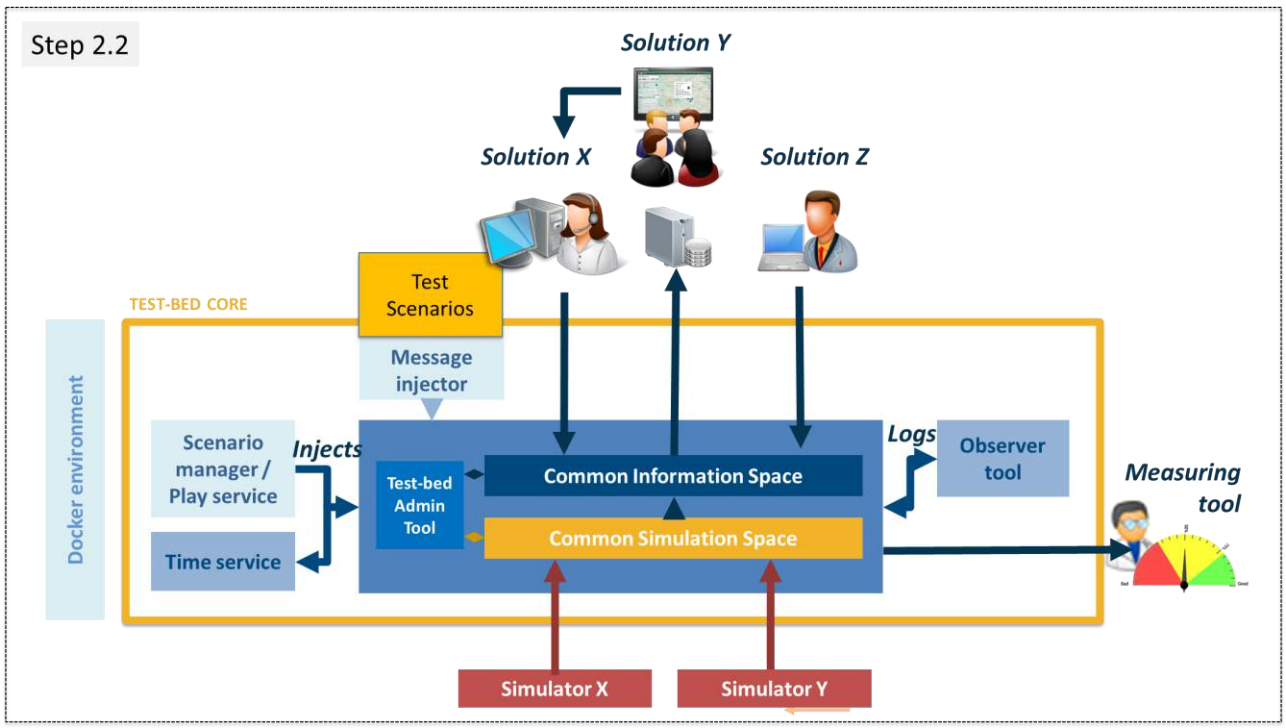

Figure 3.7: Step 2.2 scope

3.4.1 Requirements and data preparation

The requirements which are tested in Step 2.2 are the information workflow as a whole and all Trial specific data which require multiple solutions for their testing:

- The information workflow is tested against test scenarios (section 3.5.4).
- Other requirements are tested by test cases (section 3.5).

3.4.2 Integration and testing

The main tests, which need to be passed during Step 3 are the test scenarios. Guidelines about the test scenarios can be found in section 3.5.4. The Test-plan may also plan other tests for Dry run 1.

This step uses the Test-bed in its "Preparation phase" mode "In which all solutions and simulators are connected to the Test-bed, the scenario is built and the entire set-up is tested to assure a good run of the Trial." (13).

The high-level use-case: "Set-up and testing of the Test-bed" of the Test-bed is used to support this step (5).

At term, the Test-bed implementation will include the Scenario manager[3]. Test scenarios will be developed and run using the Scenario manager.[4]

The following sequence is followed:

- Install Test-bed on Platform hardware.

[3] "The scenario manager can be used to create scenarios (master event lists) that are injected, via the Test-bed, into the CSS or CIS. For example, it injects a message to start a flooding, to send out emails to participants, or to instruct a role-player to perform an act. Naturally, it can also control the time, and (re-)start/pause/stop a scenario." (5)

[4] "The test-scenario allows you to test the solution in a relevant context." (7)

- Install all solutions (and if needed: simulators, measuring tools), or deploy hardware where solution is already installed.
- Test installation of solutions and Test-bed.
- Connect solutions to local network (if needed).
- Connect solutions to internet (if needed).
- Pass test scenarios.
  - If a test scenario fails (and only if it fails) because of a solution, the test cases used for the individual integration and testing of this solutions might be re-passed by solution owner under the supervision the solution coordinator and Test-bed infrastructure coordinator, to detect and solve error before the end of Dry run1.
- Pass additional test cases as planned in Test-plan (e.g. test cases related to deployment requirements).
- Correct errors and repeat until all test scenarios' assessments are passed successfully.

If at the end of Dry run 1, in spite of actions taken during Dry run 1 to solve the issues, some tests are still not successfully passed, corrective actions may be decided and/or another Dry run 1, called Dry run 1 - v2 needs to be planned and conducted before Dry run 2.

3.5 Other guidelines

This section provides guidelines regarding the writing of requirements, user stories, test cases and test scenarios.

The notion of requirements and tests are key concepts of "traditional" system engineering in general and Software engineering in particular.

- Requirements are defined as "any necessary (or sometimes desired) function, attribute, capability, characteristic, or quality of a system for it to have value and utility to a customer" (14). There are several ways the requirements can be described. We consider two: the traditional **system engineering** requirements way and the **user story** (15) style which was introduced by the agile methodology (16).
- Testing is defined as "the process of validating and verifying that a software program/application/product: meets the business and technical requirements that guided its design and development; works as expected". (17). In short, tests are meant to check that requirements are met. For this reason, there is a correspondence between tests and requirements. Each test is meant to verify at least one requirement.

The following sections give guidelines for the writing of requirements and tests for this procedure.

3.5.1 How to write "traditional" requirements?

Traditional requirements and user stories both describe the features of a system. The main difference between them is a discrepancy of point of view: user stories express things from the point of view of the user, whereas traditional requirements express a neutral point of view. For this reason, it is often advised to prefer user stories for functional requirements (involving end-users), and to prefer traditional requirements for more technical aspects (i.e. non-functional requirements).[5]

However, the term "requirement" is used generically in the document, and except when dealing with the way how to formulate a requirement, there is no clear differentiation between "traditional requirements" and "user stories".

[5] "I often get asked by clients whether or not all of their requirements should be written as User Stories…Here is the advice that I give: Business requirements: Yes." (24)

The "traditional" way to write requirements in system engineering is to write an active sentence which says that the <system> can perform certain actions.

In DRIVER+, the solution <solution> shall enable < action>

Example:

"The SMAP solution shall enable the collection of posts from the social media based on their content"

The requirement "title" needs to be completed by the requirement "body" which gives the following information:

- The **preconditions** of the action in terms of user type, data, status of the solution, phase of work (e.g. an internet connection is available and a collection request is formulated by user who provides a list of keywords).
- The **processing required** by the action (e.g. collection is performed on the commercial interface of Twitter and Tweets corresponding to the request are collected).
- The **consequences** of the actions (e.g. collected Tweets are automatically stored in a corpus and indexed in order to enable search).

It is recommended that the formulation of the requirements follows the so called "SMART" criteria:

- Specific – target a specific area.
- Measurable – give success criteria.
- Assignable – assign it to a solution, or to the whole technical set-up.
- Realistic – state what results can realistically be achieved, given available resources.
- Time-related – specify when the result(s) should be achieved.

3.5.2 How to write user stories?

The concept of user stories comes from the Agile Methodology (16). Even though the Integration of solutions is not managed as a purely Agile process, the "user story" formalism can be used as a convenient way to express requirements. This formalism has the advantage of being easier to understand by end-users, because it is expressed from their point of view.

The syntactic rule is to write a sentence, which follows the pattern:

As a <type of user>, I want <some goal or objective >, so that <benefit, value>

Example:

"As a Firefighter I want the SMAP solution to collect information from social media based on provided lists of keywords, so that I can access additional sources of information to get a better picture of the situation at hand."

As for the "traditional requirement", the user story "title" needs to be completed by the following information: preconditions, processing, expected results.

Some examples of "user stories" entered in DRIVER+ PoS are shown in Annex 3.

3.5.3 How to describe test cases?

Test-cases are defined during Step 1 (section 2.5). Test-cases aim at making sure that the integration requirements are fulfilled. Test-cases are attached to a requirement. Each requirement is verified by one or more test cases. It is recommended in this procedure to have a maximum of three test cases per requirement. The requirement is considered as fulfilled when all associated test cased have been passed successfully.

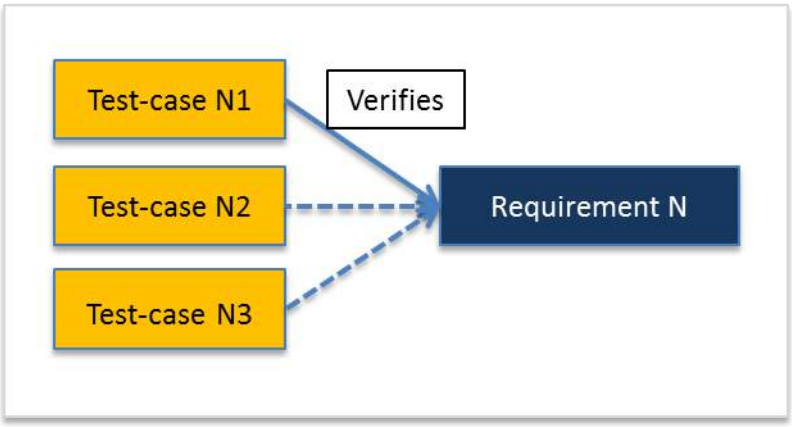

Figure 3.8: Test-cases verify requirements

A test case consists of a sequence of simple actions to be performed in certain conditions to obtain certain results.

A test case is described by the following:

- Some pre-conditions (concerning for example the status of the solution, the kind of user, etc.).
- A sequence of simple technical actions.
- The processing required by the actions (e.g. collection is performed on the commercial interface of Twitter, Tweets corresponding to the request are collected).
- The consequences of the actions (e.g. collected Tweets are automatically stored in a corpus and indexed in order to enable searching).
- Some success criteria (e.g. a number of Tweets containing relevant information about the situation has been successfully collected and made available to the system).

The technical actions to be performed during the test case depend on the requirement it is verifying. For example: an interface requirement will be tested by information exchange type of test cases, while a functional requirement will be tested by activating the considered function of the solution.

3.5.4 How to write test scenarios?

The Trial technical set-up is tested against test scenarios. Test-scenarios are derived from the Trial scenario. They are based on use cases: simplified fragments of the whole Trial story. Their writing does not require the Trial scenario to be completed, but the use cases to be identified and the Trial specific requirements to be known.

Test scenarios are of the same nature than Trial scenario, they can be managed by the Test-bed scenario manager.

Each Test scenario is built around a use case derived from the Trial scenario, and is both:

- **A fragment** of the whole Trial scenario (use case).
- **A technical interpretation** of the Trial scenario: for example, if the operational scenario says that the Field Command post sends its tactical situation to the regional level, this will be interpreted in the test scenario as (Field Command Post C2 system sends tactical situation to Regional level C2 system), where Tactical situation can for example be a set of tactical objects (e.g. ambulances, roadblocks, fire trucks, fire area, plume)
- **A reduction** of the Trial scenario**:** operations which might be repeated many times in the Trial Scenario need only to be tested two or three times (to avoid the "works only once bug"). For example: the sending of the tactical situation to the regional level which might be done every 30 minutes in Trial scenario, will only be tested twice, but with all the possible data types inside of it.

Together, all test scenarios must:

- Activate all the solutions functions which are going to be activated by the Trial scenario (at least once).
- Address all the types of data which will be addressed by the Trial scenario (but not necessarily as many instances).
- Contain all types of information exchanges which will be performed during the Trial scenario.

Example of a test-scenario:

1. *Plume is requested to technical experts (to evaluate the future dispersion of a chemical)*
2. *Plume is received, and included in COP*
3. *Plume is visible from all Command Posts where COP is deployed*
4. *Decision to contain area is village is mentioned in COP daybook, with link to plume)*
5. *Containment area is drawn on COP map.*
6. *Other Commands Posts can see containment area and read Decision relative to containment in handbook)*

3.5.5 When to test which requirements?

This section provides examples of requirements and provides guidelines regarding the step at which a test may be planned in the Test plan.

Table 3.2: Examples of requirements of various types and planning

Type of requirement	Tested in Step	Recommended style	Example
Test-bed integration	Step 0 for Standard implementation Step 1 if Trial specific Test-bed.	Traditional requirement.	Whether a certain information exchange will be done through or outside the Test-bed is fixed by the information workflow. The integration process with the Test-bed is described in section 3.1.
Other Interfaces	Step 2.	Traditional requirement.	The Emergency Health Service dispatching solution sends the location of an ambulance, and its destination to the traffic Simulator which in return sends back the updated location of the ambulance every ten seconds, for the dispatching system, and the COP to display them on a map (to enable tracking of ambulances).
Scenario data	Step 1 for solution internal processing of display. Step 2 for information exchange.	Traditional requirement.	The Command and Control system enables to manage and display the following data types: ambulances, victims, survivors, fire trucks, fire bombers, roadblocks, chemical plume, traffic jams.
Main functions (called "user stories")	Step 0.	User story (if functional).	Requirement ("User story")[6]: **Solution Reference:** CrowdTasker. **Title of US:** Gather information from the field by distributing tasks. **Text of US:** As a Tactical level actor from CM organisation, I want to get accurate, relevant information directly from the concerned area by approaching relevant citizens/volunteers based on their skills and/or location. **So that:** So that I can improve the decision support and are able to better plan the resources or priorities to efficiently manage the crisis.

[6] The complete example can be found in its PoS version in Annex 3.

Type of requirement	Tested in Step	Recommended style	Example
Adaptation	Step 1.	User story (if functional).	**Requirement:** As an Incident manager working at Field level, the system enables me to send the current tactical situation as a map layer which the Common Operational Picture will be able to display as a layer on its common map. **Comment:** This capacity is an evolution that shall be implemented and demonstrated before <Date>.
Measurement and data logging	Step 1 if can be tested with standalone solution. Step 2 if testing requires multiple solutions.	Traditional requirement.	The COP system shall enable the logging and the export of the following information of its daybook for later analysis: text, authority, date.
Deployment	Step 2.	Traditional requirement.	**Example 1:** Requirement: the incident management solution shall be deployable on tablet devices. **Example 2:** As the COP solution application will run on a distant server, the Platform shall provide a 1Gb/s internet connection.
Performance	Step 1 if can be tested with standalone solution. Step 2 if testing requires multiple solutions.	Traditional requirement.	The social media analysis solution collects and processes the tweets during three months on certain topics of interest. This requires the ability to store and process at least three million tweets.

4. Mapping of solutions

This section provides the mapping of the current DRIVER+ PoS on the CM Function taxonomy (2). Only the Internal solutions are currently described in the PoS. Each solution is associated with a set of CM functions according to the DRIVER+ CM functions taxonomy (2).

There are currently 14 internal solutions in the Portfolio of Solutions. These solutions are proposed by the DRIVER+ partners. External solutions will be added to the Portfolio of Solutions once they have been selected for one of the trials and properly documented. The 14 solutions and their providers are listed in Table 4.1.

Table 4.1: CM solutions provided by consortium partners

Solution name	Solution provider
Social Media Analysis Platform	TCS
Airborne and terrestrial situational awareness	DLR
CrowdTasker	AIT
Humlog	WWU
Psychological First Aid	DRC
SOCRATES (C3…)	GMV
Rumour Debunker	AIT
Life-X COP	FRQ
MDA Command and Control system	MDA
GDACS mobile	WWU
Protect	EDI
IO-DA	ARMINES
PROCEED	ITTI
Debris management	DWR

Each solution has been associated with or mapped to one or more functions from the Taxonomy of CM functions (D934.10) by the respective solution provider, in communication with and assisted by the research team (in particular the team from CSDM lead beneficiary in the design of the Taxonomy of CM functions, which guaranteed that the concepts underlying the design of the Taxonomy were taken into account in the mapping process).

The full mapping of these solutions against the Taxonomy of CM functions can be found in Annex 3. This table shows that the 14 solutions relate to 63 CM functions. Table 4.2 shows that only seven of these 63 CM functions are addressed more than once. Only three of the solutions do not address any of these seven functions, i.e. they have a more specific purpose. Five of these seven CM functions are part of the “Response” functional area of the Taxonomy, and the other two of the functional area “Crisis Communications and Information Management” (CCIM).

Table 4.2: Main CM functions addressed by current PoS solutions

Functional area	Taxonomy category (CM function)	Social Media Analysis Platform	Airborne and terrestrial situational awareness	CrowdTasker	Humlog	Psychological First Aid	SOCRATES (C3…)	Rumour Debunker	Life-X COP	MDA Command and Control system	GDACS mobile	Protect	IO-DA	PROCEED	Debris management	Sum
Response	Maintain shared situational awareness	x	x				x		x			x				5
Response	Conduct damage and needs assessment		x						x		x					3
Response	Provide decision support		x		x								x			3
Response	Conduct coordinated tasking and resource management			x			x						x			3
Response	Manage organized volunteers			x		x										2
CCIM	Provide for crowd sourcing	x									x					2
CCIM	Detect and debunk deception and rumours in social media	x						x								2

The full mapping of PoS solutions on the Taxonomy of CM functions (Annex 3) shows that, on average, a solution addresses 2.9 different Functional Areas (i.e. the first level grouping in the Taxonomy). However, three of the solutions (Psychological First Aid, Rumour Debunker, and IO-DA) aim to address only one functional area.

Table 4.3: Distribution of solutions and CM functions against functional areas

Area #	Functional area	Number of solutions	Number of CM functions covered
1	Mitigation	2	2
2	Capability development	5	12
3	Strategic adaptiveness	1	1
4	Protection	6	6
5	Response	11	24
6	Recovery	4	5
7	Crisis communications and information management	6	7

Area #	Functional area	Number of solutions	Number of CM functions covered
8	Command, Control and Coordination	3	3
9	Logistics	3	3
10	Security management	0	0

Table 4.3 shows that "Response" is clearly the functional area that is most frequently addressed, and with the highest diversity in terms of number of CM functions covered. It is followed by the functional areas "Capability development" and "Crisis communications and information management". All but one of the functional areas ("Security management") is addressed by at least one of the 14 solutions.

Analysis of this type, with the respective visualization, can serve a number of purposes.

It demonstrates that solutions to be tested in the series of DRIVER+ trials provide new and/or enhanced crisis management functionalities. This can be visualized using the Taxonomy of Crisis Management functions (D934.10), as in Figure 4.1, which represents the functionalities of solutions, provided by members of the DRIVER+ consortium. (Table 4.4 below the following two figures explains the notation Fx.y, i.e. the first-level functions in the Taxonomy. Most of these functions have sub-functions, and many of them include a set of subordinated tasks.)

Figure 2.1 illustrates the mapping between CM gaps addressed in Trial 1 and the set of solutions provided by members of the DRIVER+ consortium.

This illustration, adapted for example to map the set of solutions (both internal for the DRIVER+ consortium and external), selected for a particular Trial to the crisis management gaps addressed by this Trial, can support a number of objectives.

First, it may facilitate the analysis whether the set of solutions, selected for the particular trial, covers all functionalities related to the gaps, which the Trial aims to address. If that is not the case, this analysis will identify additionally required functionalities and, respectively, may trigger a search for additional solutions of relevance to the practitioners' needs.

Second, it may be used to identify functionalities of the solutions that match best the practitioners' needs i.e. identify those functionalities that fit missing or inefficiently performed functionalities to which the description of a gap relates. This may be used to prioritise and focus the efforts of the consortium, e.g. in the development of test cases, investments in integrating the solutions into the DRIVER+ Test-bed and evaluating their performance during the trial.

Third, it allows identifying functionalities of solutions that do not represent a particular interest for the practitioners and, hence, the consortium does not need to invest time and effort into their detailed testing.

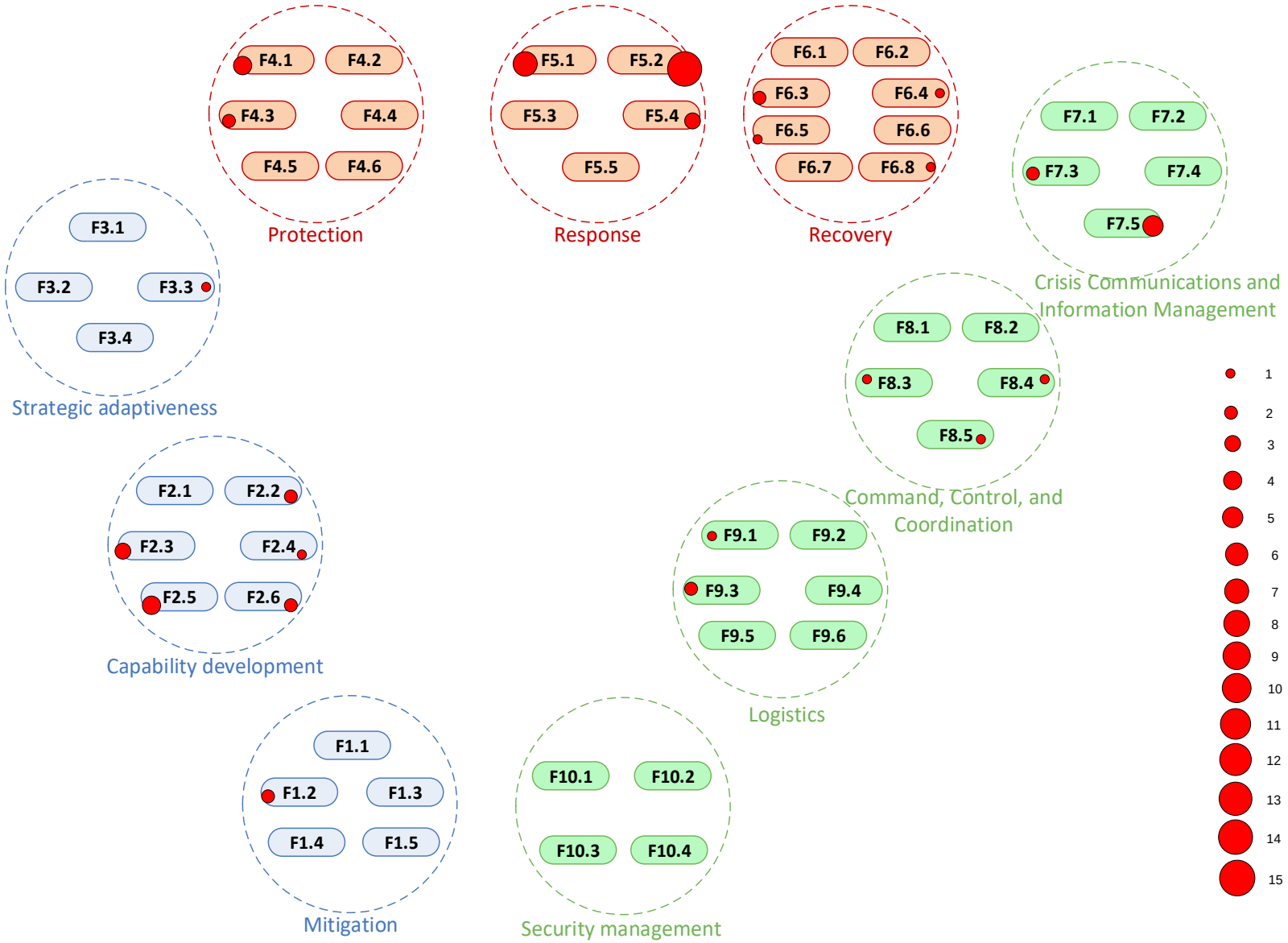

Figure 4.1: Mapping the consortium internal solutions to the taxonomy of CM functions

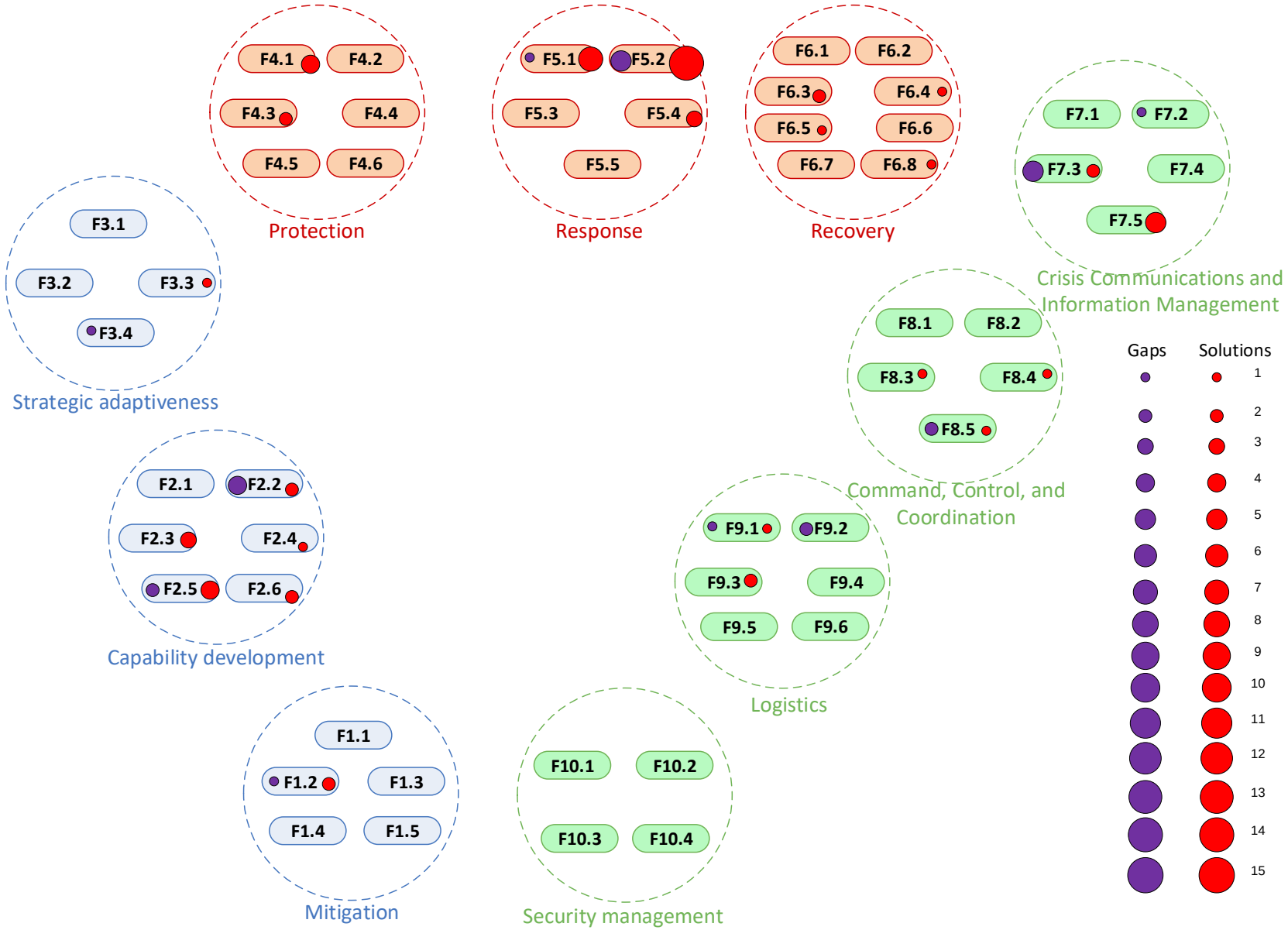

Figure 4.2: Mapping the Trial 1 gaps and consortium internal solutions to the taxonomy of CM functions

Table 4.4: Taxonomy fields and descriptions.

Func-tion	Description	Function	Description
Preparatory Functional Area "**MITIGATION**"		Preparatory Functional Area "**CAPABILITY DEVELOPMENT**"	
F 1.1	Organise for mitigation	**F 2.1**	Plan for CM capabilities
F 1.2	Assess the risks	**F 2.2**	Manage CM system of systems development
F 1.3	Elaborate mitigation policy and strategy	**F 2.3**	Manage human resources
F 1.4	Implement mitigation measures	**F 2.4**	Organise for crisis management
F 1.5	Keep the mitigation strategy relevant	**F 2.5**	Establish CM doctrine and train organisations and people
		F 2.6	Establish a CM lessons learning system
Preparatory Functional Area "**STRATEGIC ADAPTIVENESS**"		Operational Functional Area "**PROTECTION**"	
F 3.1	Promote CM organisational agility	**F 4.1**	Conduct systematic monitoring and data collection
F 3.2	Conduct civil security foresight	**F 4.2**	Conduct operational planning
F 3.3	Develop capacity to adapt	**F 4.3**	Conduct incident/emergency response below the level of "crisis"
F 3.4	Build and measure community resilience	**F 4.4**	Coordinate and provide public protection
		F 4.5	Protect critical infrastructures
		F 4.6	Coordinate and provide CII protection
Operational Functional Area "**RESPONSE**"		Operational Functional Area "**RECOVERY**"	
F 5.1	Orient and decide	**F 6.1**	Adjust the recovery planning
F 5.2	Respond to the hazard	**F 6.2**	Provide immediate relief support

Func-tion	Description	Function	Description
F 5.3	Limit the impact of the crisis	**F 6.3**	Engage the population
F 5.4	Support affected people	**F 6.4**	Manage humanitarian recovery
F 5.5	Build the ground for relief and recovery	**F 6.5**	Recover public lifelines
		F 6.6	Manage economic recovery
		F 6.7	Manage infrastructure recovery
		F 6.8	Manage environmental recovery
Common Functional Area "**CRISIS COMMUNICATIONS AND INFORMATION MANAGEMENT**" (CCIM)		Common Functional Area "**COMMAND, CONTROL, AND COORDINATION (C3)**"	
F 7.1	Establish CCIM organisation	**F 8.1**	Build and maintain the C3 system
F 7.2	Conduct and coordinate communications and information planning	**F 8.2**	Establish the command component
F 7.3	Create CCIM networks	**F 8.3**	Establish the control component
F 7.4	Continuously improve CCIM	**F 8.4**	Establish the coordination component
F 7.5	Exploit CCIM for protection, response, and recovery	**F 8.5**	Exploit the C3 system
Common Functional Area "**LOGISTICS**"		Common Functional Area "**SECURITY MANAGEMENT**"	
F 9.1	Establish crisis logistics management system	**F 10.1**	Conduct security orientation and planning
F 9.2	Manage materiel logistics	**F 10.2**	Establish security management organisation
F 9.3	Conduct transportation logistics	**F 10.3**	Provide key security capabilities
F 9.4	Provide medical logistics	**F 10.4**	Exercise on-site security control
F 9.5	Manage facilities		
F 9.6	Provide logistics services		

5. Conclusion and way forward

The solution testing procedure described in this document covers the integration and testing activities of standalone solutions (SP93 Portfolio of CM Solutions) and technical set-up of a Trial as a whole (SP94 Trials). It will help technical stakeholders of the integration process coordinate their work and facilitate the coordination between SP93 and SP94, up to Dry run 1 of each Trial.

This procedure is designed to be supportive, it is mainly descriptive, and gives orientations and recommendations. The solution coordinators will be able to customise it in agreement with the Trial Committee to take into account her/his own familiar methods, the specific constraints of the Trial, which can concern for example the schedule, the nature of the technical set-up, or the participating solutions themselves.

This procedure is purely technical, it does not address the actual assessment of the solutions which will be performed during the Trial, but it enables it.

The mapping of the current content of the PoS against the Taxonomy of crisis management functions for the classification of solutions (2) shows how internal solutions are aligned with the DRIVER+ gaps (3).

Both guidelines and mapping of solutions are inputs for potential evolutions of the PoS website. This procedure is based on the experience of the past DRIVER experiments and will benefit from the DRIVER+ Trials. At terms, it may provide a contribution to an updated version of the TGM (1).

References

1. *Trial Guidance Methodology.* s.l. : DRIVER+ project, 2018. D922.21.

2. *Taxonomy of Crisis Management functions for classification of solutions.* s.l. : DRIVER+ project, 2018. D934.10.

3. *List of CM gaps.* s.l. : DRIVER+ project, 2018. D922.11.

4. *4.2 - Use case : Integration process for a single solution or simulator.* s.l. : DRIVER+, 2018. pp. 30-31. D923.21.

5. *Functional specification.* s.l. : DRIVER+ project, 2018. D923.11.

6. Task 934.2: Solution adaptation. *DRIVER+ Description of Work.* s.l. : DRIVER+ project, 2018, p. 298.

7. *Reference implementation (v1, v2, v3).* 2018. D923.21, D923.22, D923.23.

8. Integration process. *DRIVER+ Test-bed reference implementation Github.* [En ligne] 2018. https://github.com/DRIVER-EU/test-bed#integration-process.

9. Test-bed integration. *DRIVER+ Slack channels.* [En ligne] 2018. https://driver-eu.slack.com/.

10. *DRIVER+ Solution scenarios and integration test results v1.* s.l. : DRIVER+ project, 2018. D934.31.

11. ISO. *Societal security — Emergency management — Message structure for exchange of information.* 2015. 22351.

12. *Experiment 41 design & report.* s.l. : DRIVER+ project, 2018. D934.11.

13. Use modes. *Functional specification of the Test-bed.* s.l. : DRIVER+ project, 2018, 4.4.1, p. 29.

14. Wikipedia. Requirements. [En ligne] https://en.wikipedia.org/wiki/Requirement.

15. —. User stories. [En ligne] https://en.wikipedia.org/wiki/User_story.

16. Agile software development. *Wikipedia.* [En ligne] https://en.wikipedia.org/wiki/Agile_software_development.

17. Introduction to software engineering / Testing. *Wikibooks.* [En ligne] https://en.wikibooks.org/wiki/Introduction_to_Software_Engineering/Testing.

18. Wikipedia/UML. [En ligne] https://en.wikipedia.org/wiki/Unified_Modeling_Language.

19. UML Website. [En ligne] https://www.uml-diagrams.org/use-case-diagrams.html .

20. The Complete Guide to UML Diagram Types with Examples. [En ligne] https://creately.com/blog/diagrams/uml-diagram-types-examples/.

21. Wikipedia/BPML. [En ligne] https://en.wikipedia.org/wiki/Business_Process_Model_and_Notation.

22. Object Management Group. [En ligne] www.bpmn.org.

23. VisualParadigm. Business Process Model and Notation. *Youtube.* [En ligne] 2011. www.bpmn.org https://www.youtube.com/watch?v=m4eRGv-CVwI.

24. Scrum organisation. Should all requirement be written as user stories ? *Forum.* [En ligne] https://www.scrum.org/forum/scrum-forum/5418/should-all-requirements-be-written-user-stories.

25. *4.3- Pre-trial integration testing.* s.l. : DRIVER+ project, 2018. p. 33. D923.21.

Annexes

Annex 1 – DRIVER+ Terminology

In order to have a common understanding within the DRIVER+ project and beyond and to ensure the use of a common language in all project deliverables and communications, a terminology is developed by making reference to main sources, such as ISO standards and UNISDR. This terminology is presented online as part of the Portfolio of Solutions and it will be continuously reviewed and updated[7]. The terminology is applied throughout the documents produced by DRIVER+. Each deliverable includes an annex as provided hereunder, which holds an extract from the comprehensive terminology containing the relevant DRIVER+ terms for this respective document.

Table A1: DRIVER+ Terminology

Terminology	Definition	Source
Dry run 1	First rehearsal of a Trial, focusing on the technical integration of solutions, reference implementation of the Test-bed, and scenario validation; it also serves as a readiness review to approve the maturity of technical solutions.	Initial DRIVER+ definition.
Dry run 2	Full scale rehearsal of a Trial without external end-users participation, aimed at detection of technical issues and last second fine-tuning; Dry Run 2 is organised as a complete mirror of the Trial.	Initial DRIVER+ definition.
Gap	Gaps between the existing capabilities of responders and what was actually needed for effective and timely response.	Project Responder 5.
System function	Broad category of activity performed by a system.	ISO 6385:2016(en) Ergonomics principles in the design of work systems, 2.21.
Test-bed	The software tools, middleware and methodology to systematically conduct Trials and evaluate solutions within an appropriate environment. An "appropriate environment" is a testing environment (life and/or virtual) where the trialling of solutions is carried out using a structured, all-encompassing and mutual learning approach. The Test-bed can enable existing facilities to connect and exchange data, providing a pan-European arena of virtually connected facilities and crisis labs where users, providers, researchers, policy makers and citizens jointly and iteratively can progress on new approaches or solutions to emerging needs.	Initial DRIVER+ definition.
Trial Action	The main Trial planning document, facilitating	Initial DRIVER+ definition.

[7] Until the Portfolio of Solutions is operational, the terminology is presented in the DRIVER+ Project Handbook and access can be requested by third parties by contacting coordination@projectdriver.eu.

Terminology	Definition	Source
Plan (TAP)	collaborative planning and supporting combined execution. It covers all areas related to the Trial organization and will be used to record efforts, circulate decisions and assess progress.	
Portfolio of Solutions (PoS)	A database driven web site that documents the available Crisis Management solutions. The PoS includes information on the experiences with a solution (i.e. results and outcomes of Trials), the needs it addresses, the type of practitioner organisations that have used it, the regulatory conditions that apply, societal impact consideration, a glossary, and the design of the Trials.	Initial DRIVER+ definition.
Scenario	Pre-planned storyline that drives an exercise; the stimuli used to achieve exercise objectives [pre-planned storyline that drives an exercise (3.83), as well as the stimuli used to achieve exercise project performance (3.167) objectives (3.153)]	ISO22300 (2015) 9 [DRAFT 2017, p 27].
Trial	An activity for systematically finding and testing valuable solutions for current and emerging needs in such a way that practitioners can do this in a pragmatic yet systematic way.	
Domain / Area	Broad operational categories in which similar needs are consistently identified.	Project Responder 5.

Annex 2 – Mapping of current PoS solution of CM functions taxonomy

This Annex presents the complete mapping of the current PoS solutions (provided by partners in the DRIVER+ consortium) against the Taxonomy of CM functions. In Table A2 the relevant taxonomy functions are listed by the number of solutions, which address them. Table A3 shows the same mapping, but the functionalities of the solutions are ordered by functional areas of the Taxonomy of CM functions (D934.10).

Table A2: CM functions addressed by internal solutions, ordered by frequency.

Ref #	Functional area	Taxonomy category	Social Media Analysis Platform	Airborne and terrestrial situational awareness	CrowdTasker	Humlog	Psychological First Aid	SOCRATES (C3…)	Rumour Debunker	Life-X COP	MDA Command and Control system	GDACS mobile	Protect	IO-DA	PROCEED	Debris management	No. of solutions addressing the function
5.2.2	Response	Maintain shared situational awareness	x	x				x		x			x				5
5.1.2	Response	Conduct damage and needs assessment		x						x		x					3
5.1.3	Response	Provide decision support		x		x								x			3
5.2.3	Response	Conduct coordinated tasking and resource management			x			x						x			3
7.5.6	CCIM	Detect and debunk deception and rumours in social media	x						x								2
5.2.4.4	Response	Manage organized volunteers			x		x										2
7.3.1.3	CCIM	Provide for crowd sourcing	x									x					2
1.2.1.2	Mitigation	Map the hazards per geographic area								x							1

Ref #	Functional area	Taxonomy category	Social Media Analysis Platform	Airborne and terrestrial situational awareness	CrowdTasker	Humlog	Psychological First Aid	SOCRATES (C3…)	Rumour Debunker	Life-X COP	MDA Command and Control system	GDACS mobile	Protect	IO-DA	PROCEED	Debris management	No. of solutions addressing the function
1.2.5	Mitigation	Estimate cascading effects													x		1
2.3.2.3	Capability development	Task volunteers			x												1
2.2.4	Capability development	Develop decision support systems				x											1
2.4.3	Capability development	Identify and analyse bottlenecks				x											1
2.3.2	Capability development	Manage volunteers									x						1
2.3.2.5	Capability development	Establish organization for spontaneous volunteers			x												1
2.5.4	Capability development	Certify personnel training and education													x		1
2.5	Capability development	Establish CM doctrine and train organisations and people													x		1
2.2.5	Capability development	Establish resource management and mutual aid system											x				1
2.6.1	Capability development	Develop after-action and lessons learned reporting											x				1
2.6.2	Capability development	Provide cross-border learning											x				1
2.5.3.1	Capability development	Develop and conduct all-hazards training													x		1
2.5.3.3	Capability development	Develop hazard-specific simulations and conduct													x		1

Ref #	Functional area	Taxonomy category	Social Media Analysis Platform	Airborne and terrestrial situational awareness	CrowdTasker	Humlog	Psychological First Aid	SOCRATES (C3…)	Rumour Debunker	Life-X COP	MDA Command and Control system	GDACS mobile	Protect	IO-DA	PROCEED	Debris management	No. of solutions addressing the function
		CAX															
3.3.1.2	Strategic adaptiveness	Estimate resource requirements				x											1
4.1.1	Protection	Conduct monitoring and anticipation		x													1
4.1	Protection	Conduct systematic monitoring and data collection			x												1
4.1.2.3	Protection	Maintain public awareness on hazards and respective services			x												1
4.3.1	Protection	Detect pending emergencies and provide early warning								x							1
4.3	Protection	Conduct incident or emergency response									x						1
4.1.2.2	Protection	Provide predictive analysis and situational awareness													x		1
5.2.2.2	Response	Develop and sustain COP									x						1
5.1.1	Response	Determine nature of the crisis		x													1
5.2.1	Response	Activate crisis management bodies									x						1
5.2.4.5	Response	Manage spontaneous volunteers					x										1

Ref #	Functional area	Taxonomy category	Social Media Analysis Platform	Airborne and terrestrial situational awareness	CrowdTasker	Humlog	Psychological First Aid	SOCRATES (C3…)	Rumour Debunker	Life-X COP	MDA Command and Control system	GDACS mobile	Protect	IO-DA	PROCEED	Debris management	No. of solutions addressing the function
5.2.4.3	Response	Deploy first responders									x						1
5.4.5	Response	Provide off-site health and MHPSS services					x										1
5.4.1	Response	Conduct SAR operations		x													1
5.4.7	Response	Provide MHPSS					x										1
6.3.2	Recovery	Organise volunteers and communities for recovery			x												1
6.3.1	Recovery	Maintain population's operational awareness										x					1
6.4.6	Recovery	Address needs of vulnerable populations	x														1
6.5.10	Recovery	Restore the solid waste collection system														x	1
6.8.4	Recovery	Remove damaged structures and debris														x	1
7.5.3	CCIM	Support C3 decision making						x									1
7.5.2.3	CCIM	Provide communications with volunteers										x					1
7.5.2.1	CCIM	Communicate operational information across chain of command											x				1
8.5.1	C3	Monitor the affected area										x					1

Ref #	Functional area	Taxonomy category	Social Media Analysis Platform	Airborne and terrestrial situational awareness	CrowdTasker	Humlog	Psychological First Aid	SOCRATES (C3…)	Rumour Debunker	Life-X COP	MDA Command and Control system	GDACS mobile	Protect	IO-DA	PROCEED	Debris management	No. of solutions addressing the function
8.3.3	C3	Determine principles of information exchange								x							1
8.4.4	C3	Establish trans-border coordination											x				1
9.1.1	Logistics	Identify components of crisis logistics support				x											1
9.3.1	Logistics	Plan, organize and resource transportation logistics		x													1
9.3.5	Logistics	Transport debris and waste														x	1
		Number of CM functions addressed by the solution	4	7	7	5	4	3	1	5	5	5	6	2	6	3	

Table A3: CM functions addressed by internal solutions, ordered by functional area.

Ref #	Functional area	Taxonomy category/solutions	Social Media Analysis Platform	Airborne and terrestrial situational awareness	CrowdTasker	Humlog	Psychological First Aid	SOCRATES (C3…)	Rumour Debunker	Life-X COP	MDA Command and Control system	GDACS mobile	Protect	IO-DA	PROCEED	Debris management	No. of solutions addressing the function
1.2.1.2	Mitigation	Map the hazards per geographic area								x							1
1.2.5	Mitigation	Estimate cascading effects													x		1
2.2.4	Capability development	Develop decision support systems				x											1
2.2.5	Capability development	Establish resource management and mutual aid system											x				1
2.3.2	Capability development	Manage volunteers									x						1
2.3.2.3	Capability development	Task volunteers			x												1
2.3.2.5	Capability development	Establish organization for spontaneous volunteers			x												1
2.4.3	Capability development	Identify and analyse bottlenecks				x											1
2.5	Capability development	Establish CM doctrine and train organisations and people													x		1
2.5.3.1	Capability development	Develop and conduct all-hazards training													x		1
2.5.3.3	Capability development	Develop hazard-specific simulations and													x		1

Ref #	Functional area	Taxonomy category/solutions	Social Media Analysis Platform	Airborne and terrestrial situational awareness	CrowdTasker	Humlog	Psychological First Aid	SOCRATES (C3…)	Rumour Debunker	Life-X COP	MDA Command and Control system	GDACS mobile	Protect	IO-DA	PROCEED	Debris management	No. of solutions addressing the function
		conduct CAX															
2.5.4	Capability development	Certify personnel training and education													x		1
2.6.1	Capability development	Develop after-action and lessons learned reporting											x				1
2.6.2	Capability development	Provide cross-border learning											x				1
3.3.1.2	Strategic adaptiveness	Estimate resource requirements				x											1
4.1	Protection	Conduct systematic monitoring and data collection			x												1
4.1.1	Protection	Conduct monitoring and anticipation		x													1
4.1.2.2	Protection	Provide predictive analysis and situational awareness													x		1
4.1.2.3	Protection	Maintain public awareness on hazards and respective services			x												1
4.3	Protection	Conduct incident or emergency response									x						1
4.3.1	Protection	Detect pending emergencies and provide early warning								x							1

Ref #	Functional area	Taxonomy category/solutions	Social Media Analysis Platform	Airborne and terrestrial situational awareness	CrowdTasker	Humlog	Psychological First Aid	SOCRATES (C3…)	Rumour Debunker	Life-X COP	MDA Command and Control system	GDACS mobile	Protect	IO-DA	PROCEED	Debris management	No. of solutions addressing the function
5.1.1	Response	Determine nature of the crisis		x													1
5.1.2	Response	Conduct damage and needs assessment		x						x		x					3
5.1.3	Response	Provide decision support		x		x								x			3
5.2.1	Response	Activate crisis management bodies									x						1
5.2.2	Response	Maintain shared situational awareness	x	x				x		x			x				5
5.2.2.2	Response	Develop and sustain COP									x						1
5.2.3	Response	Conduct coordinated tasking and resource management			x			x						x			3
5.2.4.3	Response	Deploy first responders									x						1
5.2.4.4	Response	Manage organized volunteers			x		x										2
5.2.4.5	Response	Manage spontaneous volunteers					x										1
5.4.1	Response	Conduct SAR operations		x													1
5.4.5	Response	Provide off-site health and MHPSS services					x										1
5.4.7	Response	Provide MHPSS					x										1

Ref #	Functional area	Taxonomy category/solutions	Social Media Analysis Platform	Airborne and terrestrial situational awareness	CrowdTasker	Humlog	Psychological First Aid	SOCRATES (C3…)	Rumour Debunker	Life-X COP	MDA Command and Control system	GDACS mobile	Protect	IO-DA	PROCEED	Debris management	No. of solutions addressing the function
6.3.1	Recovery	Maintain population's operational awareness										x					1
6.3.2	Recovery	Organise volunteers and communities for recovery			x												1
6.4.6	Recovery	Address needs of vulnerable populations	x														1
6.5.10	Recovery	Restore the solid waste collection system														x	1
6.8.4	Recovery	Remove damaged structures and debris														x	1
7.3.1.3	CCIM	Provide for crowd sourcing	x									x					2
7.5.2.1	CCIM	Communicate operational information across chain of command											x				1
7.5.2.3	CCIM	Provide communications with volunteers										x					1
7.5.3	CCIM	Support C3 decision making						x									1
7.5.6	CCIM	Detect and debunk deception and rumours in social media	x						x								2
8.3.3	C3	Determine principles of information								x							1

Ref #	Functional area	Taxonomy category/solutions	Social Media Analysis Platform	Airborne and terrestrial situational awareness	CrowdTasker	Humlog	Psychological First Aid	SOCRATES (C3…)	Rumour Debunker	Life-X COP	MDA Command and Control system	GDACS mobile	Protect	IO-DA	PROCEED	Debris management	No. of solutions addressing the function
		exchange															
8.4.4	C3	Establish trans-border coordination											x				1
8.5.1	C3	Monitor the affected area										x					1
9.1.1	Logistics	Identify components of crisis logistics support				x											1
9.3.1	Logistics	Plan, organize and resource transportation logistics		x													1
9.3.5	Logistics	Transport debris and waste														x	1
		Number of CM functions addressed by the solution	4	7	7	5	4	3	1	5	5	5	6	2	6	3	**63**

Annex 3 – Example user story and test case format in PoS

Annex 3 highlights the current structure for defining user stories (US) and test cases (TC) for per available solution. The examples below represent one UC and corresponding TC which are currently documented in the Portfolio of Solutions (PoS) tool for the CrowdTasker solution. The listed structure and description is directly exported from PoS using the provided "Export to Word" functionality. There, the necessary references to the dedicated PoS page can be found and also the last modification date to ensure traceability between exported or printed document representations.

The shown US and TC description is still under development and will be detailed and refined in the following months to reflect the necessary information and granularity explained in the guideline sections 3.5.2 and section 3.5.3.

Published on *Driver+ PoS* (https://driver-pos.atosresearch.eu)

Home > Crowdtasker US: Gather information from the field by distributing tasks

Crowdtasker US: Gather information from the field by distributing tasks

Solution Reference: CrowdTasker: Crowdtasking solution for managing of the pre-registered volunteers

User Story:

As a:

Tactical level actor

Operational level actor

From:

CM organisation

Volunteer Organisation

I want to:

I, as an organisation want to get accurate, relevant information directly from the concerned area by approaching relevant citizens/volunteers based on their skills and/or location.

So that:

So that I can improve the decision support and are able to better plan the resources or priorities to efficiently manage the crisis.

Process summary:

To get accurate, up-to-date information from an area concerned by a catastrophic event an organization uses CrowdTasker to approach citizens/volunteers in the area. To achieve this goal the organizations crisis manager defines a crisis event and one or more tasks, each in turn consisting of multiple steps, in the CrowdTasker backend. Volunteers located in the area of the event will receive notifications, and execute the tasks defined in the backend. The information that is sent back to the crisis manager and used to trigger an according crisis response.

Process description (Storyline):

Action:

The crisis manager defines an area for the crisis event, a start and (optional) end date for the event and desired volunteer skillset (spoken languages, medical skills...).

Expected Result:

Crisis event information is sent to all volunteers or only to volunteers in the desired area, if a specific flag is set.

Action:

The crisis manager defines one or more tasks. A task can either be chosen from an existing task template or defined from scratch. Tasks consist of a series of steps that should be conducted by the volunteers.

Expected Result:

Tasks are saved to the web application database.

Action:

The crisis manager assigns one or more tasks to a previously defined crisis event.

Expected Result:

The event/task configurations are saved to the web application database

Action:

The crisis manager activates one or more tasks that have been assigned to a crisis event.

Expected Result:

Activated tasks are sent to suitable volunteers. Each volunteers Crowdtasker-Application shows a notification that informs the user that a crisis event has received new tasks.

Action:

Volunteers interact with the CrowdTasker Application to complete activated tasks.

Expected Result:

Task results are sent to the crisis manager for evaluation

Action:

The crisis manager opens the Analytics view in CTA to display the feedbacks in a dynamic map and inspects aggregated feedback responses.

Expected Result:

CTA presents all submitted feedbacks in a dynamic map and visualizes available data: * Aggregated data per task for step types: Multiple Choice, Single Choice, Numbers * Received information for pictures and free text response types

---

Source URL (modified on 04/11/2018 - 14:49): https://driver-pos.atosresearch.eu/content/crowdtasker-us-gather-information-field-distributing-tasks

Published on *Driver+ PoS* (https://driver-pos.atosresearch.eu)

Crowdtasker TC: Gather information from the field by distributing tasks

Solution Reference: CrowdTasker: Crowdtasking solution for managing of the pre-registered volunteers

US reference: Crowdtasker US: Gather information from the field by distributing tasks

TC Summary:

This TC verifies, that crisis managers receive information from citizens in field that previously downloaded the app and have now performed one or more tasks defined by the crisis manager.

Test description:

Step	Action	Expected results	Required?	Results	Comment
1	Define Crisis Event	A crisis event (consisting of title, description and area) is visible in the CrowdTasker backend and stored in the applications database	x	-	Skip if event is already defined
2	Define Task(s) for Crisis Event	The defined tasks (consisting of title, description and a series of task steps) are visible in the CrowdTasker backend and stored in the applications database.	x	-	Skip if predefined tasks are used
3	Assign Task(s) to Crisis Event	The selected tasks are assigned to the selected crisis event. This is visible in the application and stored in the database.	x	-	Predefined tasks can be used
4	Activate Task(s)	Suitable volunteers, that previously installed the CrowdTasker APP, receive the tasks on their mobile phones. The CrowdTasker backend shows that the tasks have been sent out successfully	x	-	
5	Complete Task(s)	Task feedback from volunteers in the field has been sent back to the CrowdTasker backend and were stored in the applications database.	x	-	
6	Evaluate Task results	In CTA analytics view, the created Crisis Event and Task is selected and the returned geo-referenced feedback results are visible in the map. By clicking on each feedback the detailed response information (photo, single/multiple choice selection, number or free text) is shown.	x	-	

Annex 4 – Guidelines regarding the technical set-up view

This Annex gives additional explanations and practical guidelines regarding the technical-set.

The Trial scenario (which is defined outside of this solution testing procedure) defines which organizations will be involved as participants in the Trial. The organisational view shows, which solution will be used by which participating organisation.

Organisational view

Figure A1 gives the example of the organizational architecture of Trial 1. In this figure, the solutions are NowForce, Drones Rapid Mapping, 3Di, and Socrates OC. In this case, the specific organisations are not mentioned, but each solution is allocated to some command levels (Regional Operational Centre, Local Operational Centre, and Field).

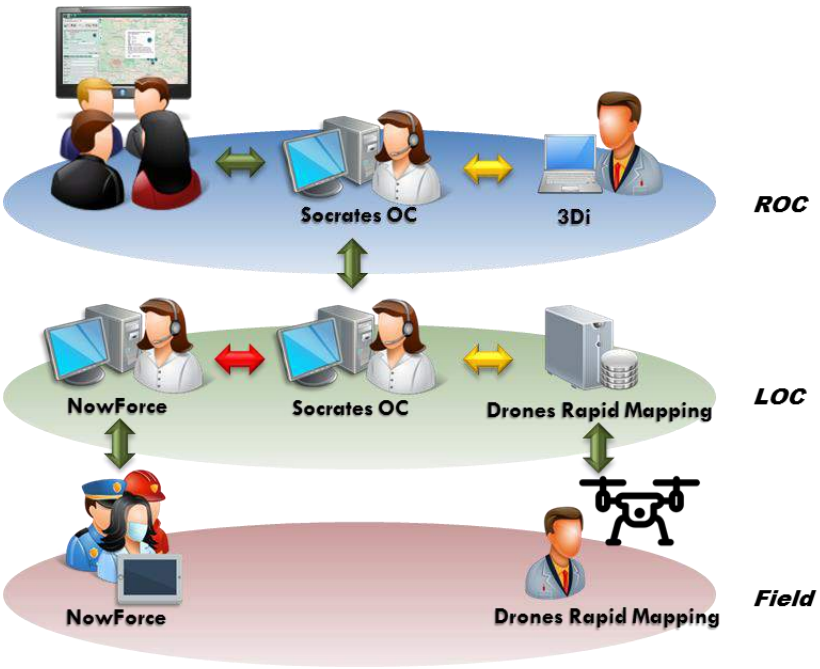

Figure A1: Trial 1 organisational view

Methodologies

The design and representation of information workflows can be supported by various system engineering methodologies. We recommend solution coordinators and Test-bed infrastructure coordinators to choose the methodology they are most familiar with.

The following methodologies are well known, and well adapted for this:

- **Unified Modelling Language** (UML) "is a general-purpose, developmental, modelling language in the field of software engineering, that is intended to provide a standard way to visualize the design of a system." (18). The Use case diagrams and Sequence Diagrams can be of particular help. Extensive explanations and examples can be found at (19) and (20).
- **Business Process Model and Notation** "is a graphical representation for specifying business processes in a business process model." (21). The specifications and some examples can be found in (22). A tutorial of BPMN, can be found in (23).