



Driving Innovation in Crisis Management
for European Resilience



D933.21 DRIVER+ POS DATABASE AND GUIDANCE TOOL PROTOTYPES

SP93 - SOLUTIONS

JULY 2019 (M63)



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:	D933.21 DRIVER+ PoS database and guidance tool prototypes
Deliverable Nature:	Prototype (P)
Dissemination Level:	Public (PU)
Due Date:	July 2019 (M63)
Submission Date:	17/07/2019
Subproject (SP):	SP93 - Solutions
Work Package (WP):	WP933 - DRIVER+ online platforms
Deliverable Leader:	Denis Havlik, AIT
Reviewers:	Héctor Naranjo Setién, GMV Maurice Sammels, XVR Erik Vullings, TNO
File Name:	DRIVER+_D933.21 DRIVER+ PoS database and guidance tool prototypes.docx
Version of template used:	V2.2 – February 2019

DISCLAIMER

The opinion stated in this report reflects the opinion of the authors and not the opinion of the European Commission. All intellectual property rights are owned by the DRIVER+ consortium members and are protected by the applicable laws. Except where otherwise specified, all document contents are: “©DRIVER+ Project - All rights reserved”. Reproduction is not authorised without prior written agreement.

The commercial use of any information contained in this document may require a license from the owner of that information.

All DRIVER+ consortium members are also committed to publish accurate and up to date information and take the greatest care to do so. However, the DRIVER+ consortium members cannot accept liability for any inaccuracies or omissions nor do they accept liability for any direct, indirect, special, consequential or other losses or damages of any kind arising out of the use of this information.

Revision Table

Issue	Date	Comment	Author
V0.00	10/04/2019	Initial draft.	Dražen Ignjatović, AIT
V0.01	10/05/2019	Initial description of the two prototypes; design and implementation overview based on materials prepared for D932.11 (1)	Dražen Ignjatović, AIT
V0.02	11/05/2019	Requirement status implementation table finalised	Dražen Ignjatović, AIT
V0.03	15/05/2019	Contributions to section 1	Dražen Ignjatović, AIT
V0.04	20/05/2019	Contributions to section 3	Dražen Ignjatović, AIT
V0.05	25/06/2019	Cleaned up the document by moving the instructions into comments, to be deleted later. Updated the implementation status table in Table A2: Implementation status Drafted the executive summary	Denis Havlik, AIT
V0.06	25/06/2019	Contributions to section 5 and 6	Miquel Milá Prat, ATOS
V0.07	27/06/2019	Contributions to sections 2 and 4	Dražen Ignjatović, AIT
V0.08	05/07/2019	Restructuring to achieve 1:1 mapping to D933.11 (2); removing or merging all contents into five sections structure (introduction, three “features” sections and conclusions/way forwards)	Denis Havlik, AIT
V0.09	08/07/2019	Peer review	Maurice Sammels & Hector Naranjo
V0.10	11/07/2019	Finalizing all sections.	Denis Havlik, Dražen Ignjatović, AIT
V0.11	12/07/2019	Peer Review	Hector Naranjo, GMV
V0.12	13/07/2019	Final check and approval for submission	Maurice Sammels, XVR
V0.13	16/07/2019	Final check and approval for submission	Tim Stelkens-Kobsch, DLR, Quality Manager
V0.14	17/07/2019	Final check and approval for submission	Marijn Rijken, TNO, Project Director
V1.0	17/07/2019	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:
 - a. Develop a common guidance methodology and tool, supporting Trials and the gathering of lessons learnt.
 - b. Develop an infrastructure to create relevant environments, for enabling the trialling of new solutions and to explore and share Crisis Management capabilities.
 - c. Run Trials in order to assess the value of solutions addressing specific needs using guidance and infrastructure.
 - d. Ensure the sustainability of the pan-European Test-bed.
2. Develop a well-balanced comprehensive Portfolio of Crisis Management Solutions:
 - a. Facilitate the usage of the Portfolio of Solutions.
 - b. Ensure the sustainability of the Portfolio of Solutions.
3. Facilitate a shared understanding of Crisis Management across Europe:
 - a. Establish a common background.
 - b. Cooperate with external partners in joint Trials.
 - c. Disseminate project results.

In order to achieve these objectives, five Subprojects (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 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 (FD). **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 standardisation.

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. Most important will be to build and structure a dedicated Community of Practice in Crisis Management, thereby connecting and fostering the exchange of lessons learnt and best practices between Crisis Management practitioners as well as technological solution providers.

Executive summary

This document is the **D933.21 DRIVER+ PoS database and guidance tool prototypes** deliverable of the DRIVER+ project. It relates to the prototype development work that was performed in **WP933 DRIVER+ online platforms** up to the month 63 of the project (July 2019) and summarizes the implementation status of the DRIVER+ Portfolio of Solutions (PoS) and Trial Guidance Tool (TGT) prototypes. These prototypes have been implemented as a single web site, available for the public beta testing at <https://pos-dev.driver-project.eu/> since September 2018 and gradually improved following an AGILE development approach.

The main audience of this document are the developers that are interested in learning how the PoS and TGT were developed in DRIVER+, e.g. to re-use and extend the platform. However, the effort has been made to make most of the deliverable comprehensible for non-technical audience.

The level of implementation of the online platforms is presented in comparison to specifications from the **D933.11 DRIVER+ online tools - implementation specifications** (2) document. Therefore, the current document closely follows the structure and naming conventions of the **D933.11** (2), with sections 2, 3 and 4 summarizing the implementation status of the functions that are common to both platforms, PoS specific, and TGT specific, respectively. The sub-sections thereof corresponds to key functionalities of the platforms¹.

Overall, the level of achievement is satisfactory and in line with the stakeholder expectations.

However, this does not mean that all the initial requirements and all the planned features have been implemented exactly in the way as initially requested or intended. In some cases, the stakeholders decided that the initial requirements do not need to be implemented, in other the already implemented features were found not fully satisfactory and the implementation had to deviate from the specifications to accommodate the user feedback. More detailed information on the level of implementation, issues encountered and deviations per feature and per requirement is provided in tabular form in sections 2, 3, and 4 and in the annex 2 of this document.

Table 2.1, Table 3.1 and Table 4.1 compare the implementation status to the specifications for common, PoS-specific and TGT-specific functionalities respectively. Table A2 in the annex of this document corresponds to annex 2 of **D933.11** (2) and compares the level of implementation with the requirements originating from relevant **WP922** and **WP932** deliverables as well as those that were received from the stakeholders independently prior to the publication of **D933.11** (2) in April 2019. This table is complementary to Table 2.1, Table 3.1 Table 4.1 and show the implementation status of individual requirements for each of the functionalities that the PoS and the TGT prototypes have.

With publication of this deliverable, the development of features of the DRIVER+ online platforms is officially finalized. In the remaining project duration, no new features will be implemented. However, the current platform functionality will be further improved in terms of usability and look & feel, and bugs will be fixed until the project end, as a part of the **T933.4** “industrialization” work. In addition, the platform(s) will be packaged for easy installation and initialization at new sites, as a part of the effort aiming to make the results more sustainable.

¹ E.g., the section 2.6 Validation of this document explains how the validation functionality has been implemented and corresponds to the section 2.6 Validation of D933.11, where this functionality has been specified.

TABLE OF CONTENT

1.	Introduction.....	11
1.1	Scope of this document and its relationship to other DRIVER+ work.....	11
1.2	Aim of the PoS and TGT	11
1.3	Development approach.....	12
2.	Functionalities shared among the PoS and the TGT	14
2.1	Overarching Site Design (GUI, structure, multilingual features)	15
2.2	Registration, user management, authentication and authorization	16
2.3	Collaborative work environment	17
2.4	Supervision and quality control	17
2.5	Search/matching and filtering	17
2.6	Validation.....	18
2.7	Help functionality.....	18
2.8	E-mail notification	19
2.9	Content exports	20
2.10	The EU law cookie compliance.....	20
2.11	Taxonomies and Terminology/Glossary	21
2.12	Country profiles.....	22
3.	Portfolio of Solutions	23
3.1	Solution group	24
3.2	Solution team management.....	27
3.3	Solution use case.....	27
3.4	Solution “used/tested in” reference	29
3.5	Solution documentation.....	31
3.6	Solution feedback	32
3.7	Solution search and matching functionality	33
3.8	Solution exports	35
3.9	Related solutions.....	36
3.10	Group-level solution validation.....	36
3.11	Solution landing pages.....	37
3.12	CM gaps	38
4.	Trial Guidance Tool	41
4.1	Trial group	42
4.2	Trial team management	46
4.3	Trial gap.....	46
4.4	Trial objective	47
4.5	Trial research question.....	48
4.6	Trial data collection plan	49

4.7	Trial evaluation approaches and metrics	50
4.8	Trial scenario.....	51
4.9	Solution selection.....	52
4.10	Test case	54
4.11	Execution phase	56
4.12	Evaluation phase	57
4.13	Trial search and matching.....	57
4.14	Trial exports	57
4.15	Knowledge Database	57
4.16	Group-level Trial validation.....	59
4.17	Trial landing pages.....	60
5.	Conclusions and the way forward	61
	References.....	63
	Annexes.....	64
	Annex 1 – DRIVER+ Terminology	64
	Annex 2 – Requirements implementation status	66
	Annex 3 – Solution PDF export example	74

List of Figures

Figure 1.1: Prototype relations.....	13
Figure 2.1: Multi-language menu	15
Figure 2.2: Modal window	16
Figure 2.3: Log in form	16
Figure 2.4: TRL taxonomy term tagged by different solutions	18
Figure 2.5: Custom help desk form.....	19
Figure 2.6: Interactive tutorial example	19
Figure 2.7: Custom e-mail module	20
Figure 2.8: Cookie usage information.....	21
Figure 2.9: Glossary term example.....	21
Figure 2.10: Glossary – alternative definitions.....	22
Figure 2.11: Country profile page example.....	22
Figure 3.1: Form display.....	26
Figure 3.2: Display	27
Figure 3.3: Example structure of CM functions taxonomy	28
Figure 3.4: Use case form display.....	29
Figure 3.5: Use case display - default	29
Figure 3.6: Used/tested in form display.....	30
Figure 3.7: Example of Entity browser implementation	30
Figure 3.8: Used/tested in display – default	31
Figure 3.9: Documentation template form display	31
Figure 3.10: Documentation template display - default.....	32
Figure 3.11: Feedback form display.....	32
Figure 3.12: Feedback group form display.....	33
Figure 3.13: Feedback display	33
Figure 3.14: Solutions overview page.....	34
Figure 3.15: Search index configuration	34
Figure 3.16: Similar solutions	35
Figure 3.17: Visualised Restful request.....	35
Figure 3.18: Related solutions.....	36
Figure 3.19: Positive validation result.....	36
Figure 3.20: Negative validation result.....	37
Figure 3.21: “Use cases” view	37
Figure 3.22: Result of the “Use cases” view.....	38
Figure 3.23: CM gap form display.....	39
Figure 3.24: CM gap display	39
Figure 3.25: CM gap.....	40
Figure 4.1: “Add” form mode	44

Figure 4.2: Default form mode	44
Figure 4.3: Display	45
Figure 4.4: Drupal entity	47
Figure 4.5: Individual form and display for group and non-group entities	47
Figure 4.6: Trial objective form mode	48
Figure 4.7: Trial objective display	48
Figure 4.8: Trial research question form mode	49
Figure 4.9: Trial research question display	49
Figure 4.10: Trial data collection form mode	50
Figure 4.11: Trial data collection display	50
Figure 4.12: Trial evaluation approaches and metrics form mode	51
Figure 4.13: Trial evaluation approaches and metrics display	51
Figure 4.14: Trial scenario form mode	52
Figure 4.15: Trial scenario display	52
Figure 4.16: Trial solution form display	53
Figure 4.17: Example of suggested solutions	54
Figure 4.18: Trial solution display	54
Figure 4.19: Test case form display	55
Figure 4.20: Test case display	56
Figure 4.21: Checklist example	56
Figure 4.22: Checklist edit form example	57
Figure 4.23: SLR content page	58
Figure 4.24: Lessons learned form mode	58
Figure 4.25: Lessons learned display	58
Figure 4.26: Positive validation result	59
Figure 4.27: Negative validation result	59
Figure 4.28: Example of a view used in TGT	60
Figure 5.1: Distribution wizard installer	61

List of Tables

Table 2.1: Shared functionality: implementation vs. 932.11 specifications	14
Table 3.1: PoS functionalities: implementation vs. 933.11 specifications	23
Table 3.2: Implementation of additional functionalities	24
Table 4.1: TGT functionalities: implementation vs. 933.11 specifications	41
Table A1: DRIVER+ Terminology	64
Table A2: Implementation status	66

List of Acronyms

Acronym	Definition
API	Application programming interface
CM	Crisis Management, see also Table A1
CSS	Cascading Style Sheets
DoW	Description of Work
EU	European Union
EVA	Entity Views Attachment
GUI	Graphical User Interface
HTML	Hypertext Markup Language
JSON	JavaScript Object Notation
MS	Microsoft
PDF	Portable Document Format
PoS	Portfolio of Solutions, an online database to maintain a list of CM solutions
QA	Quality assurance
REST	Representational State Transfer
SLR	Systematic literature research
TGT	Trial Guidance Tool, (online workflow tool for developing the Trials according to TGM)
TGM	Trial Guidance Methodology, as defined in deliverable D922.41_Trial guidance methodology and guidance tool specifications (version 2)
WP	Work Package

1. Introduction

1.1 Scope of this document and its relationship to other DRIVER+ work

This document is “a report accompanying the online platform prototypes” part of the **D933.21 DRIVER+ PoS database and guidance tool prototypes** deliverable of the DRIVER+ project. It relates to the prototype development work that was performed in **WP933 DRIVER+ online platforms** up to the month 63 of the project (July 2019) and summarizes the implementation status of the DRIVER+ Portfolio of Solutions (PoS) and Trial Guidance Tool (TGT) prototypes.

The main audience of this document are the developers that are interested in learning how the PoS and TGT were developed in DRIVER+, e.g. to re-use and extend the platform. However, the effort has been made to make most of the deliverable comprehensible for non-technical audience.

Sections 2, 3, and 4 of this document explain the extent of the PoS and GT implementation and relate the actually implemented features to specifications from the **D933.11 DRIVER+ online tools - Implementation specifications (2)** document. Section 5 summarises the result of implementation work and gives an overview of future actions. In Annex 1, a table of DRIVER+ terms is shown to ensure the use of a common language in all project deliverables and finally, Annex 3 of this document shows an example of a PDF export of a solution on the PoS, which is one of many functionalities that the described prototypes provide.

In turn, the implementation specifications reflect the requirements from the following sources:

- **DRIVER+ Description of Work (3)**, especially **WP932** and **WP933** descriptions.
- **D932.11 Functional design of the PoS database (1)**.
- **D922.21 Trial Guidance Methodology and Guidance Tool Specifications (4)**.
- **D932.12 PoS Tutorials and recommendations (5)**.
- **D922.41 Trial Guidance Methodology and Guidance Tool Specifications v2 (6)**.
- Additional requests from various sources, including PoS/TGT test and training sessions, teleconferences with main TGT and PoS stakeholders and individual user feedbacks that were received after the finalization of these deliverables and maintained on the <http://driver-pos-ticket.atosresearch.eu/wishes> platform.

For the sake of transparency, the relation of the implemented functionality to these requirements is summarized in the Annex 2 of this document. This table is complementary to Table 2.1 , Table 3.1 Table 4.1 and show the implementation status of individual requirements for each of the functionalities that the PoS and the TGT prototypes have.

1.2 Aim of the PoS and TGT

The Portfolio of Solutions (PoS) and Trial Guidance Tool (TGT) aim to support CM practitioners by providing two core functionalities:

1. Assistance in discovery of innovative solutions in the field of crisis and disaster management that are accommodated inside of the PoS’s database.
2. Assistance in trialling and assessing of solution’s performance by providing support in following methodological steps, defined by Trial Guidance Methodology (TGM) in the **WP922 Guidance methodology**, accommodated inside of the TGT.

In addition to these two main functions, the PoS and TGT aims to support solution providers and CM practitioners in the following ways:

- By allowing the solution providers to advertise their innovative solutions in a practitioners-friendly way.
- By allowing the solution providers and practitioners to publish their experiences with solutions in DRIVER+ Trials, in other research projects as well as in the real-world use.
- By supporting the practitioners in formulating the CM gaps independently of the trials, automatically listing the potential solutions to these gaps and making the already defined gaps available for use in Trial planning.
- By making the information on policy, legislation, organisation, procedures & capabilities (PLOPC) in crisis management and disaster response that was previously gathered in DRIVER+ for 28 EU states, Albania, Iceland, Israel, Montenegro and Turkey publicly available and linked with Trials and other examples of solution use.
- By providing a comprehensible glossary of CM terms, with their definitions used in DRIVER+ as well as with the alternative definitions that are commonly used by specific CM sectors and communities and automatically linking these definitions to glossary terms whenever these are used on the site.

1.3 Development approach

Both PoS and TGT prototypes were developed simultaneously using a mixture of two software development approaches:

1. Agile approach.
2. Prototyping approach.

The first approach was chosen because it resonated well with the needs of the project, with the emphasis on the team development rather than following predefined structured development process. It encouraged cooperation and team work between partners, while maintaining contact with end users allowing quick responses to changing specifications.

Two environments were set, one being a development environment and the other a production environment. New features were developed in monthly sprints in the development environment, and subsequently migrated to the production environment, where they could be immediately used by relevant stakeholders.

A simple ticketing system was used to support the agile development process, where all new specifications as well as problems that occurred were documented and tracked.

The second approach was chosen to improve end user participation, where smaller scale models of desired functionalities were introduced, to ensure that they meet the user's needs. New features were presented during different workshops and online meetings where the feedback was collected and evaluated which lead to further improvement of the both prototypes.

Important note about the development of the PoS and the TGT prototypes is that they were developed and stored using a common database, which allowed different interactions between individual components of both tools. Nevertheless, it is possible to use each tool separately, not providing a limited part of the functionalities due to inability to establish relations between some components that are a part of the other tool. Most notably, the automated linking of trials and solutions is only possible if the two tools are used together, as illustrated in Figure 1.1.

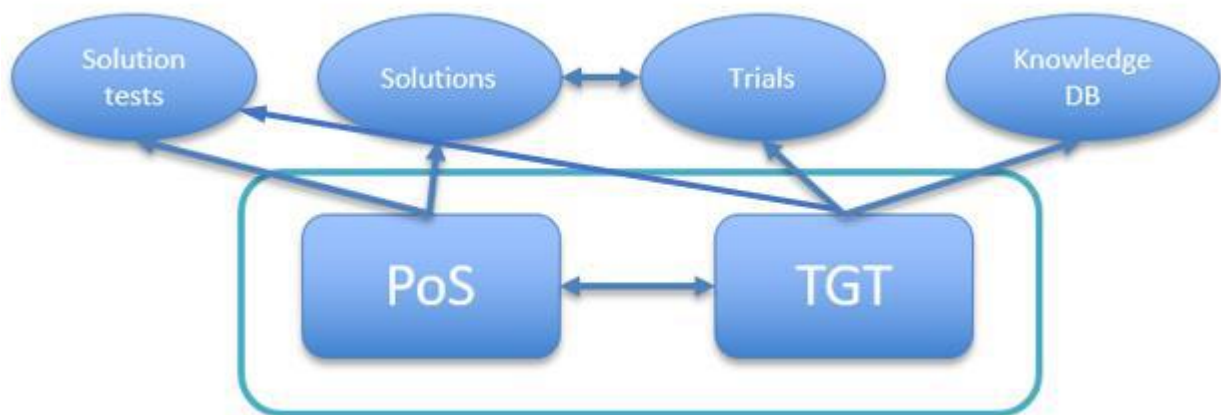


Figure 1.1: Prototype relations

Another important development constrain is the use of the Drupal 8 framework – an open-source, content management framework written mainly in PHP. The main reasons for choosing the Drupal as the basis for the development of the PoS and the TGT prototypes were already introduced in the section 1.3 of **D933.11 (2)**:

1. Drupal is one of the most used content management systems, with a large installed base and a long track record of usage in real-world applications.
2. Drupal is open source, which assures that practically any improvement and extension can be developed within the project if necessary.
3. Unlike the other widely used Open Source content management systems (WordPress, Joomla), Drupal is designed to be a web application framework and suitable for developing interactive applications for knowledge management and business collaboration, rather than a straightforward content management system.

Drupal platform offers a wide variety of “core” and “contributed” modules² that are highly configurable to the point where “configuration” becomes a form of higher-level programming. Large part of the PoS and TGT functionality was implemented in this way, with remaining part being achieved through development of the special purpose modules by the DRIVER+ development team.

² The term “contributed module” in this context is to be understood a Drupal platform extension written in PHP that was developed by the members of the Drupal community that are not a part of the PoS and TGT prototype development team.

2. Functionalities shared among the PoS and the TGT

Functionalities that are shared between the PoS and the TGT prototypes have been specified in the section 2 of **D933.11 DRIVER+ online tools - Implementation specifications** (2). The following table provides an overview of the implementation status for each of the functionalities, with functionality names corresponding to the sub-section names in section 2 of **D933.11** (2)³. Additional comment field summarizes challenges encountered, implementation issues and change with regards to **D933.11** specifications.

Description of implementation status values:

- *Implemented* – the functionality has been fully implemented and is available on the prototypes;
- *in course of implementation* – the functionality has been partially implemented and may not be available on the public version of the PoS and TGT prototype;
- *stalled* – the functionality has not been implemented for a specific reason which is stated in the comment field.

Table 2.1: Shared functionality: implementation vs. 932.11 specifications

Functionality	Implementation status	Comment
Overarching Site Design (GUI, structure, multilingual features).	Implemented	Technical implementation exists that allows translating of Trials and solutions. Translations will be performed by professional translators (pending the acceptance of the DoW amendment request)
Registration, user management, authentication and authorization.	Implemented	In addition to standard Drupal user management functions, a customized module was implemented to allow the usage of a LinkedIn (https://www.linkedin.com/) credentials for the login.
Collaborative work environment.	Implemented	Solutions and Trials are implemented as Drupal groups to allow collaborative work on the content and self-organization of the working groups.
Supervision and quality control.	Implemented	Contrary to D933.11 (2) specifications, users can decide when to publish their contents without requesting the quality assurance by editors. If necessary, the editors can unpublish the contents if necessary. The initially specified process was deemed unsustainable.
Search/matching and filtering.	Implemented	

³ E.g. “Collaborative work environment” functionality in the table corresponds with the section 2.3 Collaborative work environment in **D933.11**.

Functionality	Implementation status	Comment
Validation.	Implemented	
Help functionality.	Implemented	Static help texts are gradually replaced by interactive content based on https://h5p.org .
E-mail notification.	Implemented	
Content exports.	Implemented	There was a discussion what formats should be available, and it was decided that only the PDF export will be provided.
The EU law cookie compliance.	Implemented	
Taxonomies and Terminology/Glossary.	Implemented	
Country profiles.	Implemented	

More details for each of the implemented features is provided hereafter.

2.1 Overarching Site Design (GUI, structure, multilingual features)

Since the PoS is designed to be used by various stakeholders, ranging from solution providers and practitioners to all other interested parties that use the site, certain expectations needed to be met. One of them was a possibility to offer the content in different languages. The version of Drupal that was used already provides the functionality to create multilingual sites in its core, which was then adjusted to match the requirements in the project. The PoS site has been configured to enable the translations of some content like the Trials, the solutions, the solution use cases and the Trial gaps. At this initial approach, the defined translatable parts for Solutions and Trials were: main text fields (title, summary) and short taxonomies.

The objective to add multilingual functionality into the PoS site was to give it the possibility to publish its main data in different languages, not to provide a full multilingual site.

For this prototype, PoS site has been configured to use six chosen languages: English (as default site language), German, French, Dutch, Polish and Italian; but this number can easily be incremented (or also decremented) at any time, using the backend configuration of the site. If some other language is needed in the future it can be easily added.

When users access any of the translatable contents in the PoS site, they can see a menu as shown in Figure 2.1.

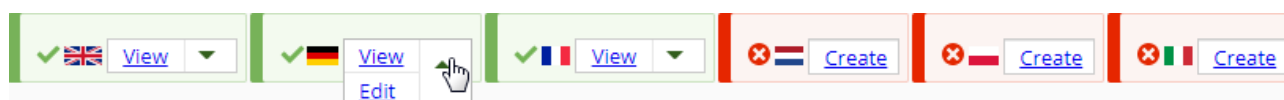


Figure 2.1: Multi-language menu

This menu enables users to view, edit or create new translations for the content, in case that they have sufficient permissions to do so, just by clicking on the appropriate option. Figure 2.2 illustrates the modal window that is opened when a user clicks the “create” new translation of an item.

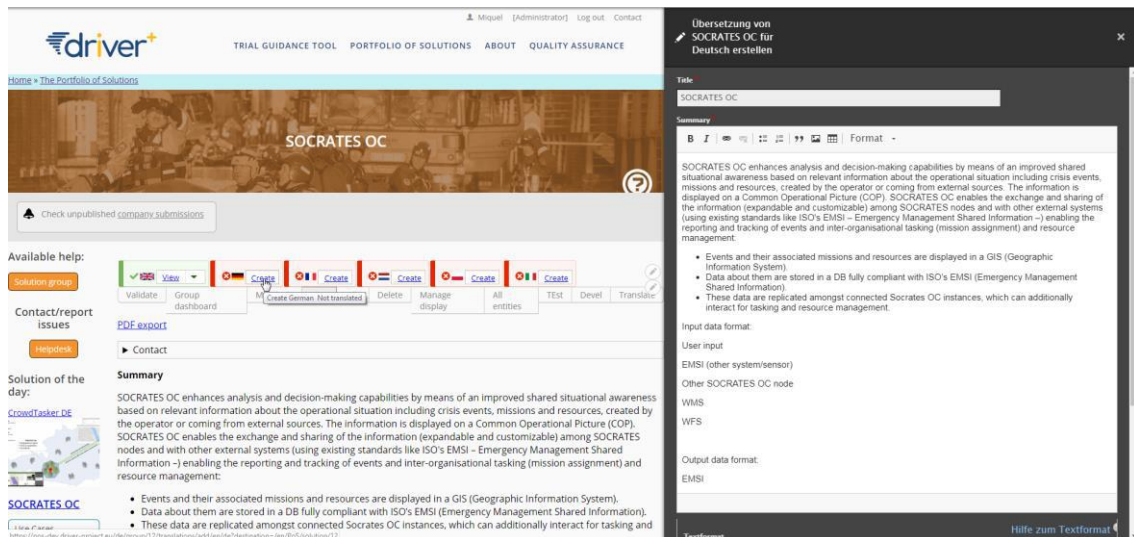


Figure 2.2: Modal window

The graphical user interface (GUI) elements are further elaborated in sections 3.11 and 4.17.

At a time of writing this document, the consortium has requested a DoW amendment that would allow us to engage a professional translation service. If this amendment is accepted, the site multilingual features will be further improved and extended and most (all?) of the site contents translated.

2.2 Registration, user management, authentication and authorization

Both prototypes implement functionality for registration and management of the users as well as their authentication and authorisation. This was implemented using several individual elements which allow following actions to be performed:

1. All anonymous users of the prototypes can create a new account.
2. An administrator can accept or reject an account request as described in section 2.4.
3. Enabled users can log in with their credentials, or with their LinkedIn account credentials.
4. Enabled users can reset their own password.

Figure 2.3 illustrates the login form of the PoS and the TGT prototypes with all options that are presented to the anonymous user.

Figure 2.3: Log in form

All users of the prototypes can be divided into following groups based on their permissions:

1. Anonymous users - all users that are not logged in and can only view published content.
2. Registered users – all users that have an enabled account and can add new and edit own content.
3. Quality assurance – all users that have an enabled account but have additional privileges that allow them to publish/unpublish and edit all content.
4. Administrator – a super user that has all permissions and can modify the prototypes.

2.3 Collaborative work environment

The PoS and the TGT prototypes are meant to be used not only by individual users, but also for working groups of users that are collaborating on a joint project, for example on describing a new innovative solution or conducting a Trial for solution assessment. In order to provide this functionality, the prototypes use the Drupal's built-in user management system together with a contributed "Group" module that allows micro-user management to be performed within the user group without involving a centralised administrator role. In addition, this module assures that the group of users can organise itself and assign individual roles with different permissions to work on the same content based on their role.

Following group types have been implanted within the prototypes:

1. Solution group: used by the PoS prototype to describe solution offers (section 3).
2. Trial group: used by the TGT prototype to define Trials and document progress and results (section 4).
3. Country profile group: used in both prototypes to publish summary how Crisis Management is organised in different EU states (section 2.12).

2.4 Supervision and quality control

Both prototypes implement a supervision and quality control of content that is divided into two levels:

1. The users can decide on their own if they wish to publish or unpublish their content.
2. The quality assurance users (described in chapter 2.2) can ban the published content and provide comments/requests for improvements to the authors if needed.

In addition, quality assurance users are also responsible for processing new user account requests, where the plausibility of the accounts is checked based on the following criteria:

1. User is found on the web (LinkedIn account, company e-mail address).
2. User is relevant for the crisis and disaster management domain (based on the previous condition).
3. Real name is provided.
4. Users profile is filed in.

The difference to the specification of this functionality as described in the **D933.11** (2) where the content was only published at request is that all content is set to be published per default with the possibility to unpublish it at any time. This was done for practical reasons, to improve the usability and to stimulate the users in providing content, as they are immediately rewarded with the result.

2.5 Search/matching and filtering

The PoS and the TGT prototypes implement several ways to assist the users in finding the information they are interested in:

1. Full text search is implemented with configurable relevance ranking and sorting criteria.
2. Faceted search is implemented to limit the full text search results by a specific taxonomy term.
3. Taxonomy terms show what entities are tagged with them (example shown in Figure 2.4).

4. More elaborated automatic matching mechanisms were implemented for “similar solutions” (section 3.7) and “solutions addressing the gap” (sections 3.12 and 4.13).

System prototype demonstration in operational environment.

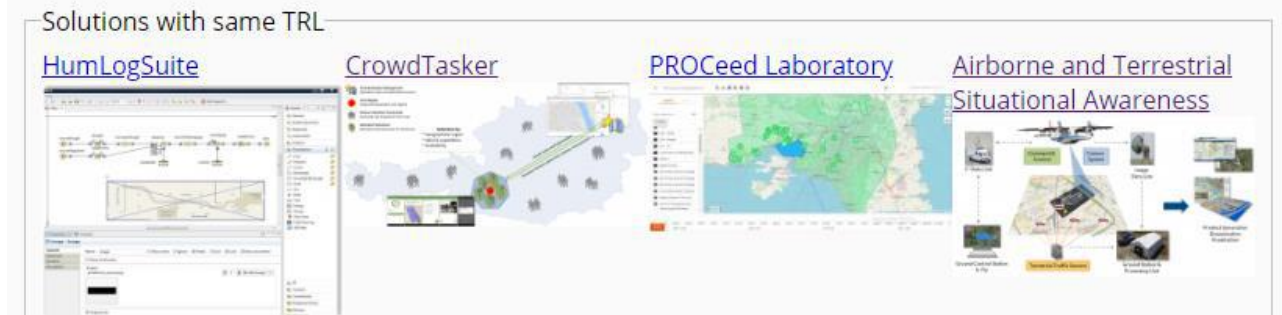


Figure 2.4: TRL taxonomy term tagged by different solutions

2.6 Validation

One of the basic ideas behind the PoS and the TGT prototypes was that they automatically assist the user in their activities. Together with other functionalities that aim the same goal, a validation functionality was implemented that checks the user input and provides feedback to the user. The feedback can be positive and negative, and the negative feedback can further be divided into:

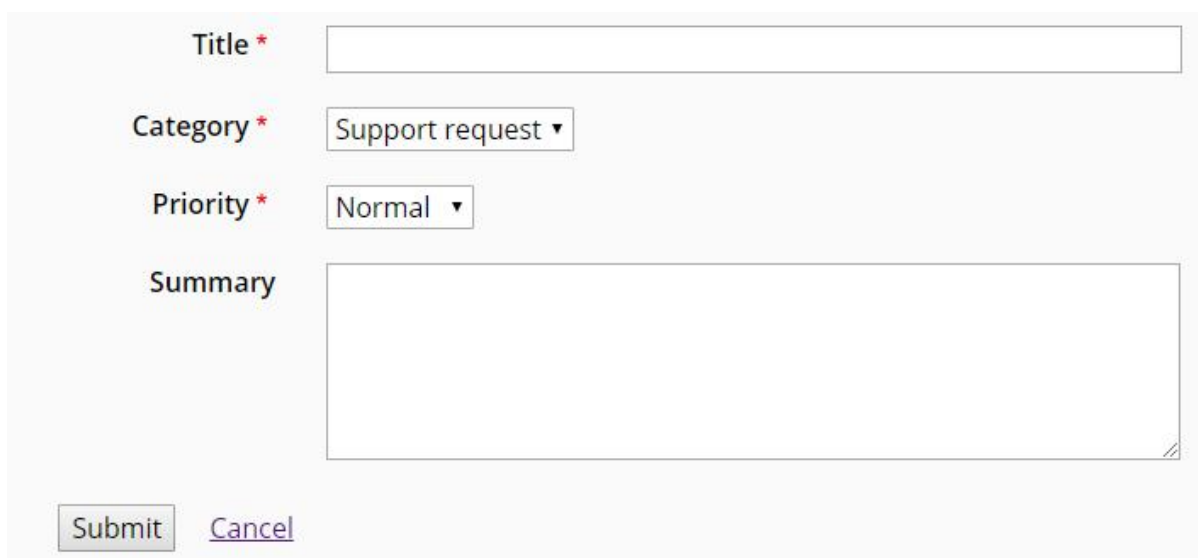
1. Critical error – an error that indicates that the user didn't provide the correct input.
2. Warning – an error that indicates that the user should perform additional actions but can decide not to do so.

This functionality together with implementation examples are further elaborated in sections 3.10 and 4.16.

2.7 Help functionality

The PoS and the TGT prototypes offer several help functionalities:

1. Contextual help, custom developed functionality that displays help texts to the user based on his location on the site based on the unique URL.
2. Tutorials, interactive content developed with the help of “h5p” module (<https://h5p.org/>) that provides additional help to the user while navigating the site's structure. This module allows utilisation of HTML5, which is the latest evolution of the standard that defines HTML, providing new elements, attributes and behaviours (Figure 2.6).
3. Help desk, a custom developed module that allows all users to contact site administrators for help. In addition, this module is configured to hide the user's real e-mail address to preserve his privacy and to send the URL from where the form was called to provide additional information about the problem that the user faced (Figure 2.5).



The form is titled 'Custom help desk form' and contains the following fields:

- Title ***: A text input field.
- Category ***: A dropdown menu with 'Support request' selected.
- Priority ***: A dropdown menu with 'Normal' selected.
- Summary**: A large text area for a detailed description.
- Submit**: A button to submit the form.
- Cancel**: A link to cancel the form.

Figure 2.5: Custom help desk form

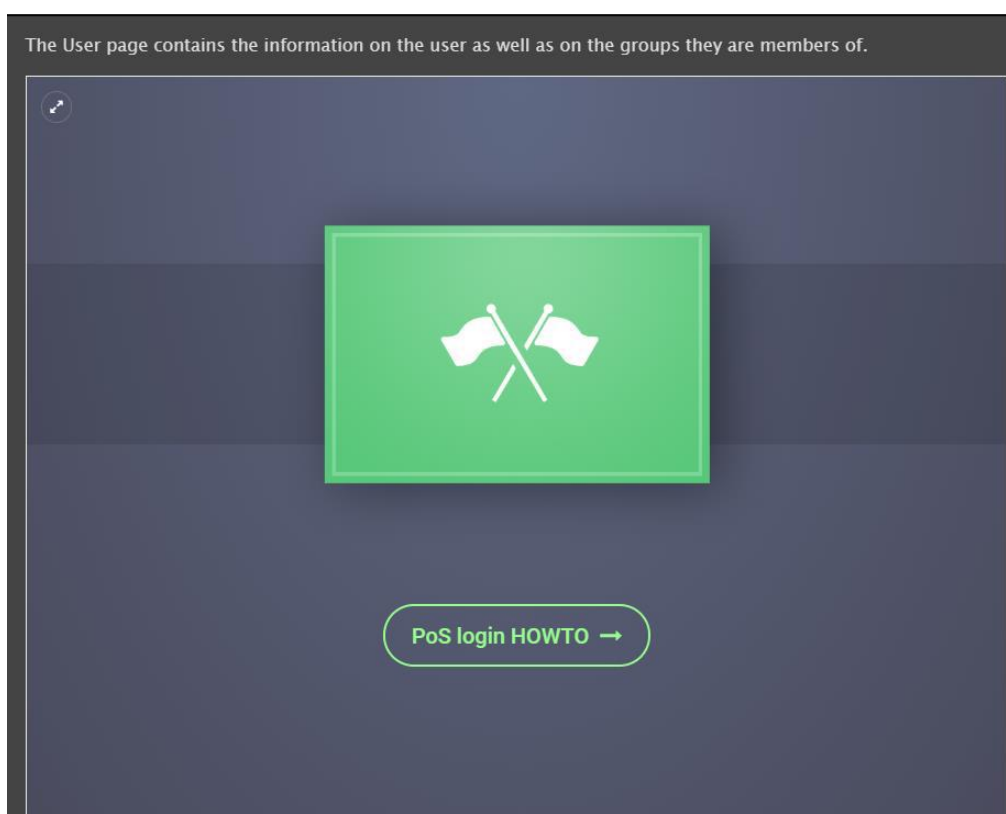


Figure 2.6: Interactive tutorial example

2.8 E-mail notification

E-mail notification functionality for the PoS and the TGT prototypes has been implemented through a custom Drupal module. With this module, administrator can send e-mails to a group of users at once. It can be configured to send weekly and monthly digests automatically, to remind the users of validation errors in their content. In addition, all changes in their contents that happened since the user was last logged in are also sent. Administrative configuration user interface of this module is shown in Figure 2.7, showing different user groups to be targeted.

This module also adds a Boolean field “accept_email_notifications” to user profile that allows users to opt out of the e-mail notifications.

Please select the recipients of the e-mail *

☐ All the users of the site

☐ Users of the selected roles

☐ Members of the selected groups

☐ Only to me

Subject e-mail *

Subject of the e-mail.

Body e-mail *

Write here the body of the e-mail. If you use [user:display-name] and [site:name] they will be replaced by the appropriate values.

Figure 2.7: Custom e-mail module

2.9 Content exports

In order to provide the possibility to export information in a readable format such as PDF, the PoS and TGT prototypes use a contributed “Entity Print” module which was configured and fine-tuned to meet the stakeholder expectations. By utilising this module, a designated “PDF” display mode was implemented and configured. Another contributed “Entity Views Attachments” (EVA) module is used to further enhance the export. It allows “attaching” of entities, therefore making it possible to display any type of derived data at the entity pages and PDF exports.

The biggest advantage of this approach is the high customisability of the output, where almost any desired output can easily be achieved by combining the functionalities of the two modules. Example of such PDF export is provided in Annex 3 of this document.

2.10 The EU law cookie compliance

Since the PoS and the TGT prototypes are implemented as a web application, certain laws are applicable when using them. An example of such case is the usage of cookies in web applications which is defined by the EU law. To comply with this requirement a contributed module is used. This module has been configured so that all site visitors are informed that some data about their usage will be collected, and the “More info” link is provided which redirects the user to the terms and conditions page, where it is defined under what circumstances this data is used. Figure 2.8 illustrates the cookie compliance window, as shown to the logged in user.

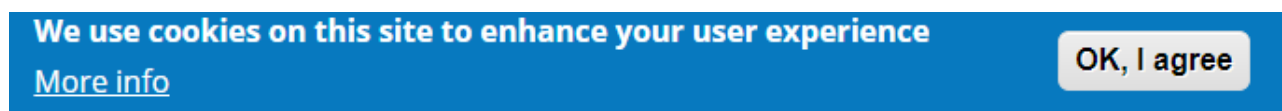


Figure 2.8: Cookie usage information

2.11 Taxonomies and Terminology/Glossary

The PoS and the TGT prototypes aim to improve common understanding of the crisis and disaster management domain across Europe and therefore also implement a glossary, an alphabetical list of words from this domain with their definitions. This list was enriched with visual elements that aim to automatically link words from texts that are stored in prototypes with their definitions, if any are available. This functionality is supported by a contributed module called “Onomasticon”, which requires a list of words with their definitions to be provided as a Drupal taxonomy and automatically inserts links to corresponding term definitions wherever necessary. Words with glossary definitions appear underlined in the text and a definition is shown when a mouse cursor hovers over the word, as illustrated in Figure 2.9.

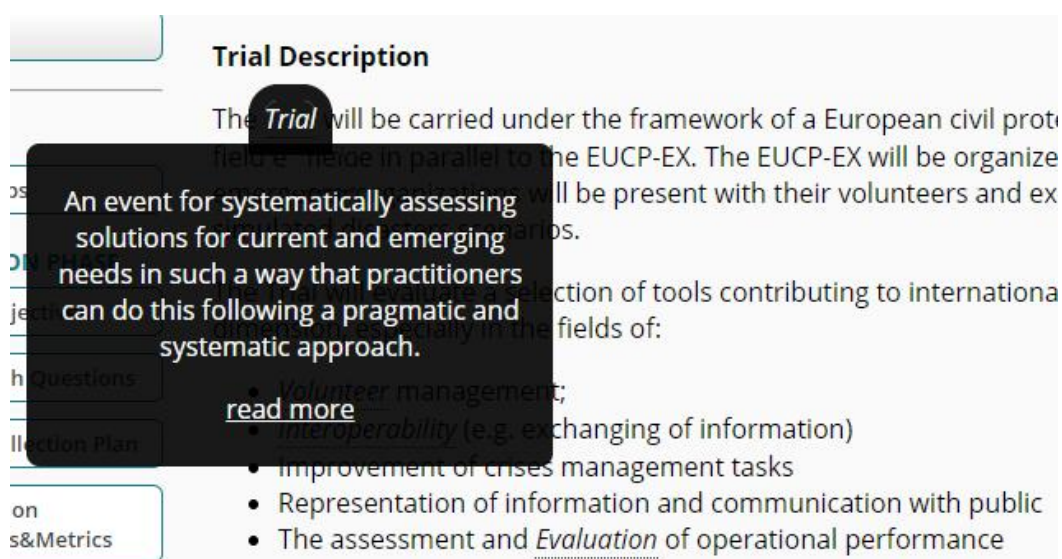


Figure 2.9: Glossary term example

An extension of the onomasticon module has been implemented to allow showing additional information through a “read more” link. This allows us to present the users with an overview of alternative definitions for the glossary terms, as shown in the Figure 2.10.

✎ Assessment
<p>▼ Definition</p> <p>The process of estimating the quality of an explored object, i.e. crisis management solution, based on the outcome of experimentation activities and other sources (e.g. operational experience). In an experimentation context the assessment process interprets the results of the experiments in a wide operational context, focusing on potential effects and impacts. Compared to evaluation, which results in knowledge about the outcome of a particular experiment, assessment synthesises the evaluation results in a wider context with the purpose of drawing more general conclusions, considering a broad set of aspects, typically aiming at informing decisions.</p> <p>Source: ISO/IEC 23988:2007(en) Information technology — A code of practice for the use of information technology (IT) in the delivery of assessments.</p>
<p>▼ Definition</p> <p><i>Assessment (Conformity):</i> Conformity assessment is a demonstration that specified requirements relating to a product, process, system, person or body are fulfilled</p> <p>Source: another source</p>
<p>▼ Definition</p> <p><i>Assessment (Dynamic Risk):</i> Continuing assessment appraisal, made during an incident or emergency, of the hazards involved in, and the impact of, the response</p> <p>Source: another source</p>

Figure 2.10: Glossary – alternative definitions


First definition is the one used in DRIVER+, whereas other definitions warn the PoS/TG users that some stakeholder groups may be using the term differently.

2.12 Country profiles

The PoS and the TGT prototypes implement a collection of summaries how Crisis Management is organised in different EU states. Individual country profiles are uploaded as documents and presented on an overview page, with a possibility to view them online, or to download them. Additionally, individual landing pages were implemented, where a short summary with indication if a Trial was held in that country is presented. Figure 2.11 illustrates an example of how this information is presented to the user.


Trials held in this country

[DRIVER+ Trial 3 - AUSTRIA](#)



In 2003, the Ministry of the Interior became the main responsible federal organisation for the coordination of disaster protection management, crisis management and international disaster relief. In this context, the Federal *Crisis and Disaster Protection Management* was established and became a major pillar of civil defence in Austria. It defines the measures and responsibilities in crisis and disaster case on the basis of two fundamental principles: the principle of subsidiarity and the principle of solidarity. The first principle is a political maxim specifying that intervention measures have to be implemented in the sense of self-help acting on the lowest possible level, e.g. the local level. This implies a bottom-up principle ensuring that measures necessary to manage crisis and disasters, are as long as possible performed by local organizations. The second principle ensures that in a case of an event, which exceeds the capacities at the local level, the community mechanism to overcome the crisis and disasters will be activated, ensuring that challenges are tackled with the help from the next higher organisational level. While the departments at the federal state are mainly responsible for prevention and financial recovery measures, the authorities of the provinces operate as the core institutions in

ZIVILSCHUTZ



ÖSTERREICH

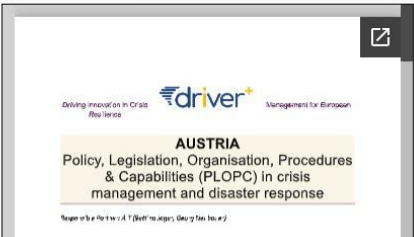


Figure 2.11: Country profile page example

3. Portfolio of Solutions

The PoS is a web-based database driven catalogue whose main purpose is to document new innovative solutions in the field of crisis and disaster management. It must support solution providers in systematic description of solutions as well as practitioners and other stakeholders in the process of discovering such solutions in a most efficient way. In order to fulfil these two main objectives, several functionalities had to be implemented.

The main idea of the PoS prototype is to have one main entity called “Solution” which can be extended with several other sub-entities, namely “Use case”, “Used/tested in”, “Solution documentation”, “Feedback”.

The main functionalities that the PoS prototype offers are:

1. Overview page backed up by a search engine and filters to allow easy discovery of solutions.
2. Predefined template backed up by a dedicated taxonomy to describe a solution.
3. Solution team management capabilities.
4. Predefined template to describe use cases with relation to CM functions taxonomy.
5. Predefined template to describe references to solutions.
6. Predefined template to store solution related documentation.
7. Feedback section for all site users to leave comments and rate the solutions.
8. Solution search and matching.
9. PDF export to allow easy extraction of information.
10. Restful web interface to communicate with other platforms.
11. Solution landing pages to display all information about a solution.
12. Validation function that automatically checks if some information is missing.

Table 3.1 provides an overview of the implementation status for each of the functionalities, with functionality names corresponding to the sub-section names in section 3 of **D933.11** (2), with exception of the sections 3.11 and 3.12 which were added in this document. Additional comment field summarizes challenges encountered, implementation issues and change with regards to **D933.11** (2) specifications.

Description of implementation status values:

- *Implemented* – the functionality has been fully implemented and is available on the prototypes;
- *in course of implementation* – the functionality has been partially implemented and may not be available on the public version of the PoS and TGT prototype;
- *stalled* – the functionality has not been implemented for a specific reason which is stated in the comment field.

Table 3.1: PoS functionalities: implementation vs. 933.11 specifications

Functionality	Implementation status
Solution group.	Implemented
Solution team management.	Implemented
Solution use case.	Implemented
Solution “used/tested in” reference.	Implemented
Solution documentation.	Implemented
Solution feedback.	Implemented
Solution search and matching functionality.	Implemented

Functionality	Implementation status
Solution exports.	Implemented
Related solutions.	Implemented
Group-level solution validation.	Implemented

Table 3.2 provides an overview of additional functionalities that were not covered in **D933.11** (2) but were implemented in the PoS prototype.

Table 3.2: Implementation of additional functionalities

Functionality	Implementation status	Comment
CM gaps	Implemented	Though no specification for having CM gaps outside of a Trial were defined in D933.11 (2), out of practical reasons this functionality was added to the PoS prototype.

3.1 Solution group

Solution group template is divided into three tabs:

1. Solution summary.
2. Meta information.
3. Administrative.

The fields used for this template are limited to the following types:

1. Entity reference, field used to add a reference to other content on the website.
2. Text, textual field used for storing different textual information types, such are long, short, formatted etc.
3. Boolean, field used to store information that has one of two possible values.
4. Image, field used to store uploaded image file with related information.
5. Video, field used to store information about embedded video file, hosted outside of the host website.

All fields have individual settings, but they all share common ones, such as:

1. Field label, a help text to be shown to the user under the field.
2. Required status, information if the field is mandatory or not.
3. Number of values for a field - ranging from 1 to unlimited number of values, etc.

In the solution description template several entity reference fields are used and they can be divided into two groups:

1. Fields that reference taxonomy vocabularies:
 - a. "Crisis cycle phase" containing following values:
 - Preparedness.
 - Risk assessment.
 - Response.
 - Recover.
 - Mitigation.
 - b. "Crisis size" containing following values:
 - Cross-border.
 - Large scale.
 - Regional.
 - Local.

- c. “Innovation stage” containing following values:
 - Stage 1: Concept.
 - Stage 2: Research and development.
 - Stage 3: Initial Piloting.
 - Stage 4: Early adoption/distribution.
 - Stage 5: Market growth.
 - Stage 6: Wide-scale adoption.
- d. “Readiness” containing following values:
 - TRL 2 - Technology concept formulated.
 - TRL 3 - Experimental proof of concept.
 - TRL 4 - Technology validated in lab.
 - TRL 5 - Technology validated in relevant environment.
 - TRL 6 - Technology demonstrated in relevant environment.
 - TRL 7 - System prototype demonstration in operational environment.
 - TRL 8 - System complete and qualified.
 - TRL 9 - Actual system proven in operational environment.
2. Fields that reference other content.
 - a. “Provider”, field referencing “Organisation” which is a content type implemented to store information about organisations.
 - b. “Supported standards”, which is a content type implemented to store information about standards that are relevant in the field of crisis and disaster management.

Two types of text fields are used in the solution description template:

1. Formatted long:
 - a. “Summary”, used to store formatted text description of a solution.
2. Plain long:
 - a. “QA comments”, used to store plain text information regarding QA process.

Four Boolean fields as well are used in the solution description template:

1. “IPR confirmation”, field to indicate acknowledgment of IPR.
2. “QA approved?”, field to indicate if the solution description meets the QA requirements.
3. “Request publication?”, field to indicate that the solution provider wishes to publish the solution description.
4. “Terms and conditions confirmation”, field to indicate agreement with site’s terms and conditions of usage.

Finally, two fields that do not fit any of these groups are also used in the solution description template:

1. “Illustrations”, field used to store solution illustrations.
2. “Video illustrations”, field used to store hyperlinks to solution video illustrations.

Figure 3.1 shows the form mode of the solution group template. In Drupal, form mode is used to define how information is going to be displayed to the user in create/edit forms.

FIELD	WIDGET	
+ Solution summary	Details ▼	Default state closed Mark as required
+ Title	Textfield ▼	Textfield size: 60
+ Provider	Autocomplete ▼	Autocomplete matching: Contains Textfield size: 60 No placeholder
+ Illustrations	Image ▼	Preview image style: Thumbnail (100×100) Progress indicator: throbber
+ Summary	Text area (multiple rows) ▼	Number of rows: 5
+ Video illustrations	Video Textfield ▼	
+ Meta Information	Details ▼	Default state closed Mark as required
+ Crisis Cycle Phase	Autocomplete ▼	Autocomplete matching: Contains Textfield size: 60 No placeholder
+ Crisis size	Autocomplete ▼	Autocomplete matching: Contains Textfield size: 60 No placeholder
+ Innovation stage	Autocomplete ▼	Autocomplete matching: Contains Textfield size: 60 No placeholder
+ Readiness	Autocomplete ▼	Autocomplete matching: Contains Textfield size: 60 No placeholder
+ Supported standards	Autocomplete ▼	Autocomplete matching: Contains Textfield size: 60 No placeholder
+ Administrative	Details ▼	Default state closed Mark as required
+ I accept that the content provided by me may be forwarded to interested participants.	Single on/off checkbox ▼	Use field label: Yes
+ Publish?	Single on/off checkbox ▼	Use field label: Yes
+ Publication approved?	Single on/off checkbox ▼	Use field label: Yes
+ QA comments	Text area (multiple rows) ▼	Number of rows: 5
+ Terms and conditions confirmation	Single on/off checkbox ▼	Use field label: Yes

Figure 3.1: Form display

Figure 3.2 shows the display mode of the solution description template. In Drupal, display mode is used to define how information is going to be presented on the website's frontend. To extend the possibilities, a contributed module "Display suite" has been installed and configured which allowed adding different elements, such as tab, HTML element, accordion etc. In addition, different view modes have been added to reuse the same information but to present it differently in the related context. For the solution description

template other contributed modules, as well as CSS supported graphical design techniques, were used to fine-tune the content presentation to make the information more readable and user-oriented.

FIELD	LABEL	FORMATTER	WIDGET	OPERATIONS
Header				
✚ EVA: QA approved EVA - QA approved EVA				
✚ View PDF	PDF export			
✚ Contact		Details	Default state closed	delete
✚ Provider	Above	Link	Link text trimmed to 180 characters Open link in new window Field template: default	
✚ EVA: EVA Anonymous users - EVA				
✚ EVA: Solution landing page(EVA/viewgroup content) - Solution members EVA				
✚ Summary	Above	Default	Field template: default	
✚ References		Fieldset		delete
✚ EVA: Solution landing page(EVA/viewgroup content) - Solution references				
✚ Related Solution(s)	Above	Label	Link to the referenced entity Field template: expert	
Left				
✚ Innovation stage	Above	Label	Link to the referenced entity Field template: expert	
✚ Readiness	Above	Label	Link to the referenced entity Field template: expert	
✚ Crisis size	Above	Label	Link to the referenced entity Field template: expert	
✚ Crisis Cycle Phase	Above	Label	Link to the referenced entity Field template: expert	
✚ Supported standards	Above	Label	Link to the referenced entity Field template: expert	
Right				
✚ Illustrations	Above	View	View: illustrations_slideshow_for_trials_and_solutions Display: solutions_illustrations_block Arguments(s): Entity ID Empty views: Display empty views Multiple: Disabled Field template: default	
✚ EVA: Solution landing page(EVA/viewgroup content) - Solution documentation EVA				
✚ Video illustrations	Above	Video	Embedded Video (Responsive, autoplaying) Field template: default	
Footer				
✚ Supported Use Cases		Details	Default state open	delete
✚ EVA: Solution landing page(EVA/viewgroup content) - Solution Use Cases				
✚ Similar Solutions		Details	Default state open	delete
✚ EVA: Similar solutions views - similar Solutions EVA				

Figure 3.2: Display

3.2 Solution team management

The PoS prototype implements a team management functionality which allows defining and assigning individual roles to members of the group. The following roles are defined:

1. Solution owner:
 - a. Can add/view/modify and delete all solution information.
 - b. Can add other site users to (their) solution group.
 - c. Can assign roles to other solution group members.
 - d. Can remove a member from group.
 - e. Can delete the group.
2. Solution member:
 - a. Can add/view/modify all solution information, but not delete content provided by other members.
3. Contact:
 - a. No rights to change group content, but the member with this (sub-)role is advertised on the solution landing page and can be contacted by other site users.

3.3 Solution use case

In order to support solution providers in defining use cases for their solutions, a predefined template, consisting of the following fields was implemented:

1. "Attachments", file upload field to store any additional information in a file form.
2. "Illustrations", image upload field.

3. “Related CM functions”, entity reference field used to establish relations to the CM functions taxonomy vocabulary.
4. “Summary”, formatted textual field to describe the use case.

“Use cases” entity was then added to the solution description entity as a sub-entity by utilizing the functionality of “Group” and “Group node” contributed modules. The same was done for all other sub-entities that will further be elaborated in the following sections.

One of the main advantages of the PoS prototype is based on the usage of the CM functions taxonomy, which was defined during the course of the project and described in the **D934.10 Taxonomy of CM functions for classification of solutions** (7). Figure 3.3 shows one example of the structure that the taxonomy has, to illustrate how the classification of solutions is performed. The “Use cases” template makes use of this by adding references to taxonomy terms which are later used to implement many other functionalities of the prototype, which will be further elaborated in the following sections.

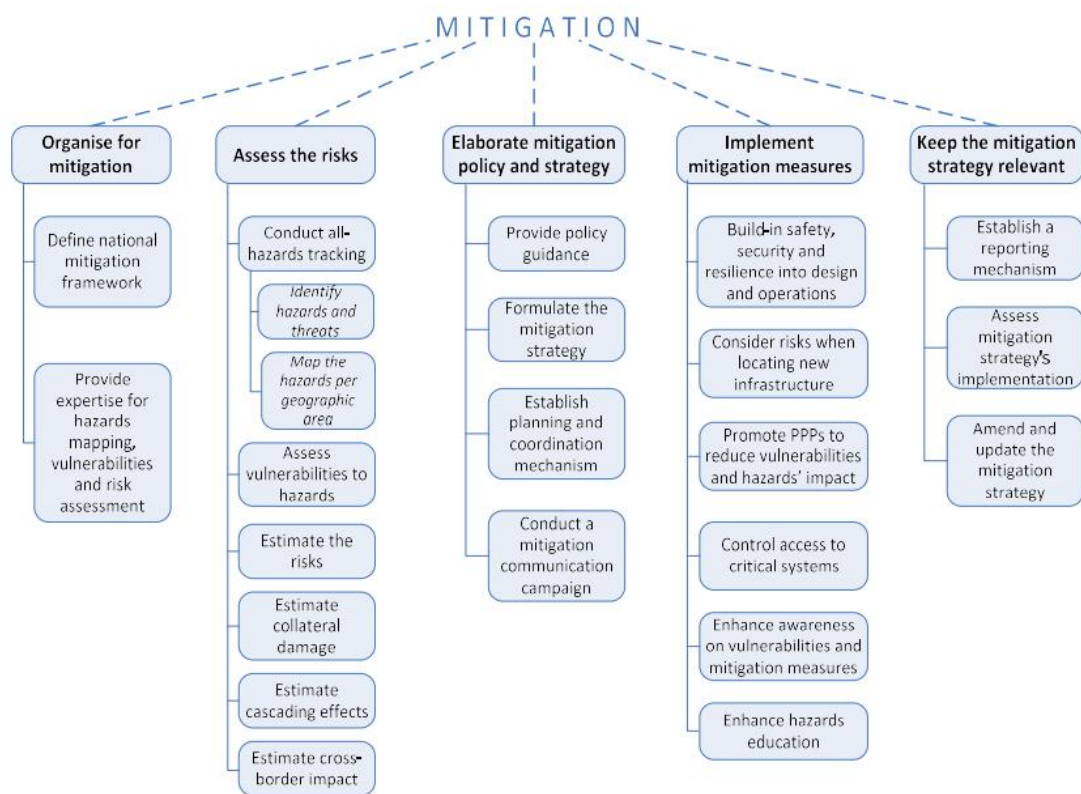


Figure 3.3: Example structure of CM functions taxonomy

Figure 3.4 illustrates form display that was used to define how create/edit form of the template is going to be presented to the user. In order to allow adding of CM functions to a use case, a contributed module “Entity browser” was implemented, which provides a generic entity browser/selector that can be used in any context.

FIELD	WIDGET	
✚ Title	Textfield ▼	Textfield size: 60
✚ Summary	Text area with a summary ▼	Number of rows: 9 Number of summary rows: 3
✚ Illustrations	Image ▼	Preview image style: Thumbnail (100×100) Progress indicator: throbber
✚ Related CM functions	Entity browser ▼	Entity browser: CM functions Selection mode: Append to selection Entity display: Entity label
✚ Attachments	File ▼	Progress indicator: throbber
✚ Language	Language select ▼	
✚ Translation		

Figure 3.4: Use case form display

In order to make the presentation of the data stored in the “Use cases” template, several display modes were defined, example shown in Figure 3.5.

FIELD	LABEL	FORMATTER	WIDGET
Content			
✚		Fieldset ▼	
✚ Summary	- Hidden - ▼	Default ▼	Field template: default
✚ Related CM functions	Above ▼	Label ▼	Link to the referenced entity Field template: expert
✚ Illustrations	- Hidden - ▼	Image ▼	Original image Field template: default
✚ Attachments	- Hidden - ▼	Generic file ▼	Use description as link text Field template: default

Figure 3.5: Use case display - default

3.4 Solution “used/tested in” reference

In order to support solution providers in adding references to where their solution has been used or tested, a predefined template consisting of the following fields was implemented:

1. “Documentation”, file upload field used to store information in a file form.
2. “External reference”, link field used to store information in a form of a hyperlink.
3. “Summary”, formatted textual field to describe the reference.
4. “Trial reference”, entity reference field used to establish relations to the TGT prototype (section 4).

By implementing the “Trial reference” field, the PoS prototype allowed easy re-use of the data that is stored in the TGT prototype, where a direct link could be established to any entity that was defined as described in the section 4, and the information would be displayed in the PoS prototype as a part of the solution entity that established the relation.

Figure 3.6 illustrates the form display that was used to define the presentation of the template to the user in create/edit form. In order to support the user in establishing relation to the TGT prototype entities, a contributed “Entity browser” module was implemented, similarly to how it was used for the “Use cases” prototype, described in the section 3.3.

FIELD	WIDGET	
✚ Title	Textfield ▼	Textfield size: 60
✚ Summary	Text area (multiple rows) ▼	Number of rows: 5
✚ Language	Language select ▼	
✚ Trial reference	Entity browser ▼	Entity browser: Trials Selection mode: Append to selection Entity display: Entity label
✚ External reference	Link ▼	No placeholders
✚ Documentation	File ▼	Progress indicator: throbber

Figure 3.6: Used/tested in form display

Figure 3.7 illustrates how the contributed “Entity browser” module was implemented in “Used/tested in” template to add references. The user is presented with the list of possible entities to add as references. The desired entity can be marked by using the checkboxes on the left-hand side and selected by clicking on the “Select Trial” button.

▼ TRIAL REFERENCE

Use this field to reference to a full Trial description. Alternatively, you can leave this field empty and add external reference instead.

Apply

<input type="checkbox"/>	Test trial
<input type="checkbox"/>	DRIVER+ Trial 1 – Poland
<input type="checkbox"/>	DRIVER+ Trial 2 – France
<input type="checkbox"/>	DRIVER+ Trial 3 – AUSTRIA
<input type="checkbox"/>	DRIVER+ Trial 4 – The Netherlands

Select trial

Figure 3.7: Example of Entity browser implementation

In order to present the data stored in the “Used/tested in” template, a display mode as shown in the Figure 3.8 was implemented.

FIELD	LABEL	FORMATTER	WIDGET
Content			
✚ Title	- Hidden - ▼	Default ▼	Link: no Wrapper: h4 Field template: default
✚ Details		Details ▼	Default state closed
✚ Summary	- Hidden - ▼	Default ▼	Field template: default
✚ Trial reference	Above ▼	Rendered entity ▼	Rendered as Teaser Field template: default
✚ External reference	Above ▼	Link ▼	Link text trimmed to 80 characters Field template: default
✚ Documentation	Above ▼	Generic file ▼	Use description as link text Field template: default

Figure 3.8: Used/tested in display – default

3.5 Solution documentation

In order to support solution providers in adding documentation to their solution descriptions, a predefined template consisting of the following fields has been implemented:

1. “Documentation”, file upload field used to store information in a file form.
2. “Documentation type”, entity reference field to reference extendable taxonomy which is used to categorise solution documentation.
3. “External reference”, a link field used to store information in a form of a hyperlink.
4. “Summary”, formatted textual field to describe the documentation.

Figure 3.9 illustrates the form display used to define how the template is presented to the user in create/edit form.

FIELD	WIDGET	
✚ Title	Textfield ▼	Textfield size: 60
✚ Documentation type	Select list ▼	
✚ Summary	Text area (multiple rows) ▼	Number of rows: 5
✚ Documentation	File ▼	Progress indicator: throbber
✚ External reference	Link ▼	No placeholders

Figure 3.9: Documentation template form display

Figure 3.10 illustrates a display mode defined to present the data stored in the “Solution documentation” template.

FIELD	LABEL	FORMATTER	WIDGET
Content			
⊕ Title	- Hidden - ▼	Default ▼	Link: no Wrapper: h4 Field template: default
⊕ Details		Details ▼	Default state closed
⊕ Documentation type	- Hidden - ▼	Label ▼	No link Field template: default
⊕ Summary	Inline ▼	Default ▼	Field template: default
⊕ Documentation	Inline ▼	Generic file ▼	Use description as link text Field template: default
⊕ External reference	Inline ▼	Link ▼	Link text trimmed to 80 characters Field template: default

Figure 3.10: Documentation template display - default

3.6 Solution feedback

The PoS prototype implements a feedback session, where all site users can store information about a solution. In order to implement this functionality, a template with following fields has been implemented:

1. “Feedback”, formatted textual field used to store feedback.
2. “Rating”, custom field used to visualise feedback.
3. “Approved”, Boolean field used to indicate if feedback is to be shown or not.
4. “Reply”, formatted textual field used to store a reply to feedback.

The way that the fields 3 and 4 were implemented differs to previously described implementations – they utilize the contributed “Group node” entity in a way that these fields are added directly to group entity. To better illustrate the need for such implementation, explanation of the fields can be used:

- “Publication status”, Boolean field to indicate if certain entity should be shown to other users of the website or not. By using the “Group node” contributed module, it is possible to set this value for particular entity individually that belong to one solution.
- “QA Summary”, formatted textual field used to store information about an entity. Similarly, this field allows storing information for particular entities of one solution. Further explanations and usage of this implementation step can be found in section 4 of the TGT prototype description, where it is explained in more detail, since it plays a more significant role in the implementation of the TGT prototype.

Figure 3.11 and Figure 3.12 illustrate form displays used to define how feedback template is displayed to the user in the create/edit form.

FIELD	WIDGET
⊕ Title	Textfield ▼
⊕ Feedback	Text area (multiple rows) ▼
⊕ Rating:	Star rating ▼

Figure 3.11: Feedback form display

FIELD	WIDGET	
⊕ Approved	Single on/off checkbox ▼	Use field label: Yes
⊕ Reply:	Text area (multiple rows) ▼	Number of rows: 5
⊕ Language	Language select ▼	

Figure 3.12: Feedback group form display

Figure 3.13 illustrates display mode used to display feedback information. A custom formatter for the “Rating” field is realised with the help of a contributed “Star rating” module, which visualises the value of this field ranging from 0 to 10 as shown.


FIELD	LABEL	FORMATTER	WIDGET
Content			
⊕ Feedback	- Hidden - ▼	Default ▼	Field template: default
⊕ Rating:	Above ▼	Star rating ▼	 Field template: default

Figure 3.13: Feedback display

3.7 Solution search and matching functionality

In order to support practitioners and other interested parties in searching for innovative solutions, the PoS prototype implements an overview page where all solutions that are stored in the database and published are shown with a possibility to filter them and search for relevant terms. Filtering functionality is implemented with the help of the contributed “Facets” module, that was configured to use predefined taxonomy terms that are a part of solution description entities as filtering parameters, such are crisis cycle phase taxonomy, CM functions taxonomy, etc. Since the provided native visualisation of data was not sufficient, custom rendering had to be used to make the overview page more appealing.

Figure 3.14 illustrates the overview page of the PoS prototype.

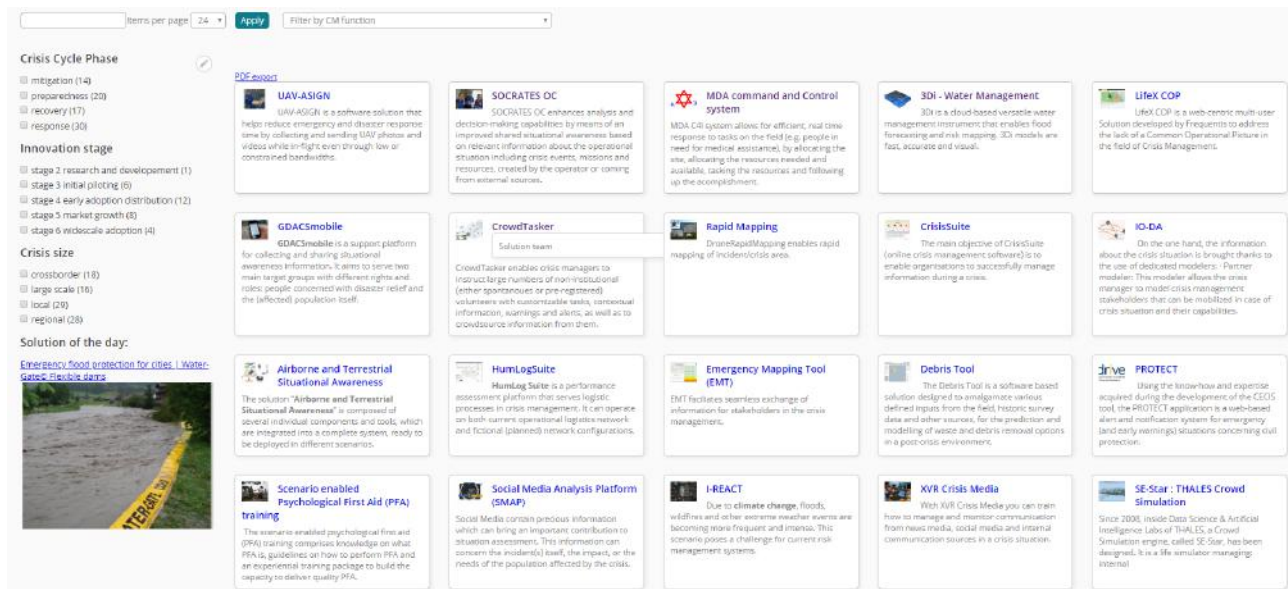


Figure 3.14: Solutions overview page

The overview page is based on a search index, which is an implementation of the Drupal's "Search API" core module. It allows searching for terms throughout all entities and their parts. Figure 3.15 illustrates the configuration of the search index that is used in the PoS prototype. Both full text search and filters can be combined to assure narrowing down of results to the most relevant ones. During the implementation, a problem with the "Search API" in which duplicates were occurring in search results was identified and later fixed with a patch to the module. The issue was documented in Drupal community as well.

LABEL	MACHINE NAME	PROPERTY PATH	TYPE	BOOST	OPERATIONS
gid	gid	id	Integer		Remove
Group Solution Summary	solution_summary	field_solution_summary	Fulltext	5.0	Remove
Group Solution Title	solution_title	label	Fulltext	8.0	Remove
Group type	type	type	String		Remove
MC: Crisis Cycle Phase » Taxonomy term » Name	solution_crisis_cycle_phase	field_crisis_cycle_phase.entity.name	String		Remove
MC: Crisis size » Taxonomy term » Name	solution_crisis_size	field_solution_crisis_size.entity.name	String		Remove
MC: Innovation stage » Taxonomy term » Name	solution_innovation_stage	field_solution_innovation_stage.entity.name	String		Remove
Node published?	field_publish	field_publish	Boolean		Remove
QA approved? (group)	field_qa_approved	field_qa_approved	Boolean		Remove
QA approved? (group nodes)	field_group_nodes_approved	search_api_reverse_entity_references_group_content__gid_field_approved	Boolean		Remove
UC » Body » Processed text	uc_body_processed	search_api_reverse_entity_references_group_content__gid_entity_id.entity.body.processed	Fulltext	1.0	Remove
UC » Related CM functions » Taxonomy term » Desc	uc_cm_description	search_api_reverse_entity_references_group_content__gid_entity_id.entity.field_cm_functions.entity.description	Fulltext	1.0	Remove
UC » Related CM functions » Taxonomy term » Desc	uc_cm_description_processed	search_api_reverse_entity_references_group_content__gid_entity_id.entity.field_cm_functions.entity.description.processed	Fulltext	1.0	Remove
UC » Related CM functions » Taxonomy term » Name	uc_cm_name	search_api_reverse_entity_references_group_content__gid_entity_id.entity.field_cm_functions.entity.name	String		Remove
UC » Related CM functions » Taxonomy term » Name	uc_cm_name_2	search_api_reverse_entity_references_group_content__gid_entity_id.entity.field_cm_functions.entity.name	Fulltext	3.0	Remove
UC » Related CM functions » Taxonomy term » Term	uc_cm_tid	search_api_reverse_entity_references_group_content__gid_entity_id.entity.field_cm_functions.entity.tid	Integer		Remove
UC » » Related CM functions » Taxonomy term » Pub	uc_cm_status	search_api_reverse_entity_references_group_content__gid_entity_id.entity.field_cm_functions.entity.status	Boolean		Remove

Figure 3.15: Search index configuration

In addition, a "similar solutions"-functionality has been implemented as a special view that calculates the similarity of solutions based on overlap in supported CM functions. A list of the similar solutions is automatically attached to solution landing page, as illustrated in Figure 3.16.

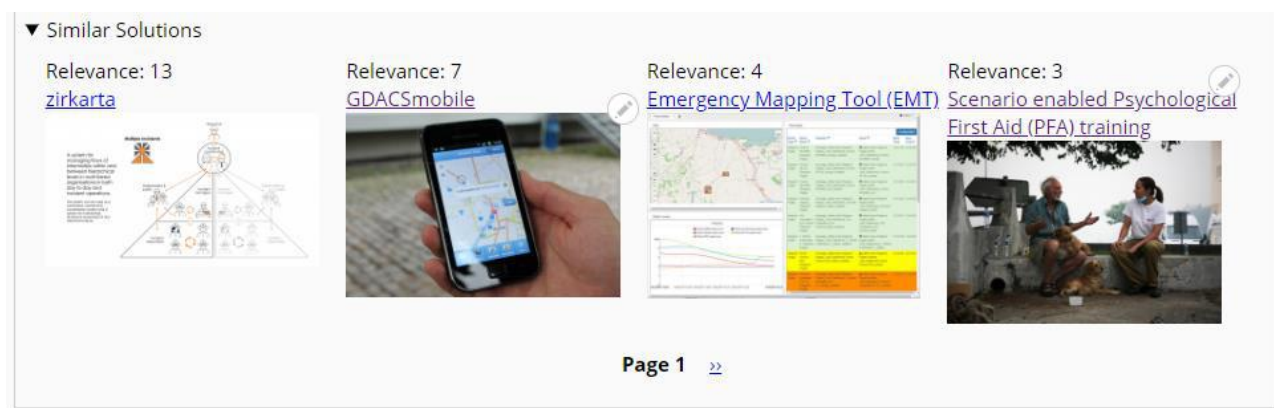


Figure 3.16: Similar solutions

3.8 Solution exports

In order to provide a Restful export functionality to enable communication with other systems, Drupal's core "Rest API" module was implemented and configured. Furthermore, to ease the usage of it, a contributed "Swagger UI Field Formatter" module that implements the swagger library (<https://swagger.io/>) was used, which visualises Restful requests to the user, illustrated in Figure 3.17. The Drupal platform allows defining dedicated views to indicate which fields are going to be exported, similarly to Drupal views that are used to display data from the database.

An example of how a Restful call looks like in a browser:

`"/group_export?_format=json&type=solution&search=crowd"`.

The result of this call would return all groups of the type solution which contain a term "crowd", and the format of the export would be "json", which is a lightweight data-interchange format. It is easy for humans to read and write and for machines to parse and generate.

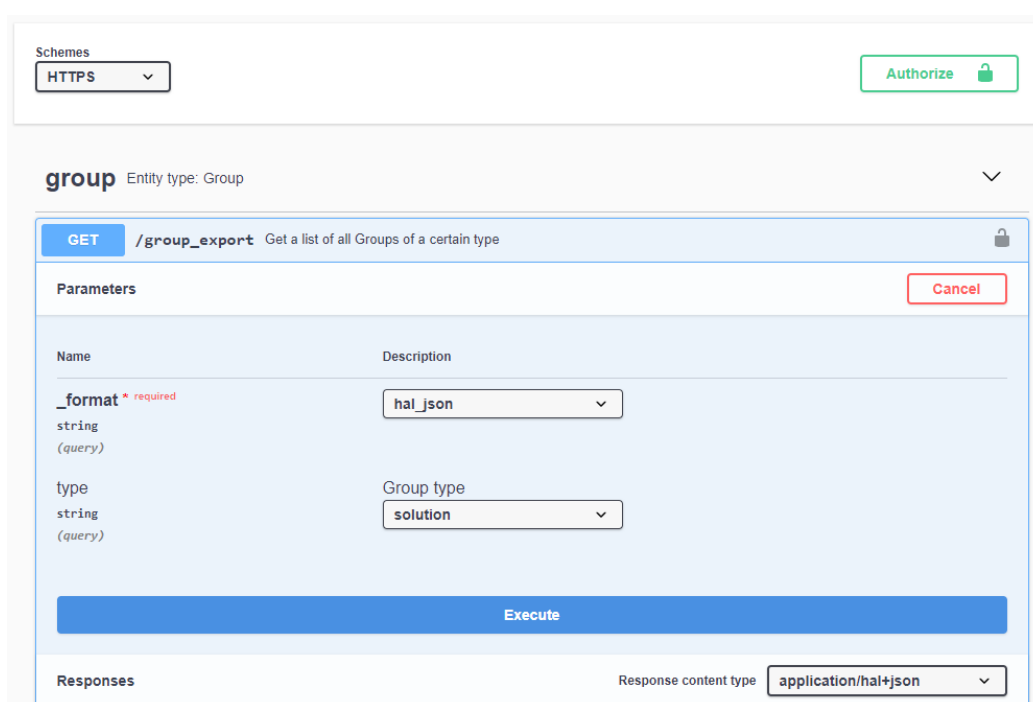


Figure 3.17: Visualised Restful request

Additionally, the PoS prototype implements PDF export functionality as described in section 2.9.

3.9 Related solutions

The PoS prototype implements a functionality of indicating which solutions can work together, in a sense that they can exchange information. This functionality requires information stored in a test case (section 4.10), therefore can only be implemented if both prototypes are implemented. Figure 3.18 illustrates how this information is displayed to the user.

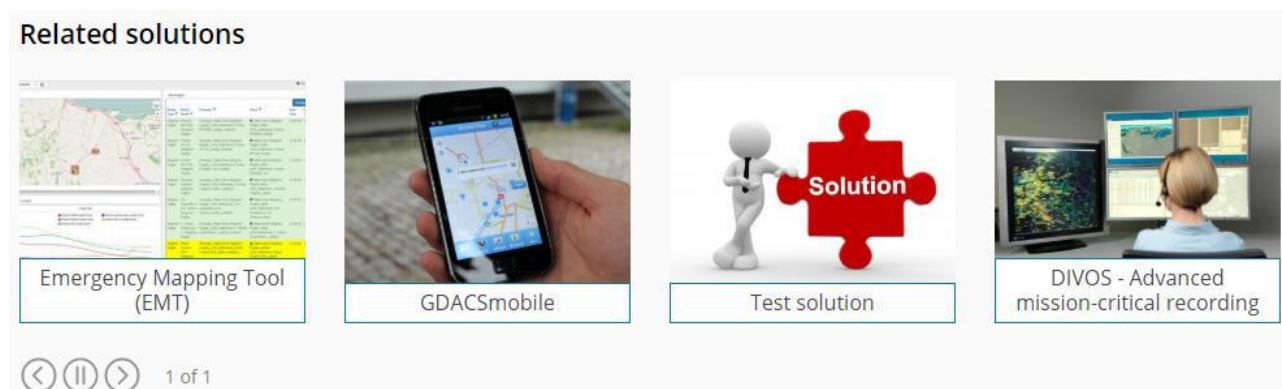


Figure 3.18: Related solutions

3.10 Group-level solution validation

In order to provide automatic support to solution providers when describing their solutions, a custom validation function was implemented in the PoS prototype. The main goal is to provide feedback to the user if the defined criteria for solution description have been met. The idea is to check if all required entities have been created and published. Examples of positive and negative validation results are shown in figures Figure 3.19 and Figure 3.20.

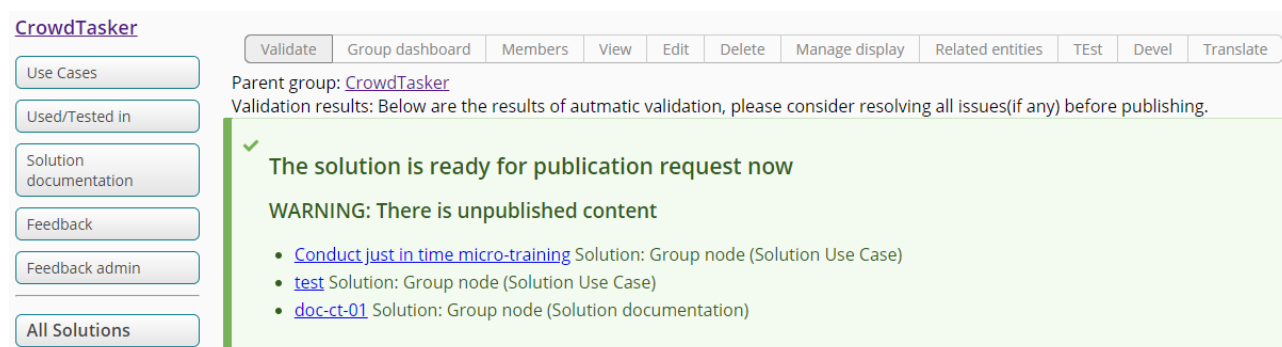


Figure 3.19: Positive validation result

SOCRATES OC

Validate Group dashboard Members View Edit Delete Manage display Related entities TEst Devel Translate

Parent group: [SOCRATES OC](#)

Validation results: Below are the results of automatic validation, please consider resolving all issues(if any) before publishing.

✗ The solution has unmet requirements for publication:

Missing Solution content

- CRITICAL: please define at least contact (group member role)
- CRITICAL: please define and publish at least one Solution Use Case
- RECOMMENDED: please indicate where the solution was used or tested)

WARNING: There is unpublished content

- [SOR-drazen-2019-03-27](#) Solution: Group node (Solution reference)

Figure 3.20: Negative validation result

3.11 Solution landing pages

After all entities were implemented as described in sections 3.3 to 3.6, the following implementation step was to implement pages where information stored about a solution is going to be presented. In order to do so, “Views” were utilised. This is a characteristic functionality of the chosen Drupal framework, which allows easy configuration and implementation of different views for any data that is stored in the database. The queries are built with the help of “Views” and the information is retrieved from the database where the user can directly influence the way it is going to be displayed. An example of such view is shown in the Figure 3.21 and a resulting outcome of a query that was executed on the PoS prototype’s database is shown in Figure 3.22.

When implementing a view, desired tables from the database are chosen, relationships to other tables are established if needed and the values of desired fields are displayed. For displaying solution related information in the PoS prototype several views have been implemented and configured according to the requirements of the prototype.

Master Solution Use Cases Page Solution References Page Solution Documentation Pa... Add

Edit view name/description

Display name: Solution Use Cases Page Duplicate Solution Use Cases Page

TITLE
Title: Use Cases

FORMAT
Format: Unformatted list / Settings
Show: Fields / Settings

FIELDS Add

- (Content relation) Edit node link (Link to edit Content) [hidden]
- (Content relation) Delete node link (Link to delete Content) [hidden]
- View relation link [hidden]
- Edit relation link [hidden]
- Global: Dropbutton (Operations on Content) [hidden]
- Dropbutton: Edit status & relations [hidden]
- Last change date [hidden]
- (Content relation) Rendered content (compact) [hidden]
- (Content relation) New or updated? [hidden]
- (Content relation) Last changed (when and by whom) [hidden]
- Rendered status & references (QA) [hidden]
- Approved or not - used to add class to each row (QA Status) [hidden]
- Render all

FILTER CRITERIA Add

- (Content relation) Content: Published (= Yes)
- Group content: Group content type (= Solution: Group node (Solution Use Case))
- (Content relation) Content: Translation language (= Interface text language selected for page)

SORT CRITERIA Add

- (Content relation) Content: Title (Exposed)
- (Content relation) Content: Changed (Exposed)

PAGE SETTINGS

Path: /group/%group/solution_u...
Menu: No menu
Access: Group permission | Leave group

HEADER Add

FOOTER Add

NO RESULTS BEHAVIOR Add

Global: Rendered entity - Block (Global: Rendered entity - Block)

PAGER

Use pager: Mini | Mini pager, 10 items
More link: No

LANGUAGE

Rendering Language: Content language of view row

ADVANCED

CONTEXTUAL FILTERS Add

Group content: Parent group

RELATIONSHIPS Add

Content relation

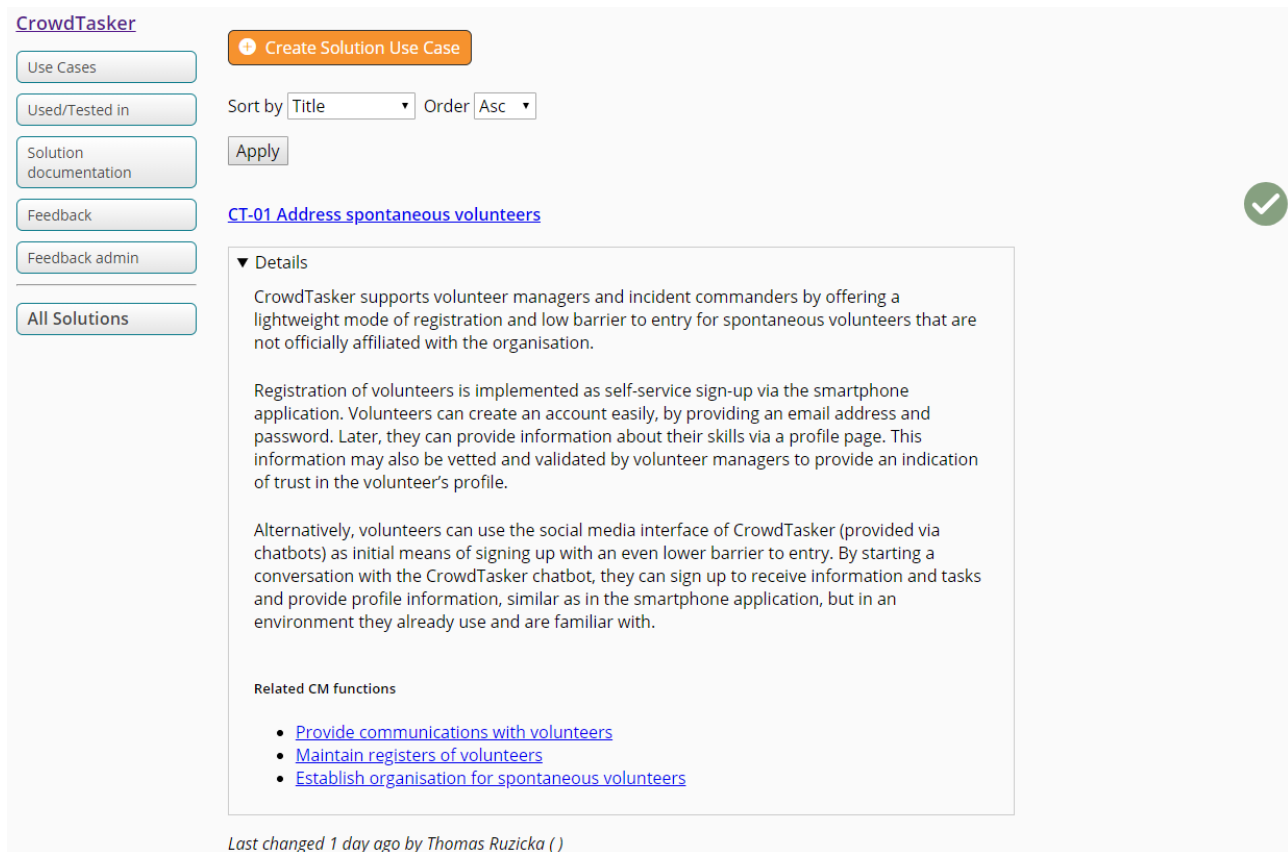
EXPOSED FORM

Exposed form in block: No
Exposed form style: Basic | Settings

OTHER

Machine Name: solution_use_cases_page
Administrative comment: None
Use AJAX: No
Hide attachments in summary: No
Contextual links: Shown
Use aggregation: No
Query settings: Settings
Caching: Tag based
CSS class: None

Figure 3.21: “Use cases” view



The screenshot shows the 'CrowdTaker' interface with a sidebar on the left containing buttons: 'Use Cases', 'Used/Tested in', 'Solution documentation', 'Feedback', 'Feedback admin', and 'All Solutions'. The main content area has a top bar with a 'Create Solution Use Case' button and sorting options (Sort by: Title, Order: Asc). Below this is the title 'CT-01 Address spontaneous volunteers' and a 'Details' section. The details section contains three paragraphs of text describing volunteer registration and social media integration. Below the text is a section titled 'Related CM functions' with three bullet points linking to other use cases. At the bottom, it says 'Last changed 1 day ago by Thomas Ruzicka ()'. A green checkmark icon is visible in the top right corner of the interface.

CrowdTaker

Create Solution Use Case

Use Cases

Used/Tested in

Solution documentation

Feedback

Feedback admin

All Solutions

Sort by: Title Order: Asc

Apply

CT-01 Address spontaneous volunteers

▼ Details

CrowdTaker supports volunteer managers and incident commanders by offering a lightweight mode of registration and low barrier to entry for spontaneous volunteers that are not officially affiliated with the organisation.

Registration of volunteers is implemented as self-service sign-up via the smartphone application. Volunteers can create an account easily, by providing an email address and password. Later, they can provide information about their skills via a profile page. This information may also be vetted and validated by volunteer managers to provide an indication of trust in the volunteer's profile.

Alternatively, volunteers can use the social media interface of CrowdTaker (provided via chatbots) as initial means of signing up with an even lower barrier to entry. By starting a conversation with the CrowdTaker chatbot, they can sign up to receive information and tasks and provide profile information, similar as in the smartphone application, but in an environment they already use and are familiar with.

Related CM functions

- [Provide communications with volunteers](#)
- [Maintain registers of volunteers](#)
- [Establish organisation for spontaneous volunteers](#)

Last changed 1 day ago by Thomas Ruzicka ()

Figure 3.22: Result of the “Use cases” view

3.12 CM gaps

According to DRIVER+ terminology (8), CM Gaps are defined as “difference between the existing capabilities of responders and what was actually needed for effective and timely response”. 21 such gaps were defined as “strategic CM gaps” in **D922.11 List of CM gaps** (9).

To allow the practitioners to define own CM Gaps, a predefined template, consisting of the following fields was implemented within the PoS prototype:

1. “Documentation”, file upload field used to store information in a form of a document.
2. “Gap description”, formatted textual field to describe the gap.
3. “Rationale & related CM function(s)”, entity reference field used to reference a paragraph – data structure in Drupal that contains several fields but can be imagined as a single field. In this case, following fields were used:
 - a. “CM function reference”, entity reference field used to reference the CM functions taxonomy vocabulary.
 - b. “Rationale”, textual field used to describe why certain CM functions are relevant.

Figure 3.23 shows the form display for the CM gap entity, which defines how fields are presented to the user in create and edit forms.

FIELD	WIDGET	
+ Language	Language select ▼	
+ Title	Textfield ▼	Textfield size: 60
+ Gap description	Text area (multiple rows) ▼	Number of rows: 5
+ Rationale & related CM function(s)	Paragraphs Classic ▼	Title: Rationale & related CM function(s) Plural title: Rationales & related CM functions Edit mode: Open Add mode: Dropdown button Form display mode: default Default paragraph type: Rationale and related CM functions
+ Documentation	File ▼	Progress indicator: throbber
+ Translation		

Figure 3.23: CM gap form display

Figure 3.24 shows the display mode for the CM gap entity, which enables configuration of how the information stored in the fields of the entity is presented to the user.

FIELD	LABEL	FORMATTER	WIDGET
Content			
+ Title	- Hidden - ▼	Default ▼	Link: no Wrapper: h2 Field template: default
+ Gap description	- Hidden - ▼	Default ▼	Field template: default
+ Documentation	Above ▼	Generic file ▼	Use description as link text Field template: default
+ Rationale & related CM function(s)	Above ▼	Rendered entity ▼	Rendered as Default Field template: default

Figure 3.24: CM gap display

PoS platform automatically links CM gaps to possible solutions. Relevance of the solutions for the CM gaps is estimated by calculating the level of overlap between the CM functions that are related to the gap and the CM functions that are supported by solutions.

An overview page with all the gaps has also been implemented. This page allows full text search on gaps themselves and related CM functions. Figure 3.25 illustrates how a defined gap is displayed together with solutions that might be used to bridge it.

Managing spontaneous volunteers

▼ Details

Insufficiencies in the management of spontaneous volunteers on the crisis scene in terms of location, tasking, capabilities, and shift duration

Rationale & CM functions

Communicate with, assess the abilities, task, track the location and engagement (e.g. shift duration) of spontaneous volunteers during a crisis.

- [Manage spontaneous volunteers](#)

Establish organisation needed to identify and register spontaneous volunteers and to assign them to teams and coordinators; assure that an adequate legal basis is in place.

- [Establish organisation for spontaneous volunteers](#)

Define responsibilities and procedures for tasking spontaneous volunteers, offering their support to response and recovery operations, and 'digital volunteers' willing to provide support on social media.

- [Prepare for crowd tasking](#)

The support of unaffiliated and 'digital' volunteers will be facilitated when opportunities for application of relevant modes of crowd sourcing are established and the information on such opportunities is widely disseminated.

- [Provide for crowd sourcing](#)

Crisis communications and information management documentation and procedures need to provide opportunities to receive information from and manage spontaneous volunteers.

- [Regulate access to CM communications and information](#)

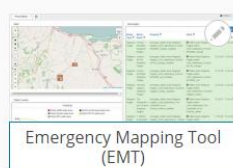
▼ Solutions addressing this gap

Relevance: 3



CrowdTasker

Relevance: 1



Emergency Mapping Tool (EMT)

Figure 3.25: CM gap

4. Trial Guidance Tool

The TGT is a web-based supportive tool whose main objective is to provide guidance to the Trial organisers and practitioners in successful implementation of the methodology by providing assistance and validation, and also to document the results and store them for future reference. It is directly derived from the TGM and therefore also represents an interactive version of the **TGM handbook** (10).

The main idea of the TGT prototype is to have one main entity called “Trial” which can be extended with several other sub-entities, namely “Trial gaps”, “Trial objectives”, “Trial research questions”, “Trial data collection plan”, “Trial evaluation approaches and metrics”, “Trial scenario”, “Trial Solutions”.

The main functionalities that the TGT provides are:

1. Overview page to display information about all Trials published with the help of TGT.
2. Predefined template to provide basic description of Trials.
3. Trial team management capabilities.
4. Predefined template to describe Trial gaps with relations to CM functions taxonomy.
5. Predefined template to describe Trial objectives.
6. Predefined template to describe Trial research questions.
7. Predefined template to describe Trial data collection plan.
8. Predefined template to describe Trial evaluation approaches and metrics.
9. Predefined template to describe and upload Trial scenario.
10. Predefined template to describe Trial solutions.
11. Predefined template to describe test cases.
12. Check-lists for all Trial steps in all Trial phases.
13. Trial search and matching.
14. PDF export to extract all information from the TGT in a form of a document.
15. Searchable knowledge database.
16. Predefined template to document lessons learned.
17. Validation function to automatically check if Trial steps were followed as defined.
18. Trial landing pages to display all information about a Trial.

Table 4.1 provides an overview of the implementation status for each of the functionalities, with functionality names corresponding to the sub-section names in section 4 of **D933.11** (2), with exception of the sections 4.17 in this document, which didn’t exist in the previous document. Additional comment field summarizes challenges encountered, implementation issues and change with regards to **933.11** (2) specifications.

Description of implementation status values:

- *Implemented* – the functionality has been fully implemented and is available on the prototypes;
- *in course of implementation* – the functionality has been partially implemented and may not be available on the public version of the PoS and TGT prototype;
- *stalled* – the functionality has not been implemented for a specific reason which is stated in the comment field.

Table 4.1: TGT functionalities: implementation vs. 933.11 specifications

Functionality	Implementation status	Comment
Trial group	Implemented	
Trial team management	Implemented	
Trial gap	Implemented	

Functionality	Implementation status	Comment
Trial objective	Implemented	
Trial research question	Implemented	
Trial data collection plan	Implemented	
Trial evaluation approaches and metrics	Implemented	As indicated in section 4.9 of the D933.11 (2), specifications for this functionality became obsolete, and no explicit support by the TGT prototype is foreseen, other than to provide a template to store information as explained in section 4.7.
Trial scenario	Implemented	
Solution selection	Implemented	
Test case	Implemented	
Execution phase	Implemented	Since no implementable requirements that are similar to those provided for preparation phase (section 4.3 to 4.9) were defined, this functionality is realized as a textual information with a checklist.
Evaluation phase	Implemented	Since no implementable requirements that are similar to those provided for preparation phase (section 4.3 to 4.9) were defined, this functionality is realized as textual information with a checklist.
Trial search and matching	Implemented	
Trial exports	Implemented	
Knowledge Database	Implemented	
Group-level Trial validation	Implemented	

4.1 Trial group

Trial group template is used to provide basic information about a Trial together with information regarding the Trial context and it is divided into four tabs:

1. Trial Summary.
2. Trial Context.
3. Meta Information.
4. Administrative.

For the Trial group template, the following types of fields have been implemented:

1. Entity reference, field used to add a reference to other content on the website.
2. Entity reference revisions, field used to reference paragraphs – data structure in Drupal that contains several fields but can be imagined as a single field.
3. Text, textual field used for storing different textual information types, such are long, short, formatted etc.
4. Boolean, field used to store information that has one of two possible values.
5. Image, field used to store uploaded image file with related information.

As within the solution group template described in section 2, all fields that were used have individual settings together with some that are common for all, such as:

1. Field label, a help text to be shown to the user under the field.
2. Required status, information if the field is mandatory or not.
3. Number of values for a field - ranging from 1 to unlimited number of values, etc.

Entity reference fields that were used for the Trial group template can be divided into two groups:

1. Fields that reference taxonomy vocabularies:
 - a. "Crisis cycle phase" (Preparedness, Risk assessment, Response, Recover, Mitigation).
 - b. "Crisis size" (Cross-border, Large scale, Regional, Local).
 - c. "Incident category" (top level terms: Incidents in general, Natural incident, Intentional incidents/attacks, Technological incident, Natural and technological)
 - d. "Trial type" (Trial, Table top, Field exercise, Mixed approach, Solution test).
2. Fields that reference other content.
 - a. "Cooperation partners", field referencing "Organisation" extendable content type which was implemented to store information about organisations, using Drupal's standard fields.
 - b. "Lead organisation", referencing the same content type as the previous field, with the difference in the number of values it can store.

Two types of text fields are used in the solution description template:

1. Formatted long:
 - a. "Doctrines, standards, laws, etc", HTML formatted table used to store information about doctrines, standards and laws that are applicable to a Trial.
 - b. "Ethical, legal, social issues (ELSI)", HTML formatted table used to store information about ELSI issues.
 - c. "Overarching objective", used to store the information about the overarching objective of the Trial which is also shown on Trial objectives page, described in section 4.4.
 - d. "Overarching scenario", used to store the information about the overarching scenario of the Trial which is also shown on Trial scenario page, described in section 4.8.
 - e. "Setting/type of area", HTML formatted table used to store information about the setting and the type of area of the Trial.
 - f. "Special equipment/facility", HTML formatted table used to store information about the special equipment and facilities that are needed for the Trial.
 - g. "Trial Description", used to store general description of a Trial.
 - h. "When are we?", HTML formatted table used to store information when is the Trial taking place.
 - i. "Where we are?", HTML formatted table used to store information where the Trial is taking place.
 - j. "Whom to involve and how?", HTML formatted table used to store information about the participants of the Trial.

Note: all fields labelled as HTML formatted table store the information that belongs to the Trial context template.
2. Plain long:
 - a. "QA comments", used to store plain text information regarding QA process.

Several Boolean fields were used in the Trial group template:

1. "IPR confirmation", field to indicate acknowledgment of IPR.
2. "QA approved?", field to indicate if the solution description meets the QA requirements.
3. "Request publication?", field to indicate that the solution provider wishes to publish the solution description.
4. "Terms and conditions confirmation", field to indicate agreement with site's terms and conditions of usage.

In addition, individual Boolean fields for every checklist provided in the TGT are also stored within the Trial data model together with them. This was necessary in order to implement the checklist (described in section 4.11) on a step level and not on a group entity level.

Figure 4.1 and Figure 4.2 illustrate two form modes that were defined for Trial group template. The difference between them is in the number of fields shown, where the “Add” form mode contains only the crucial information fields.

FIELD	WIDGET	
✚ Trial Skeleton	Fieldset	Mark as required
✚ Trial type	Select list	
✚ Trial Summary	Details	Default state open Mark as required
✚ Title	Textfield	Textfield size: 60
✚ Lead organisation	Autocomplete	Autocomplete matching: Contains Textfield size: 60 Placeholder: Start typing the organisation name
✚ Trial Description	Text area (multiple rows)	Number of rows: 5
✚ Trial illustrations	Image	Preview image style: Thumbnail (100x100) Progress indicator: chevron
✚ Administrative	Details	Default state open Mark as required
✚ Terms and conditions confirmation	Single on/off checkbox	Use field label: Yes

Figure 4.1: “Add” form mode

FIELD	WIDGET	
✚ Trial Summary	Details	Default state closed Mark as required
✚ Title	Textfield	Textfield size: 60
✚ Lead organisation	Autocomplete	Autocomplete matching: Contains Textfield size: 60 No placeholder
✚ Outstanding objectives	Text area (multiple rows)	Number of rows: 5
✚ Outstanding scenarios	Text area (multiple rows)	Number of rows: 5
✚ Cooperation partners	Autocomplete	Autocomplete matching: Contains Textfield size: 60 No placeholder
✚ Trial illustrations	Image	Preview image style: Thumbnail (100x100) Progress indicator: chevron
✚ Trial Description	Text area (multiple rows)	Number of rows: 5
✚ Trial Context	Details	Default state closed
✚ Ethical, legal, social issues (ELSI)	Text area (multiple rows)	Number of rows: 5
✚ Real world location or locations of the trial	Paragraphs EXPERIMENTAL	Title: Paragraph First title: Paragraph Edit mode: Open Closed mode: Summary Add mode: Open/Close button Form fields mode: default Default paragraph type: Location Features: Duplicate, Collapse, Edit all
✚ participants	Paragraphs EXPERIMENTAL	Title: Paragraph First title: Paragraph Edit mode: Open Closed mode: Summary Add mode: Open/Close button Form fields mode: default Default paragraph type: Trial participants table Features: Duplicate, Collapse, Edit all
✚ Setting type of area	Text area (multiple rows)	Number of rows: 5
✚ Special equipment/facility	Text area (multiple rows)	Number of rows: 5
✚ When are we?	Text area (multiple rows)	Number of rows: 5
✚ Where are we?	Text area (multiple rows)	Number of rows: 5
✚ Where to involve and how?	Text area (multiple rows)	Number of rows: 5
✚ Meta information	Details	Default state closed Mark as required
✚ Trial Location	Autocomplete	Autocomplete matching: Contains Textfield size: 60 No placeholder
✚ Crisis Cycle Phase	Autocomplete	Autocomplete matching: Contains Textfield size: 60 No placeholder
✚ Incident category	Autocomplete	Autocomplete matching: Contains Textfield size: 60 No placeholder
✚ Crisis size	Autocomplete	Autocomplete matching: Contains Textfield size: 60 No placeholder
✚ Trial type	Autocomplete	Autocomplete matching: Contains Textfield size: 60 No placeholder
✚ Administrative	Details	Default state closed Mark as required
✚ May reproduce?	Single on/off checkbox	Use field label: Yes
✚ Published?	Single on/off checkbox	Use field label: Yes
✚ Publication approved?	Single on/off checkbox	Use field label: Yes
✚ QA comments	Text area (multiple rows)	Number of rows: 5
✚ Terms and conditions confirmation	Single on/off checkbox	Use field label: Yes

Figure 4.2: Default form mode

Figure 4.3 shows an example of the display mode for the Trial description template. In Drupal, display mode is used to define how information is going to be presented on the website’s frontend. To extend the possibilities, a contributed module “Display suite” has been installed and configured which allowed adding

different elements, such as tab, HTML element, accordion etc. In addition, different view modes have been added to reuse the same information but to present it differently in the related context. For the Trial description template other contributed modules as well as CSS supported graphical design techniques were used to fine-tune the content presentation to make the information more readable and user-oriented.

FIELD	LABEL
Header	
✚ View PDF	PDF export
Contact	
✚ Lead organisation	Inline ▼
✚ EVA: EVA Anonymous users – EVA	
✚ EVA: Trial landing page(EVA)view – Trial members EVA	
✚ Trial Description	Above ▼
✚ EVA: Trial landing page(EVA)view – Trial objectives	
✚ EVA: Trial landing page(EVA)view – Trial research questions	
✚ May reproduce?	Above ▼
Left	
✚ Trial type	Above ▼
✚ Crisis size	Above ▼
✚ Crisis Cycle Phase	Above ▼
Right	
✚ Real world location or locations of the trial	Above ▼
✚ Trial illustrations	Above ▼
Footer	
✚ EVA: Trial landing page(EVA)view – Trial gaps	
Objective	
✚ Overarching objective	Above ▼
✚ Overarching scenario	Above ▼
Detailed Scenario	
✚ EVA: Trial landing page(EVA)view – Trial scenarios	

Figure 4.3: Display

4.2 Trial team management

The TGT prototype implements a team management functionality which allows defining and assigning of individual roles to members of the group. Following roles are defined:

1. Trial owner:
 - a. Can add/view/modify and delete all Trial information.
 - b. Can add other site users to (their) Trial group.
 - c. Can assign roles to other Trial group members.
 - d. Can remove a member from group.
 - e. Can delete the group.
2. Trial member:
 - a. Can add/view/modify all Trial information, but not delete content provided by other members.
3. Contact:
 - a. No rights to change group content, but the member with this (sub-)role is advertised on the Trial landing page and can be contacted by other site users.

4.3 Trial gap

DRIVER+ terminology (8) defines a Trial as “an event for systematically assessing solutions for current and emerging needs in such a way that practitioners can do this following a pragmatic and systematic approach.”. According to TGM, such needs are expressed as CM gaps (see 3.12).

To support Trial organisers in defining the gaps that are relevant for their Trial, the TGT prototype allows them to reference existing CM gaps from the PoS prototype platform, as well as to define new CM gaps directly from the “Trial gap” functionality. Within the TGT, the CM gaps are linked to the Trial group, resulting in the two-part template consisting of the following fields:

1. “Documentation”, file upload field used to store information in a form of a document.
2. “Gap description”, formatted textual field to describe the gap.
3. “Rationale & related CM function(s)”, entity reference field used to reference a paragraph – data structure in Drupal that contains several fields but can be imagined as a single field. In this case, following fields were used:
 - a. “CM function reference”, entity reference field used to reference the CM functions taxonomy vocabulary.
 - b. “Rationale”, textual field used to describe why certain CM functions are relevant.
4. “Publication Status”, Boolean field used to indicate if content should be published and therefore visible to other site users.
5. “QA Summary”, formatted textual field to write team related comments for individual entity.

To allow re-using of CM gaps in different Trials and independently from the Trials, Trial gaps data model has been split into a “Trial specific” and “Trial independent” part, as illustrated in Figure 4.4.

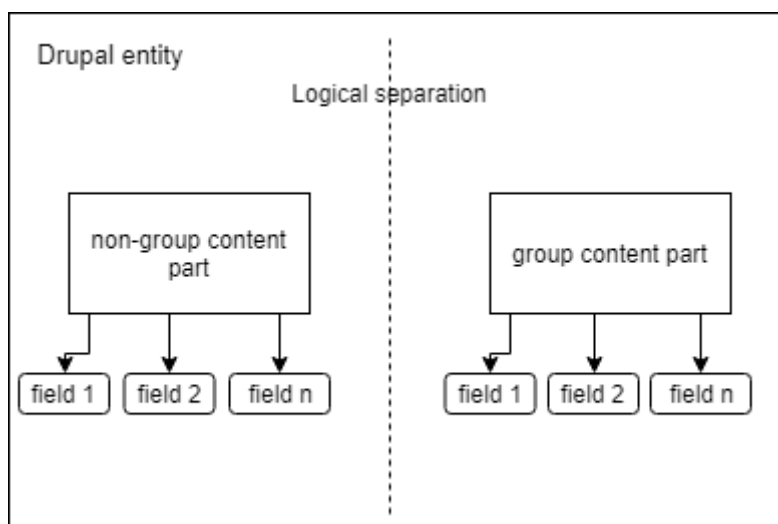


Figure 4.4: Drupal entity

4.4 Trial objective

In order to support Trial organisers in defining objectives for their Trials, a predefined template consisting of the following fields was implemented:

1. "Documentation", file upload field used to store information in a file form.
2. "Trial dimension", a select list field containing predefined values – "Crisis Management", "Solution", "Trial".
3. "Trial objective", formatted textual field to write the objective.
4. "Trial gap reference", entity reference field used to establish relationship to previously defined gaps.
5. "Publication Status", Boolean field used to indicate if content should be published and therefore visible to other site users.
6. "QA Summary", formatted textual field to write team related comments for individual entity.

Fields 4, 5 and 6 belong to the group content part of the entity and therefore are logically separated. It is important to mention, that since they are logically separated as illustrated in Figure 4.5, individual form and display modes had to be defined. Similarly, all other group content entities described in this section have them individually defined, but the implementation is not different to non-group content entities implementation, therefore it will not be explained in detail.

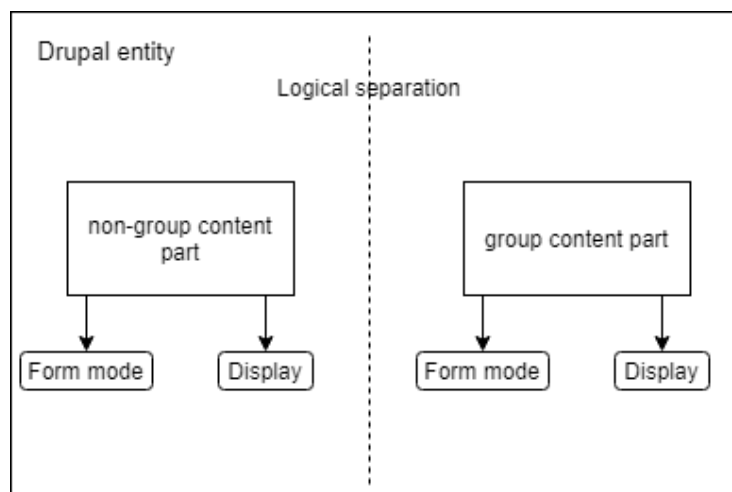


Figure 4.5: Individual form and display for group and non-group entities

Figure 4.6 shows the form mode for the Trial objectives entity, which describes how the fields are displayed to the user in create and edit forms.

FIELD	WIDGET	
+ Language	Language select ▼	
+ Content	Fieldset ▼	Extra CSS classes: help Mark as required
+ Title	Textfield ▼	Textfield size: 60
+ Trial dimension	Select list ▼	
+ Trial objective	Text area (multiple rows) ▼	Number of rows: 5
+ Documentation	File ▼	Progress indicator: throbber

Figure 4.6: Trial objective form mode

Figure 4.7 shows the display mode for the Trial objective entity, which enables configuration on who the information is presented to the user.

FIELD	LABEL	FORMATTER	WIDGET
Content			
+ Title	- Hidden - ▼	Default ▼	Link: no Wrapper: h2 Field template: default
+ Trial dimension	Inline ▼	Key ▼	Field template: default
+ Trial objective	- Hidden - ▼	Trimmed ▼	Trimmed limit: 600 characters Field template: default

Figure 4.7: Trial objective display

4.5 Trial research question

In order to support Trial organisers in defining research questions, a predefined template consisting of the following fields was implemented:

1. “Documentation”, file upload field used to store information in a file form.
2. “Research Question”, formatted textual field to write the research question.
3. “Publication Status”, Boolean field used to indicate if content should be published and therefore visible to other site users.
4. “QA Summary”, formatted textual field to write team related comments for individual entity.
5. “Trial objective”, entity reference field, to establish a relationship to previously defined objectives.

Note: fields 3, 4 and 5 that were used to implement this functionality belong to the group content part of the entity.

Figure 4.8 shows the form mode for the Trial research question entity, which describes how the fields are displayed to the user in create and edit forms.

FIELD	WIDGET	
✚ Title	Textfield ▼	Textfield size: 60
✚ Language	Language select ▼	
✚ Research Question	Text area (multiple rows) ▼	Number of rows: 5
✚ Documentation	File ▼	Progress indicator: throbber

Figure 4.8: Trial research question form mode

Figure 4.9 shows the display mode for the Trial research entity, which enables configuration on who the information is presented to the user.

FIELD	LABEL	FORMATTER
Content		
✚ Title	- Hidden - ▼	Default ▼
✚ Research Question	- Hidden - ▼	Default ▼

Figure 4.9: Trial research question display

4.6 Trial data collection plan

In order to support Trial organisers in defining of a data collection plan, a predefined template consisting of following fields has been implemented:

1. "Dimension", entity reference field used to indicate which dimension a certain data collection plan is refereeing to. It was realised as a small taxonomy containing several values ("Crisis management dimension", "Solution dimension", "Trial dimension").
2. "Documentation", file upload field used to store information in a file form.
3. "Evaluation plan", file upload field used to upload provided Excel sheet templates.
4. "Summary", formatted textual field to write a summary of the data collection plan.
5. "Publication Status", Boolean field used to indicate if content should be published and therefore visible to other site users.
6. "QA Summary", formatted textual field to write team related comments for individual entity.
7. "RQ reference", entity reference field, to establish a relationship to previously defined research questions.

Note: fields 5, 6 and 7 that were used to implement this functionality belong to the group part of the content entity.

Figure 4.10 shows the form mode for the Trial research question entity, which describes how the fields are displayed to the user in create and edit forms.

FIELD	WIDGET	
✚ Title	Textfield ▼	Textfield size: 60
✚ Language	Language select ▼	
✚ Dimension	Check boxes/radio buttons ▼	
✚ Summary	Text area (multiple rows) ▼	Number of rows: 5
✚ Evaluation plan	File ▼	Progress indicator: throbber
✚ Documentation	File ▼	Progress indicator: throbber

Figure 4.10: Trial data collection form mode

Figure 4.11 shows the display mode for the Trial research question entity, which enables configuration on how the information is presented to the user.

FIELD	LABEL	FORMATTER	WIDGET
Content			
✚ Title	- Hidden - ▼	Default ▼	Link: no Wrapper: h4 Field template: default
✚ Details		Details ▼	Default state closed
✚ Dimension	- Hidden - ▼	Label ▼	No link Field template: default
✚ Summary	- Hidden - ▼	Default ▼	Field template: default
✚ Evaluation plan	Above ▼	Embedded Google Documents Viewer ▼	Field template: default
✚ Documentation	Above ▼	Embedded Google Documents Viewer ▼	Field template: default

Figure 4.11: Trial data collection display

4.7 Trial evaluation approaches and metrics

In order to support Trial organisers in defining of evaluation approaches and metrics, a predefined template consisting of following fields has been implemented:

1. "Documentation", file upload field used to store information in a file form.
2. "Evaluation & Metrics", formatted textual field to describe evaluation approaches and metrics.
3. "Publication Status", Boolean field used to indicate if content should be published and therefore visible to other site users.

4. “QA Summary”, formatted textual field to write team related comments for individual entity.

Note: fields 3 and 4 that were used to implement this functionality belong to the group part of the content entity.

Figure 4.12 shows the form mode for the Trial evaluation approaches und metrics entity, which describes how the fields are displayed to the user in create and edit forms.

FIELD	WIDGET	
✚ Title	Textfield ▼	Textfield size: 60
✚ Evaluation & Metrics	Text area (multiple rows) ▼	Number of rows: 5
✚ Documentation	File ▼	Progress indicator: throbber

Figure 4.12: Trial evaluation approaches and metrics form mode

Figure 4.13 shows the display mode for the Trial evaluation approaches and metrics entity, which enables configuration on how the information is presented to the user.

FIELD	LABEL	FORMATTER	WIDGET
Content			
✚ Title	- Hidden - ▼	Default ▼	Link: no Wrapper: h4 Field template: default
✚ Details		Details ▼	Default state closed
✚ Evaluation & Metrics	- Hidden - ▼	Trimmed ▼	Trimmed limit: 600 characters Field template: default
✚ Documentation	Above ▼	Generic file ▼	Use description as link text Field template: default

Figure 4.13: Trial evaluation approaches and metrics display

4.8 Trial scenario

In order to support Trial organisers in defining Trial scenarios, a predefined template consisting of following fields has been implemented:

1. “Scenario”, file upload field used to store scenario file.
2. “Summary”, formatted textual field to write a scenario summary.
3. “When”, date range field to indicate the time range of the scenario.
4. “Publication Status”, Boolean field used to indicate if content should be published and therefore visible to other site users.
5. “QA Summary”, formatted textual field to write team related comments for individual entity.

6. “RQ references”, entity reference field, to establish a relationship to previously defined research questions.
7. “Solution references”, entity reference field, to establish a relationship to selected solutions.

Note: fields 4, 5, 6 and 7 that were used to implement this functionality belong to the group part of the content entity.

Figure 4.14 shows the form mode for the Trial scenario entity, which describes how the fields are displayed to the user in create and edit forms.

FIELD	WIDGET	
⊕ Title	Textfield ▼	Textfield size: 60
⊕ Language	Language select ▼	
⊕ Summary	Text area (multiple rows) ▼	Number of rows: 5
⊕ Scenario	File ▼	Progress indicator: throbber
⊕ When	Date and time range ▼	

Figure 4.14: Trial scenario form mode

Figure 4.15 shows the display mode for the Trial scenario entity, which enables configuration on who the information is presented to the user.

FIELD	LABEL	FORMATTER	WIDGET
Content			
⊕ Title	- Hidden - ▼	Default ▼	Link: no Wrapper: h4 Field template: default
⊕ Details		Details ▼	Default state closed
⊕ Summary	- Hidden - ▼	Default ▼	Field template: default
⊕ Scenario	Inline ▼	Generic file ▼	Use description as link text Field template: default
⊕ When	Inline ▼	Default ▼	Format: Fri, 06/28/2019 – 11:07 Separator: – Field template: default

Figure 4.15: Trial scenario display

4.9 Solution selection

In order to support Trial organisers in describing Trial solutions, a predefined template consisting of following fields has been implemented:

1. “Documentation”, file upload field used to store information in a file form.
2. “Solution reference”, entity reference field to establish a relationship to the solution entity described in section 3.1. Note, this relationship can only be established if both prototypes are implemented.
3. “Summary” formatted textual field to write a summary of the solution.
4. “Publication Status”, Boolean field used to indicate if content should be published and therefore visible to other site users.
5. “QA Summary”, formatted textual field to write team related comments for individual entity.

Note: fields 4 and 5 that were used to implement this functionality belong to the group part of the content entity.

Figure 4.16 shows the form mode for the Trial solution entity, which describes how the fields are displayed to the user in create and edit forms. As presented, form mode for this entity differs to other, previously described one. The main reason for this is, that in order to provide additional help to Trial organisers in selection solutions, the TGT prototype was extended with additional functionality. Based on the overlap between CM functions mentioned in the Trial gap entity, with the ones that are mentioned in the solution use case entity (section 3.3), a list of possible solutions is offered. In order to make this available at create and edit forms, a contributed “Entity browser” module was implemented, configured and adjusted to allow a user-friendly solution selection with a possibility to show only those solutions that are recommended by the prototype, as well as to make it possible to select other solutions from the list of all available ones. In order to have this functionality, both prototypes must be available. Figure 4.17 shows an example of how proposed solutions are presented to the user in this step.

FIELD	WIDGET	
⊕ Tabs	Tabs ▼	Direction: vertical
⊕ Trial-specific information	Tab ▼	Tab: closed
⊕ Acronym	Fieldset ▼	Mark as required
⊕ Title	Textfield ▼	Textfield size: 60
⊕ Summary	Text area (multiple rows) ▼	Number of rows: 5
⊕ Documentation	File ▼	Progress indicator: throbber
⊕ Solution in PoS	Tab ▼	Tab: closed
⊕ Solution reference:	Entity browser ▼	Entity browser: Solutions Selection mode: Append to selection Entity display: Entity label
⊕ Language	Language select ▼	

Figure 4.16: Trial solution form display

Suggested solutions (based on chosen gaps)

Relevance	Title	Innovation stage	Gap
6	CrowdTasker	Stage 4: Early Adoption/ Distribution	Managing spontaneous volunteers
3	Scenario enabled Psychological First Aid (PFA) training	Stage 5: Market Growth	Addressing the psychological stress of volunteers
1	Emergency Mapping Tool (EMT)	Stage 4: Early Adoption/ Distribution	Managing spontaneous volunteers
1	Social Media Analysis Platform (SMAP)	Stage 3: Initial Piloting	Managing spontaneous volunteers
1	zirkarta	Stage 3: Initial Piloting	Managing spontaneous volunteers
1	CrowdTasker	Stage 4: Early Adoption/ Distribution	Addressing the psychological stress of volunteers

Figure 4.17: Example of suggested solutions

Figure 4.18 shows the display mode for the Trial solution entity, which enables configuration on who the information is presented to the user. The “Solution reference” field that establishes a relationship to the solution entity was configured to be presented as a rendered entity, what is used in Drupal to display an entire entity with all of its fields. Similar to display options presented, a rendered entity display can also be individually configured to show the selected fields in a certain way.

FIELD	LABEL	FORMATTER	WIDGET
Content			
⊕ Title	- Hidden -	Default	Link: no Wrapper: h4 Field template: default
⊕ Details		Details	Default state closed
⊕ Solution reference:	- Hidden -	Rendered entity	Rendered as Teaser Field template: default
⊕ Summary	- Hidden -	Default	Field template: default
⊕ Documentation	Inline	Generic file	Use description as link text Field template: default

Figure 4.18: Trial solution display

4.10 Test case

In order to provide a possibility to describe test cases, where it is possible to indicate which solutions can work together (compatible to exchange information), the TGT prototype implements a predefined template consisting of the following fields:

1. “Attachment”, file upload field to store additional information related to the test case in a file form.
2. “Criteria for success”, formatted textual field used to store information as indicated by the label.
3. “Expected results”, formatted textual field to store information as indicated by the label.
4. “Illustration”, image upload field to store test case illustrations.
5. “Objective”, formatted textual field to store information as indicated by the label.
6. “Preconditions/Test data”, formatted textual field to store information as indicated by the label.
7. “Sequence of actions”, formatted textual field to store information as indicated by the label.
8. “Publication Status”, Boolean field used to indicate if content should be published and therefore visible to other site users.
9. “QA Summary”, formatted textual field.

10. "Results", entity reference revisions field used to add a paragraph that stores possible values for this field (Okay, Not Okay, Partially Ok), together with a summary, textual field to describe the results.
11. "Test case solution(s)", entity reference field used to establish a relationship to Trial solutions entity described in section 4.9.

Note: fields 8, 9, 10 and 11 that were used to implement this functionality belong to the group part of the content entity.

Figure 4.19 shows the form mode for the Trial test case entity, which describes how the fields are displayed to the user in create and edit forms.

In addition, all previously described test cases that are made public can be re-used with a different set of solutions and results.

FIELD	WIDGET	
✚ Title	Textfield ▼	Textfield size: 60
✚ Objective	Text area (multiple rows) ▼	Number of rows: 5
✚ Language	Language select ▼	
✚ Sequence of actions	Text area (multiple rows) ▼	Number of rows: 5
✚ Preconditions /Test data	Text area (multiple rows) ▼	Number of rows: 5
✚ Criteria for success	Text area (multiple rows) ▼	Number of rows: 5
✚ Expected results	Text area (multiple rows) ▼	Number of rows: 5
✚ Illustration	Image ▼	Preview image style: Thumbnail (100×100) Progress indicator: throbber
✚ Attachment	File ▼	Progress indicator: throbber

Figure 4.19: Test case form display

Figure 4.20 shows the display mode for the Trial test case entity, which enables configuration on who the information is presented to the user.

FIELD	LABEL	FORMATTER	WIDGET
Content			
✚ Title	- Hidden - ▼	Default ▼	Link: no Wrapper: h2 Field template: default
✚ Details		Details ▼	Default state closed
✚ Objective	Above ▼	Default ▼	Field template: default
✚ Sequence of actions	Above ▼	Default ▼	Field template: default
✚ Preconditions/Test data	Above ▼	Default ▼	Field template: default
✚ Criteria for success	Above ▼	Default ▼	Field template: default
✚ Expected results	Above ▼	Default ▼	Field template: default
✚ Illustration	Above ▼	Image ▼	Original image Field template: default
✚ Attachment	Above ▼	Generic file ▼	Use description as link text Field template: default

Figure 4.20: Test case display

4.11 Execution phase

In order to provide additional support to Trial organisers and practitioners in the correct implementation of the Trial Guidance Methodology, the TGT prototype implements individual checklists for each of the steps in Trial phases as they are defined in the **TGM handbook** (10). As mentioned in section 4.1, several Boolean fields have been added to the Trial data model and by using the contributed “Form mode manager” and “Display suite” modules they have been formed into checklist for the user. After the user has saved the Trial group template, the TGT guides him/her through individual steps. As soon as the user visits a certain step, he/she is presented with the checklist with the possibility to edit it at any time, as shown in Figure 4.21. Its purpose is to remind the user which things should be considered when defining a certain step.

Key-events of each gap clearly stated: ✕

Triggering conditions and injects per keyevent identified and written down: ✕

Roles and actions needed for key-event identified: ✕

Key-events combined with a conclusive storyline: ✕

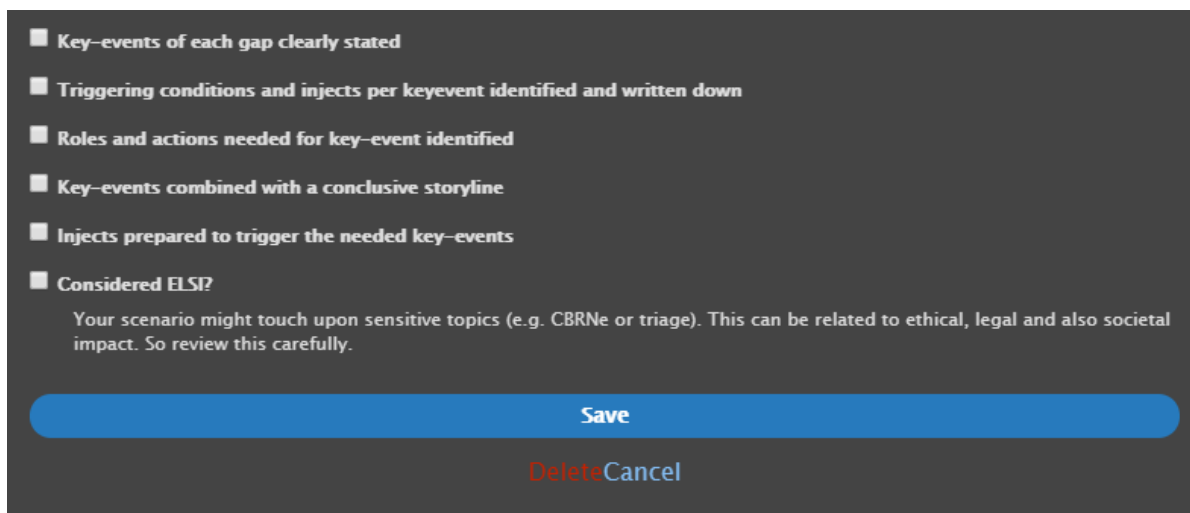
Injects prepared to trigger the needed key-events: ✕

Considered ELSI?: ✕

Edit checklist

Figure 4.21: Checklist example

Figure 4.22 illustrates such a checklist as it is shown to the user, which was implemented with the help of an already available “of_canvas” class that was released with the version 8.5.0 of Drupal. This class offers visual improvements for displaying elements.



- Key-events of each gap clearly stated
- Triggering conditions and injects per keyevent identified and written down
- Roles and actions needed for key-event identified
- Key-events combined with a conclusive storyline
- Injects prepared to trigger the needed key-events
- Considered ELSI?

Your scenario might touch upon sensitive topics (e.g. CBRNe or triage). This can be related to ethical, legal and also societal impact. So review this carefully.

Save

Delete Cancel

Figure 4.22: Checklist edit form example

As mentioned in sections 4.13 and 4.14 of the **D933.11** (2), no implementable requirements have been defined for this functionality; therefore, the TGT prototype offers support in a form of textual help and of a checklist. Nevertheless, checklists are not limited to this functionality, nor offered as a substitute for missing supportive functionalities, but are implemented as an additional way of supporting the TGT prototype users.

4.12 Evaluation phase

This functionality of the TGT is implemented similarly to the functionality described in section 4.11.

4.13 Trial search and matching

The TGT prototype implements several search and matching functionalities:

1. Search for previously defined gaps (as described in section 4.3).
2. Matching of solutions with gaps based on CM functions that are addressed (as described in section 4.9).
3. Matching of examples of previous Trials with individual functionality (contextual help, as described in section 2.7).
4. Matching of SLR content with related functionality (as described in section 4.15).

4.14 Trial exports

The TGT prototype implements PDF export functionality as described in section 2.9.

4.15 Knowledge Database

The TGT prototype implements a searchable knowledge database functionality, that gives the possibility to search through results of a systematic literature research (SLR) that took place in the project. Results are related to individual steps that are described in sections 4.3 to 4.9 and can be directly accessed from there using the contextual help functionality described in section 2.7. Figure 4.23 illustrates how this information is presented to the user.

SLR Criteria

Experiment planning and deviations

Search

Items per page

10

Apply

Publication

A 3-year Health Care Coalition Experience in Advancing Hospital Evacuation Preparedness.

A container multimodal transportation scheduling approach based on immune affinity model for emergency relief

Findings

an planning team was established to develop the training and exercise plan as well as set overarching training and exercise program objectivesThe multi-year effort utilized a variety workshops, seminars, webinars, tabletops, functional exercises, and culminated with a full-scale exercise testing hospital evacuation

Because the model is a multi-objective model, it can be transferred into single objective the following strategy: aggregating the two objectives by weight.(For emergency relief: time objective is the most important (weighed with 80%)

Figure 4.23: SLR content page

Additionally, a designated “lessons learned” template has been implemented, in order to provide a possibility to document lessons learned during Trials, a predefined template with the following fields was implemented:

- 1. “Documentation”, file upload field used to store related information in file form.
- 2. “Lessons learned”, formatted textual field used to store lessons learned.

FIELD	WIDGET	
⊕ Title	Textfield	Textfield size: 60
⊕ Lessons learned	Text area (multiple rows)	Number of rows: 5
⊕ Documentation	File	Progress indicator: throbber

Figure 4.24: Lessons learned form mode

FIELD	LABEL	FORMATTER
Content		
⊕ Title	- Hidden -	Default
⊕ Details		Details
⊕ Lessons learned	- Hidden -	Default
⊕ Documentation	Above	Embedded Google Documents Viewer

Figure 4.25: Lessons learned display

The main idea behind this entity is to provide possibility and to encourage knowledge sharing. All information saved in it can be displayed on the corresponding part of the site and reused in various occasions, but at the time this deliverable was written, no “lessons learned” information related to individual steps was provided by end-users.

4.16 Group-level Trial validation

In order to provide automatic support and feedback to Trial organisers and practitioners, the TGT prototype implements a custom validation function. The main goal of it is to ensure that all the steps of the TGM are correctly followed with help of the tool. This function checks the following conditions:

1. For each of the steps at least one entity is created, and it is published.
2. Each entity is referenced by another entity as it is foreseen by the TGM. For example, each Trial gap needs to be referenced by at least one Trial objective.
3. Each entity is referencing another entity as it is foreseen by the TGM. For example, each Trial objective references at least one Trial gap.
4. User is informed about all unpublished entities that he has, but this is not a K.O. criteria, if at least one entity of the same type is published, the validation will be positive.

The screenshot shows the DRIVER+ Trial 3 - AUSTRIA interface. On the left, there is a sidebar with a navigation menu. The main content area displays a validation result. At the top, there is a header bar with buttons: Validate, Group dashboard, Members, View, Edit, Delete, Manage display, All entities, Devel, and Translate. Below the header, the parent group is identified as DRIVER+ Trial 3 - AUSTRIA. The validation results section is highlighted in green and contains the following text:

✓ **The Trial is ready for publication request now**

WARNING: There is unpublished content

- [Incorporating information from multiple and non-traditional sources](#) (Trial Objectives)
- [Outline of the Trial scenario/aim](#) (Trial Scenario)

The sidebar on the left includes sections for STEP 0 (Trial gaps), PREPARATION PHASE (Trial Objectives, Research Questions, Data Collection Plan, Evaluation Approaches&Metrics, Scenario formulation, Solution selection), and EXECUTION PHASE (Trial integration meeting, Dry run 1).

Figure 4.26: Positive validation result

The screenshot shows the DRIVER+ Trial 3 - AUSTRIA interface with a negative validation result. The main content area is highlighted in red and contains the following text:

✗ **The trial has unmet requirements for publication:**

Missing Trial content

- please define and publish at least one Research Question
- please define and publish at least one Data Collection
- please define and publish at least one Trial Scenario

CRITICAL: The following group content is not referenced by any other content

- [Addressing the psychological stress of volunteers - S](#)
- [Communicating with the public during a large crisis \(Note: in the CfA addressed by "Interaction with populations"\)](#)

CRITICAL: The following group content is not referencing to any other content

- [Trial setup](#) (Trial Objectives)

WARNING: There is unpublished content

- [Real-time data and information fusion to support incident commander decision making](#) (Trial Objectives)
- [Volunteer Management](#) (Trial Gaps)
- [Volunteer coordination tool](#) (Trial Solution)

The sidebar on the left is identical to the previous figure, showing the navigation menu for STEP 0, PREPARATION PHASE, and EXECUTION PHASE.

Figure 4.27: Negative validation result

4.17 Trial landing pages

After all entities were implemented as described in sections 4.3 to 4.10, the following implementation step was to implement pages where information stored about a Trial is going to be presented. In order to do so, again “Views”, a characteristic Drupal functionality was utilised. This allowed easy customisable visualisation of desired fields from the TGT prototype’s database. An example of such view is shown in the Figure 4.28. Since the TGT prototype is more complex with regards to the number of fields to be used and cross-relationships between them, more such views have been implemented.

The screenshot shows the Drupal Views configuration interface for a view named 'Trial step page'. The interface is divided into several sections:

- Display name:** Trial step page
- TITLE:** Title: `{% raw_arguments type %}`
- FORMAT:** Unformatted list | Settings
- SHOW:** Fields | Settings
- FIELDS:** A list of fields to be displayed, including:
 - (Group content Content) Content: ID
 - (Group content Content) Edit content link [hidden]
 - (Group content Content) Delete content link (Delete globally) [hidden]
 - View status and relations link [hidden]
 - Edit status and relations link [hidden]
 - Delete (unlink) from trial [hidden]
 - Dropbutton: edit content [hidden]
 - Dropbutton: edit status & references [hidden]
 - (Group content Content) Rendered content (compact) [hidden]
 - (Group content Content) New or updated? [hidden]
 - (Group content Content) Last changed (when and by whom) [hidden]
 - Rendered status & references (QA) [hidden]
 - (Group content Content) Rendered check list [hidden]
 - Approved or not - used to add class to each row [hidden]
 - Render all
 - (Group content Content) view link [hidden]
 - (Group content Content) Content: Title [hidden]
- FILTER CRITERIA:** (Group content Content) Content: Translation language (= Site's default language (English))
- SORT CRITERIA:**
- PAGE SETTINGS:**
 - Path: `{group}/{group}/{type}`
 - Menu: No menu
 - Access: Group permission | View group
 - HEADER:** Global: Unfiltered text (Global: Unfiltered text)
 - FOOTER:**
 - NO RESULTS BEHAVIOR:** Global: View area (Global: View area)
 - PAGER:** Use pager: Display all items | All items
 - More link: No
 - LANGUAGE:** Rendering Language: Interface text language selected for page
- ADVANCED:**
 - CONTEXTUAL FILTERS:** Group content: Parent group
 - RELATIONSHIPS:** Group content: Content: Content type
 - EXPOSED FORM:** Exposed form in block: No; Exposed form style: Basic | Settings
 - OTHER:** Machine Name: trial_objectives_page; Administrative comment: None; Use AJAX: No; Hide attachments in summary: No; Contextual links: Shown; Use aggregation: No; Query settings: Settings; Caching: Tag based; CSS class: None

Figure 4.28: Example of a view used in TGT

5. Conclusions and the way forward

PoS and TGT development is considered successful with all the features that have been specified in **D933.11** (2) and with 114 out of 128 (89%) of the requirements being implemented. The resulting prototypes are considered feature-complete, with remaining 14 requirements being either obsolete or of a low priority. However, additional efforts should be put into making the prototypes more user friendly and easier to maintain after the project end.

For TGT, the implementation mainly covers the preparation, whereas the checklists are provided for execution phase. For PoS, the Gaps play a more prominent role than initially planned, with a Gaps search/overview page and “solutions addressing this gap” implemented.

According to DRIVER+ Description of Work, the final project months will be dedicated to **T933.4** “industrialization” of the PoS and TGT, that is:

- Consolidation of the already implemented features.
- Improving of the site usability.
- Improving of the documentation.
- Assuring that the site can be easily re-deployed at new address e.g. as a starting point for developing a similar service, or as a private tool for specific groups of users.

Concerning the “re-deploying” point, we are currently evaluating two options for achieving this goal:

1. Create a PoS/TGT Drupal “distribution”: Distributions are full copies of Drupal that include Drupal Core, along with additional software such as themes, modules, libraries, and installation profiles (source: <https://www.drupal.org/docs/8/distributions>).

The idea behind this approach is to develop a DRIVER+ PoS/TGT Drupal 8 distribution package that enables any stakeholder to install their own PoS site using the Drupal wizard installer.

2. Using Git: Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and effective (source: <https://git-scm.com/>).

The idea of this approach is to use GitHub project repository <https://github.com/DRIVER-EU/> to store all the PoS code. It is necessary to provide instructions that explain how to install a PoS site from scratch.

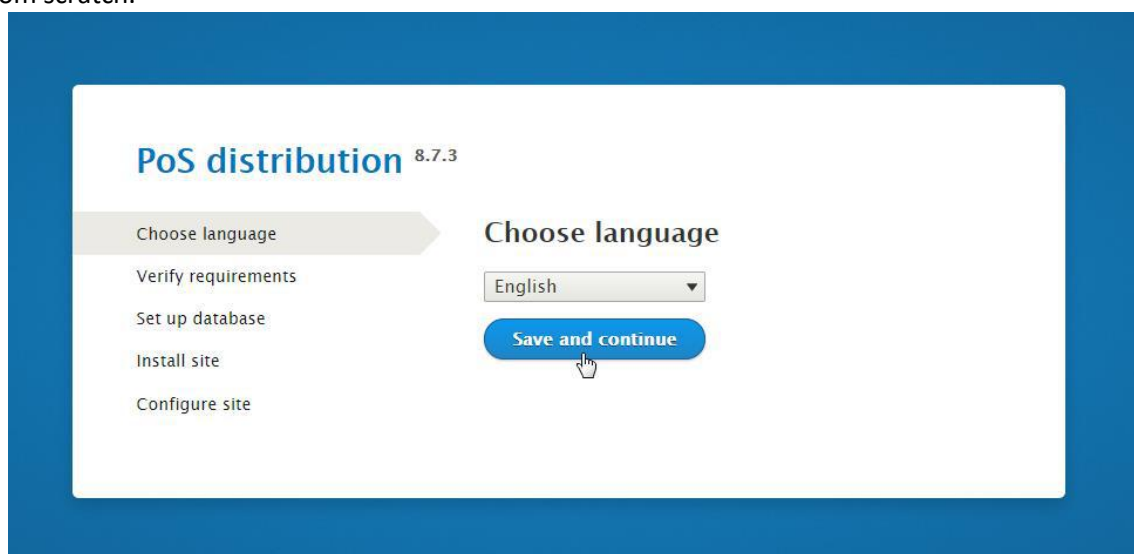


Figure 5.1: Distribution wizard installer

If this approach is selected a distribution package will be generated and uploaded to the GitHub project repository, where stakeholders can download it to install their own PoS site in their own environments.

The evaluation process will be continued, and the best option will be selected. Work done shall be described in **T933.4 DRIVER+ tools industrialization** and the results presented in **D933.41**.

References

1. **DRIVER+ project.** *D932.11 - Functional design of the PoS database.* 2018.
2. **DRIVER+ project.** *D933.11 - Online tools - Implementation specifications.* 2019.
3. **DRIVER+ project.** DRIVER+ Grant Agreement, Annex I Description of work. 2018. 4-th amendment.
4. **DRIVER+ project.** *D922.21- Trial guidance methodology and guidance tool specifications.* 2018.
5. **DRIVER+ project.** *D932.12 - PoS Tutorials and recommendations.* 2019.
6. **DRIVER+ project.** *D922.41 - Trial guidance methodology and guidance tool specifications.* 2018.
7. **DRIVER+ project.** *D934.10 - Taxonomy of CM functions for classification of solutions.* 2017.
8. **DRIVER+ project.** DRIVER+ terminology.
9. **DRIVER+ project.** *D922.11 - List of CM gaps.* 2018.
10. **DRIVER+ project.** *Trial Guidance Methodology Handbook.* s.l. : DRIVER+, 2019. D922.42.
11. **DRIVER+ project.** *D922.11 - List of CM gaps.* 2018.

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 updated⁴. 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
Crisis Management	Holistic management process that identifies potential impacts that threaten an organization and provides a framework for building resilience, with the capability for an effective response that safeguards the interests of the organization's key interested parties, reputation, brand and value-creating activities, as well as effectively restoring operational capabilities. Note 1 to entry: Crisis management also involves the management of preparedness, mitigation response, and continuity or recovery in the event of an incident, as well as management of the overall programme through training, rehearsals and reviews to ensure the preparedness, response and continuity plans stay current and up-to-date.	ISO 22300:2018(en) Security and resilience — Vocabulary. Link: https://www.iso.org/obp/ui/#iso:std:iso:22300:ed-2:v1:en:term:3.60 .
Crisis Management Taxonomy	A taxonomy of Crisis Management Functions describing strategically-directed activities to prevent, prepare, respond to and mitigate the effects of and recover from a crisis. Note 1 to entry: Taxonomy is a scheme of categories and subcategories that can be used to sort and otherwise organize itemized knowledge or information that are processed, organized and correlated to produce meaning.	ISO 5127:2017(en) Information and documentation — Foundation and vocabulary. Link: https://www.iso.org/obp/ui/#iso:std:iso:5127:ed-2:v1:en:term:3.8.6.07 .
Gap	Gaps between the existing capabilities of responders and what was actually needed for effective and timely response.	Project Responder 5, Homeland Security, Science and Technology, August 2017. Link: https://www.dhs.gov/sites/default/files/publications/Project-Responder-5-Report_170814-508.pdf .

⁴ The Portfolio of Solutions and the terminology of the DRIVER+ project are accessible on the DRIVER+ public website (<https://www.driver-project.eu/>). Further information can be received by contacting coordination@projectdriver.eu.

Terminology	Definition	Source
Lessons Learnt	Lessons learning: process of distributing the problem information to the whole project and organization as well as other related projects and organizations, warning if similar failure modes or mechanism issues exist and taking preventive actions.	ISO 18238:2015(en) Space systems — Closed loop problem solving management, 3.3.
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, as well as the stimuli used to achieve exercise project performance objectives. DRIVER+ note 1: In the context of DRIVER+ scenarios are defined for Trials not for exercises.	ISO 22300:2018(en) Security and resilience — Vocabulary. Link: https://www.iso.org/obp/ui/#iso:std:iso:22300:ed-2:v1:en:term:3.217 .
Solution	A solution is a means that contributes to a crisis management function. A solution is either one or more processes or one or more tools with related procedures.	Initial DRIVER+ definition.
Trial	An event for systematically assessing solutions for current and emerging needs in such a way that practitioners can do this following a pragmatic and systematic approach.	Initial DRIVER+ definition.
Trial Guidance Methodology (TGM)	A structured approach from designing a Trial to evaluating the outcomes and identifying lessons learnt.	Initial DRIVER+ definition.
Trial Guidance Tool (TGT)	A software tool that guides Trial design, execution and evaluation in a step-by-step way (according to the Trial Guidance Methodology) including as much of the necessary information as possible in form of data or references to the Portfolio of Solutions.	Initial DRIVER+ definition.

Annex 2 – Requirements implementation status

This table is complementary to Table 2.1 , Table 3.1 Table 4.1 and show the implementation status of individual requirements for each of the functionalities that the PoS and the TGT prototypes have.

Description of implementation status values:

- *Implemented* – the requirement has been fully implemented and the described functionality is available on the prototypes.
- *in course of implementation* – the requirement has been partially implemented and may not be available on the public version of the PoS and TGT prototypes.
- *stalled* – the requirement has not been implemented for a specific reason which is stated in the comment field.

Table A2: Implementation status

ID	Requirement	Implementation status	Comment
TGT-01	The TGT is used by Trial Committees in general and is not restricted to the project.	Implemented	
TGT-02	The TGT has a procedure for assigning accounts.	Implemented	
TGT-03	The TGT is web-based.	Implemented	
TGT-04	The TGT mainly supports the preparation phase of the Trials.	Implemented	Checklists and help texts have been implemented for the execution phase.
TGT-05	The TGT provides help functionality.	Implemented	Static help texts are gradually replaced by interactive content based on https://h5p.org .
TGT-06	The TGT provides checklists for each step and has validation criteria to ensure correctness.	Implemented	
TGT-07	The TGT provides links to the TGM Handbook.	Implemented	
TGT-08	The TGT contains a repository of examples.	Implemented	
TGT-09	The TGT implements search and filter function for examples.	Implemented	
TGT-10	The TGT validates the Trial definition.	Implemented	
TGT-11	The TGT supports different types of users.	Implemented	
TGT-12	The TGT implements a three-layer quality assurance.	Implemented	
TGT-13	The TGT provides e-mail notifications for Trial members to inform them of changes.	Implemented	

ID	Requirement	Implementation status	Comment
TGT-14	The TGT provides support in describing other types of Trial-like experiments.	Implemented	Decision was made by the Trial guidance committee to not differentiate between “real Trials” and “Trial like events/experiments”, so the same template is used to describe all Trial-like events where experiences with solutions were gathered.
TGT-15	The TGT allows Test-case descriptions.	Implemented	
TGT-16	The TGT provides live chat functionality.	Implemented	Initially implemented, but this function has been dropped due to performance issues.
TGT-17	The TGT provides a link to contact the TGM experts.	Stalled	TGM experts have not been identified (CoE). It is already possible to contact the helpdesk at any time and adding a “expert question” type of request would be trivial.
TGT-18	Access to the TGT for authorized users only.	Implemented	
TGT-19	Authorized users can add or modify Trials in the TGT.	Implemented	
TGT-20	Trials can be exported (xml/json format).	Implemented	
TGT-21	The TGT supports the iterative six-step approach.	Implemented	
TGT-22	The TGT implements a relation between six step components (in both directions).	Implemented	
TGT-23	The output of the TGT may be directly imported into section 2 of the Trial Action Plan (TAP).	Implemented	
TGT-24	The TGT extracts information from the Portfolio of Solutions (PoS).	Implemented	
TGT-25	The validated DRIVER+ CM gaps are input to the TGT.	Implemented	Validated DRIVER+ CM gaps are included and marked as “strategic”; new gaps can be promoted to “strategic” status as needed.
TGT-26	The TGT provides a possibility to define new Trial gaps.	Implemented	Gaps can be defined within a Trial but also outside of the Trial.

ID	Requirement	Implementation status	Comment
TGT-27	For each Trial, at least one gap must be selected.	Implemented	
TGT-28	Allow interaction between different users with the Trial Committee.	Implemented	
TGT-29	Trial objectives are linked to at least one CM gap and each CM gap is related to a CM function.	Implemented	
TGT-30	The TGT provides a template to facilitate the formulation of the Trial objectives in a manner that is SMART.	Implemented	
TGT-31	Each objective is categorized as either “Crisis Management objective”, “solution objective” or “Trial objective”.	Implemented	
TGT-32	The TGT provides a list of identified Trial objectives in the Trial.	Implemented	
TGT-33	Examples of Trial objectives used in other Trials are provided, supported by a search filter.	Implemented	
TGT-34	Include metrics with Trial objectives.	Implemented	
TGT-35	A research question relates to a Trial objective.	Implemented	
TGT-36	The TGT provides a template for the research question dealing with crisis management task, process, content, crisis management roles and the solution required.	Implemented	
TGT-37	Examples of research methods are provided from the DRIVER+ knowledge base, including lessons learnt.	Implemented	
TGT-38	The TGT offers a list of possible methods for data collection.	Implemented	
TGT-39	The TGT offers Excel-file templates for users to download.	Implemented	
TGT-40	Every metric is linked to at least one assessment method.	Stalled	This requirement is obsolete, since data collection plan was redesigned.
TGT-41	Examples of research methods with associated data collection plans are provided from the DRIVER+ knowledge base.	Stalled	This requirement is obsolete, since data collection plan was redesigned.

ID	Requirement	Implementation status	Comment
TGT-42	Provide a description of different data collection and analysis techniques.	Implemented	
TGT-43	Provide a checklist (for the data collection plan).	Implemented	
TGT-44	Relate metrics to the Observer Support Tool which is a component of the reference implementation of the Test-bed.	Stalled	Observer Tool Interface/specification is missing, so this requirement is considered obsolete.
TGT-45	Examples of data analysis techniques and metrics from previous Trials are derived from the DRIVER+ knowledge base.	Implemented	
TGT-46	Examples of evaluation approaches applied in previous Trials.	Implemented	
TGT-47	Provide explanation on evaluation approaches, distinguishing between literature and practice (past Trials).	Stalled	Examples from the previous Trials are missing.
TGT-48	Examples for data techniques to measure/observe metrics in a Trial.	Stalled	Examples from the previous Trials are missing.
TGT-49	Scenario text can be entered by uploading a text file.	Implemented	Scenario description consists of a (preferably) short summary text and one or more attachments that can be prepared offline and are merely uploaded to the platform as additional documentation.
TGT-50	Scenario text can be edited.	Implemented	Scenario summary text can be edited online; attached documents offline.
TGT-51	Solutions are related to one or more CM functions.	Implemented	Solutions relate to CM functions through “solution Use Cases”.
TGT-52	The TGT supports the DRIVER+ CM function taxonomy.	Implemented	Full CM taxonomy is available on the PoS/TGT and the functions are provided to allow linking of CM Gaps and solution Use Cases with the CM functions.
TGT-53	The TGT supports searching the PoS for possible solutions for the objectives formulated, using filter options.	Implemented	Fully fledged search combining the deep full text search with faceted search has been implemented for solutions.
TGT-54	The TGT offers a list of possible Solutions based on Trial gaps.	Implemented	Gaps are automatically linked with (potential) solutions, based on the shared CM functions.

ID	Requirement	Implementation status	Comment
TGT-55	Selected solutions are presented in the TGT for review, including all information relevant.	Implemented	
TGT-56	Solutions can be included / excluded into the Trial by the user.	Implemented	
PoS-01	Access right	Implemented	
PoS-02	Tagging of content with Taxonomy	Implemented	
PoS-03	Shared terminology	Implemented	
PoS-04	Notifications mechanism (e.g. per e-mail)	Implemented	
PoS-05	Different views at the same data	Implemented	
PoS-06	PoS contents validation	Implemented	
PoS-07	Search by explicitly linked content	Implemented	
PoS-08	Search for implicitly associated content	Implemented	
PoS-09	Tasking support	Implemented	
PoS-10	Cloning of data	Stalled	Cloning of Groups and group data turned out to be more difficult than anticipated.
PoS-11	Actions and decisions	Stalled	This requirement is obsolete.
PoS-12	Search by keywords	Implemented	
PoS-13	Search by data type	Implemented	
PoS-14	Search by taxonomy tags	Implemented	
PoS-15	Dedicated data types for Trials, CM Solutions, CM Tools, CM Functions, CM Gaps, Solution Capabilities and Trial Needs/Requirements	Implemented	
PoS-16	Select solutions to be used in a Trial	Implemented	
PoS-17	Describe Test-bed	Stalled	Testbed specification is missing.
PoS-18	Describe Solution	Implemented	
PoS-19	Define Mapping and linking relations between these data types	Implemented	
PoS-20	Describe Tool	Stalled	“Tools” aren’t used as a separate entity in DRIVER+ or in PoS anymore, just solutions so this requirement is obsolete.
PoS-21	Describe Trial	Implemented	

ID	Requirement	Implementation status	Comment
PoS-22	Describe Trial needs and requirements (User Stories)	Implemented	Trial needs and requirements are implicit part of the Trial description (gaps, objectives, RQs) and no explicit Trial Use Cases are defined.
PoS-23	Describe Solution Capabilities (Test Cases)	Implemented	Solution capabilities are defined as solution Use Cases in the PoS. Test cases are used within the Trial to define technical test that a solution must pass in order to be used in this Trial.
PoS-24	Describe the level of integration of a Tool or Solution in the Test-bed.	Stalled	Specification is missing.
PoS-25	Help Trial stakeholders to pre-select the solutions for use in a Trial	Implemented	
PoS-26	Help the solution providers to apply for the Trials	Stalled	A possibility to apply for the trials online was implemented but declared obsolete later.
PoS-27	Validate tool integration	Implemented	Validating the solution integration is done through solution Test Cases in Trials.
PoS-28	Mechanism for maintaining the terminology and taxonomies that are used in the PoS DB	Implemented	Taxonomies can be maintained using the standard Drupal administration tools. Vocabulary is also implemented as a Drupal taxonomy.
PoS-29	Validate test cases	Stalled	Technically possible, but this is a task for the Trial team and solution owners, therefore out of scope.
PoS-30	Indicate the results of the trialing for a specific solution	Implemented	Trial can indicate that the solution was used and provide results.
PoS-31	Search for CM Solutions or CM Tools by CM Functions	Implemented	
PoS-32	Search solution or tools by trials	Implemented	
PoS-33	Recommendations system	Implemented	
PoS-34	Website is user friendly	Implemented	
PoS-35	PoS links to the project	Implemented	
PoS-36	Solution providers can advertise own website	Implemented	
PoS-37	Solutions address taxonomy terms	Implemented	

ID	Requirement	Implementation status	Comment
PoS-38	PoS lists relevant CM gaps	Implemented	
PoS-39	PoS/TGT allow additional uploads	Implemented	
PoS-40	Solution creation is intuitive	Implemented	
PoS-41	PoS describes the difference between tools and Solutions	Implemented	
PoS-42	Website provides help text and a contact form	Implemented	
PoS-43	Website implements search functionality	Implemented	
PoS-44	Website accommodates a searchable knowledge database	Implemented	
PoS-45	The site's design should be appealing	Implemented	
PoS-46	User management and authentication	Implemented	
PoS-47	Website offers a pdf export function	Implemented	
PoS-48	PoS site links terminology terms with their definitions	Implemented	
W-01	The website is multi-lingual	Implemented	While the technical implementation exists that allows translating of Trials and solutions, it is unclear if, who and when will translate the contents.
W-02	The website supports collaborative working	Implemented	
W-03	The website provides tutorials	Implemented	PDF tutorials are gradually being replaced by the interactive ones, based on https://h5p.org
W-04	Credentials are shared with community	Implemented	It is possible to login with LinkedIn credentials.
W-05	Website users can manage own team	Implemented	
W-06	Real names are shown on the site, e-mail or username is used for login	Implemented	
W-07	Website users can access different information based on their role	Implemented	
W-08	Website offers a help desk	Implemented	
W-09	PoS site has definitions from different sources	Implemented	
W-10	PoS site calculates similarity between different definitions	Stalled	Calculating the similarity of the glossary definitions is considered beyond the project scope.

ID	Requirement	Implementation status	Comment
W-11	PoS site has feedback section for Solutions	Implemented	
W-12	PoS supports description of use cases	Implemented	
W-13	Use cases are based on CM functions taxonomy	Implemented	
W-14	PoS offers a list of similar Solutions	Implemented	
W-15	PoS allows advertising of Trials from the Solutions	Implemented	
W-16	PoS allows adding external references to the Solution	Implemented	
W-17	PoS allows adding additional documentation to the Solution	Implemented	
W-18	PoS shows which Solutions can work together	In course of implementation	There are no test cases described.
W-19	RESTful service to export data to CMINE	In course of implementation	Exact requirements are missing.
W-20	RESTful service to export data to DRMKC	In course of implementation	Exact requirements are missing.
W-21	PoS site implements various filters	Implemented	
W-22	PoS site implements a validation functionality	Implemented	
W-23	PoS stores country profiles information	Implemented	
W-24	PoS site shall provide interface to other systems	Implemented	

Annex 3 – Solution PDF export example

Table of Contents

Table of Contents	1
CrowdTasker	2
Meta-information	2
Readiness	2
Innovation stage	2
Crisis size	2
Crisis Cycle Phase	2
Supported standards	2
Provider:	2
Supported Use Cases	2
Conduct just in time micro-training	2
Related CM functions	2
Inform and warn the population	3
Related CM functions	3
Manage volunteered situation reporting	3
Related CM functions	3
Illustrations	3
Situation	3
Business Process	3
Similar solutions	4
GDACSmobile	4
LifeX COP	4

CrowdTasker

CrowdTasker enables crisis managers to instruct large numbers of non-institutional volunteers with customizable tasks.

The received feedback is evaluated and visualized and provides crisis managers with a detailed overview of the situation, which is used in turn to trigger adequate disaster relief services.

Information is provided by volunteers that are already at a disaster site allowing to exploit numerous benefits:

Meta-information

Readiness

- TRL 7 - System prototype demonstration in operational environment

Innovation stage

- Stage 4: Early Adoption/ Distribution

Crisis size

- Local
- Regional

Crisis Cycle Phase

- Preparedness
- Response

Supported standards

1. Qualifications Handbook Incident Command in Fire and Rescue Services
2. Risk management - Vocabulary
3. VISOV: Social media in emergency situation #MSGU see also DICOM
4. Crisis management. Guidance and good practice
5. Customer Notification Process for Disasters

Provider:

- [CrowdTasker description at AIT site](#)
- [CrowdTasker portal](#)

Supported Use Cases

Conduct just in time micro-training

CrowdTasker allows the responsible organisations to micro-train the solution users "just in time" and on a "need to know basis", taking into account their geographic position and profile.

For example, the pre-registered volunteers can be informed of an approaching storm or flood and given explanations on how to prepare and how to react if and when the crisis occurs. Volunteers with special skills can be given different information from those that do not have such skills and a response form can be used to control if the volunteers have understood the information.

Related CM functions

- [Train individuals, teams and organisations](#)
- [Train resilient communities](#)
- [Manage spontaneous volunteers](#)
- [Deliver public information and advice](#)

- [Manage spontaneous volunteers during recovery](#)

Inform and warn the population

CrowdTasker allows the responsible organisations to inform people (solution users) in affected areas in advance, while simultaneously offering a situation map of the measures already implemented. The information provided to CrowdTasked users can be context specific, e.g. dependent on the users position or skills. Furthermore, they can also be instructed to deliver the message to their friends, family and neighbours if necessary.

Related CM functions

- [Communicate hazard information to the public](#)
- [Set-up dissemination and information sharing](#)
- [Manage warnings](#)
- [Provide warning and alerts for secondary hazards](#)
- [Maintain public awareness on hazards and respective services](#)

Manage volunteered situation reporting

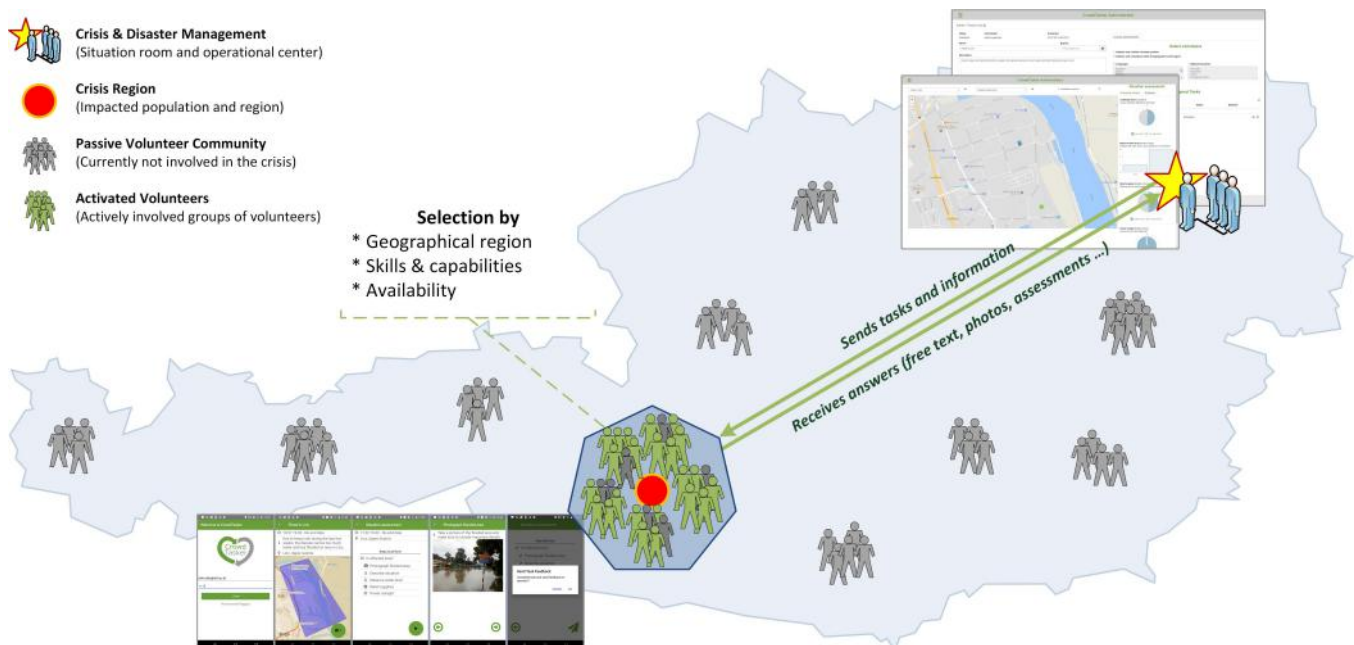
Volunteers using the CrowdTasker can be instructed to report the situation on the field, e.g. for initial damage assessment in a particular area. Helpers can be asked specific questions before reporting back on the situation and providing images where appropriate. All reports are located and displayed on an interactive map for the incident commander, helping to improve task planning and prioritisation.

Related CM functions

- [Conduct damage and needs assessment](#)
- [Monitor the affected area](#)
- [Conduct systematic monitoring and data collection](#)

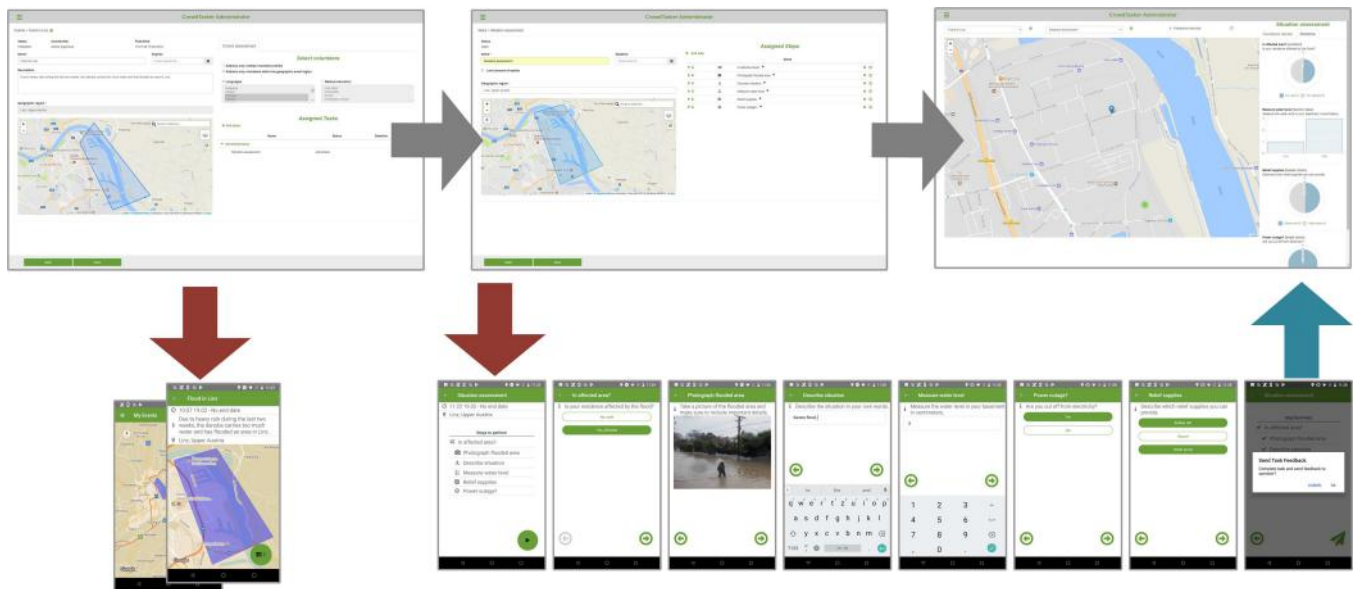
Illustrations

Situation



CrowdTasker situation overview

Business Process



CrowdTasker Business Process

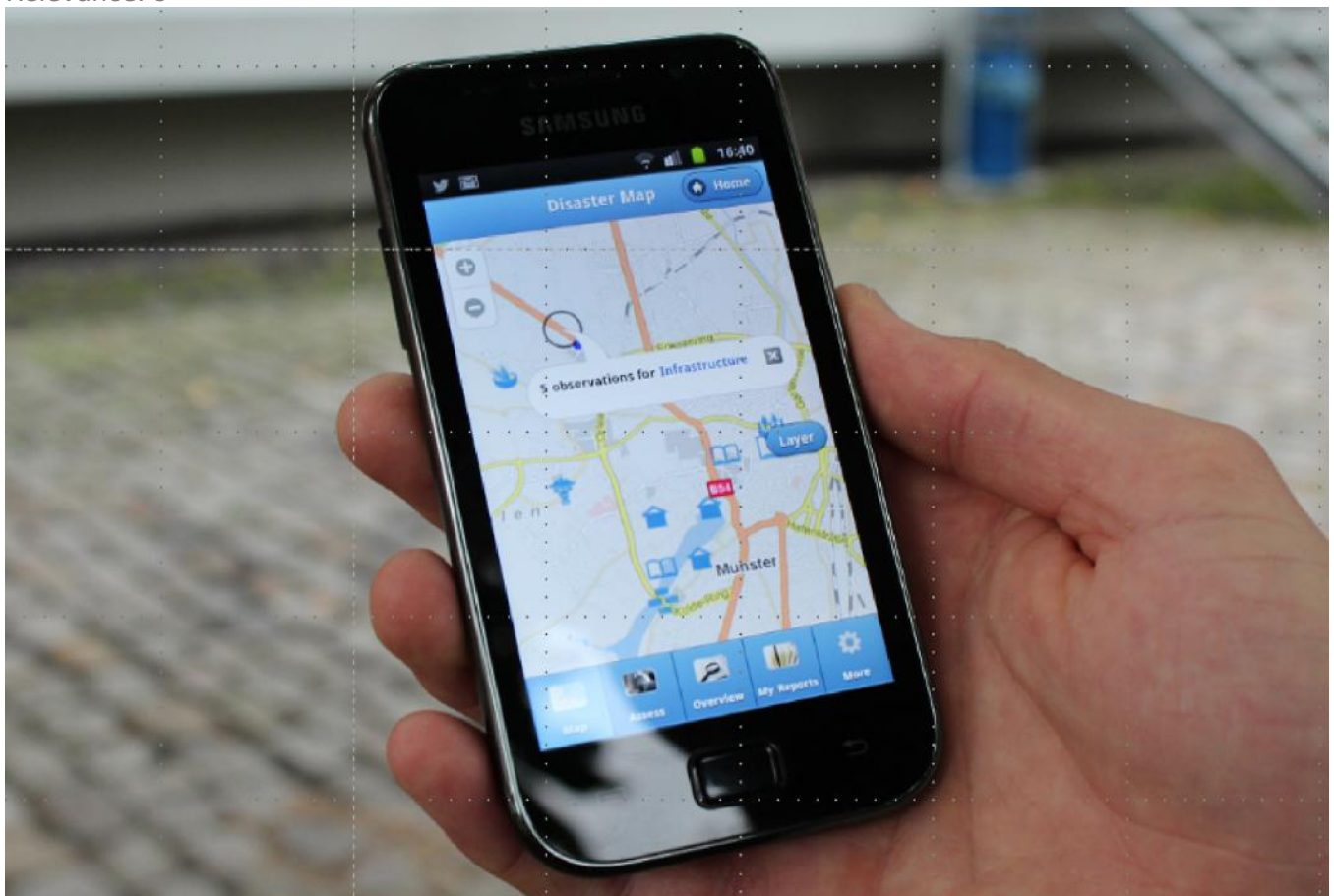
Similar solutions

Similar Solutions

GDACSmobile

[Stage 3: Initial Piloting](#)

Relevance: 9

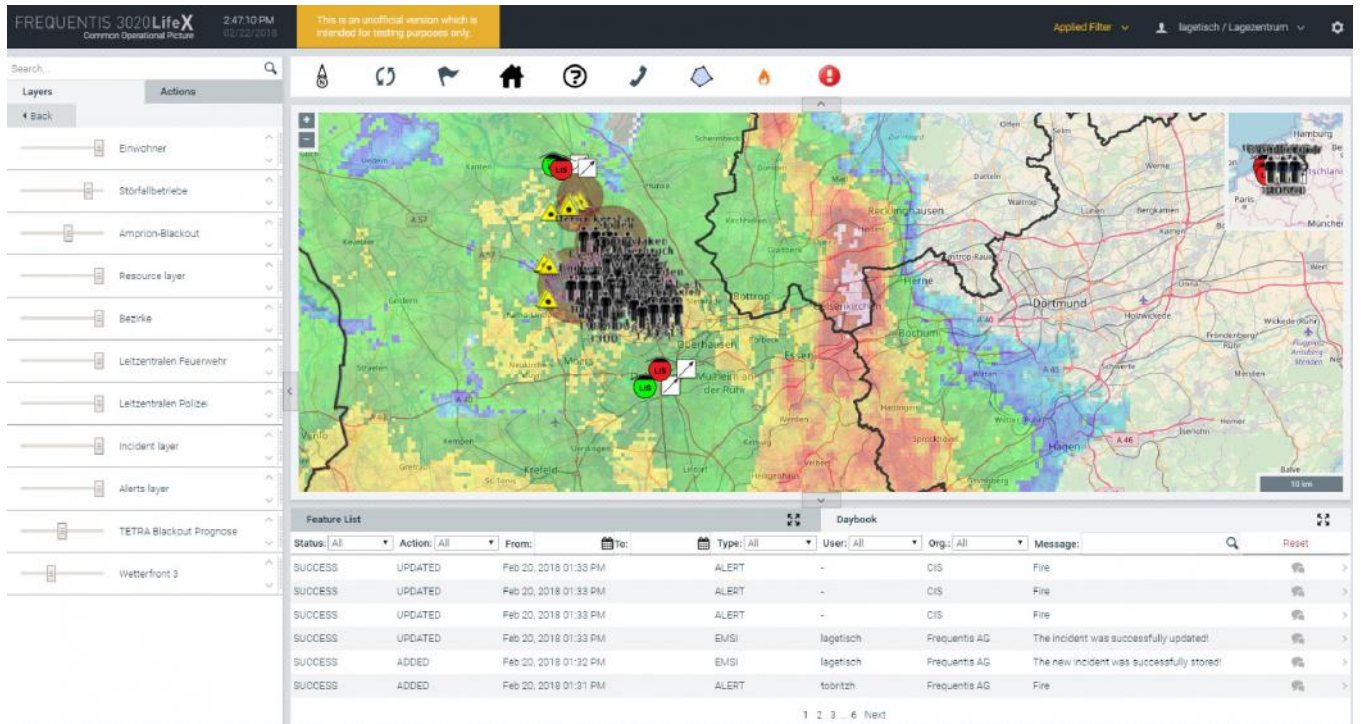


GDACSmobile device view

LifeX COP

[Stage 4: Early Adoption/ Distribution](#)

Relevance: 2



GUI