



DEVICE-EDGE-CLOUD INTELLIGENT COLLABORATION FRAMEWORK

Grant Agreement: 101092582

D1.1 Project Handbook



This project has received funding from the European Union's Horizon Europe Research and Innovation Programme under Grant Agreement No 101092582.



Document Information

Deliverable number:	D1.1
Deliverable title:	Project Handbook
Deliverable version:	0.1
Work Package number:	WP1
Work Package title:	WP1 Project Management
Responsible partner	UGOE
Due Date of delivery:	2023-03-31
Actual date of delivery:	(DD.MM.YYYY)
Dissemination level:	PU
Type:	R
Editor(s):	Stefanie Mühlhausen (GWDG) Julian Kunkel (GWDG/ UGOE) Marcus Merz (GWDG)
Contributor(s):	Jonathan Decker (UGOE)
Reviewer(s):	Vanessa End (GWDG) Sabri Pillana (FBU)
Project name:	Device-Edge-Cloud Intelligent Collaboration framEwork
Project Acronym:	DECICE
Project starting date:	2022-12-01
Project duration:	36 months
Rights:	DECICE Consortium

Document History

Version	Date	Partner	Description
0.1	2023-02-07	UGOE/GWDG	First draft

Acknowledgement: This project has received funding from the European Union’s Horizon Europe Research and Innovation Programme under Grant Agreement No 10192582.	Disclaimer: The content of this publication is the sole responsibility of the authors, and in no way represents the view of the European Commission or its services.
---	---

Contents

1 Purpose and Scope of the Deliverable	6
2 Abstract / publishable summary	6
3 Project Objectives	6
4 Changes made and/or difficulties encountered, if any	6
5 Sustainability	6
6 Dissemination, Engagement and Uptake of Results	7
6.1 Target audience	7
6.2 Record of dissemination/engagement activities linked to this deliverable	7
6.3 Publications in preparation OR submitted	7
6.4 Intellectual property rights resulting from this deliverable	7
7 Detailed report on the Deliverable	7
7.1 Abbreviations	7
7.2 Project profile	8
7.3 Introduction	11
7.3.1 Objective	11
7.3.2 Approach	11
7.3.3 Responsibilities	12
7.3.4 Supporting tools	12
7.3.5 Processes	12
7.4 Overarching tool suite	12
7.5 Project Communication	16
7.5.1 Objective	16
7.5.2 Approach	16
7.5.3 Responsibilities	17
7.5.4 Supporting tools	17
7.5.5 Processes	17
7.6 Deliverables/ Milestone Handling	25
7.6.1 Objective	25
7.6.2 Approach	25
7.6.3 Responsibilities	25
7.6.4 Supporting tools	25
7.6.5 Processes	26
7.7 Quality management	27
7.7.1 Objective	27
7.7.2 Approach	27
7.7.3 Responsibilities	28
7.7.4 Supporting tools	29

7.7.5	Processes	29
7.8	Risk management	30
7.8.1	Objective	30
7.8.2	Approach	30
7.8.3	Responsibilities	30
7.8.4	Supporting tools	31
7.8.5	Processes	31
7.9	Governance	37
7.9.1	Objective	37
7.9.2	Approach	37
7.9.3	Responsibilities	38
7.9.4	Supporting tools	38
7.9.5	Processes	39
7.10	Reporting	41
7.10.1	Objective	41
7.10.2	Approach	41
7.10.3	Responsibilities	41
7.10.4	Supporting tools	42
7.10.5	Processes	42
8	References	43

1 Purpose and Scope of the Deliverable

This document provides an overview of management and administrative procedures that are in place to guarantee the quality of project results and efficient teamwork. Key aspects of these procedures and standards are discussed and agreed upon during the kick-off general assembly. This includes a description of the collaborative environment, how to organize material, an FAQ for typical questions of project partners, and plans for quality assurance, technical integration, risk management, communication and dissemination. For example, the QA and risk management plan defines guidelines, procedures, and checklists to be followed for project specific documentation, criteria for the reviews, communication, innovation potential, progress and impact indicators, and contingency plans. The plan also includes assigned deliverable and milestone review teams and documents the potential and emerged risks. Throughout the project, the project handbook will be constantly updated.

2 Abstract / publishable summary

This text is a reference for the tools and procedures used in the DECICE project. It shall support involved employees in the understanding of the project management procedures to prevent confusion and conflicting decision-making. In preparation of this handbook, we collected best practices for efficient teamwork and project management and administrative procedures. We then identified tools to help us implement said best practices and formulated the concepts. We discussed the concepts internally and, in a next step, with all project partners. We used this feedback to iteratively improve existing concepts, add clarifications and include additional concepts if needed. This text is a compilation of the current procedures and standards. It is an open and living document that will be constantly updated throughout the project.

Note: The version published on Zenodo will not include some links as they would allow direct access to internal resources.

3 Project Objectives

This deliverable is helpful for project management, implementation and indirectly may contribute to the achievement of project macro objectives.

4 Changes made and/or difficulties encountered, if any

The work conducted for and the content of this deliverable fits the expected content – there is no deviation to the plan.

5 Sustainability

This is a central deliverable that paves the road for a successful collaboration. In terms of synergies, we gathered best-practices from other GWDG projects and created a suitable management structure for the DECICE project. During this effort, we realized that various procedures and descriptions are not well described or could be improved.

6 Dissemination, Engagement and Uptake of Results

6.1 Target audience

As indicated in the Description of the project, the audience for this deliverable is:

✓	The general public (PU)
✓	The project partners, including the Commission services (PP)
	A group specified by the consortium, including the Commission services (RE)
	This report is confidential, only for members of the consortium, including the Commission services (CO)

While we make this deliverable generally available, particularly to exchange best-practices, the target audience is the project partners. The content of the deliverable is presented in our monthly DECICE seminar.

6.2 Record of dissemination/engagement activities linked to this deliverable

See Table 1.

Type of dissemination and communication activities	Details	Date and location of the event	Type of audience activities	Zenodo Link	Estimated number of persons reached
Presentation	Internal kick-off for the project	2022-11-07	PP	-	30
Presentation	Kick-off of the project	2023-03-28/29	PP	-	40

Table 1: Record of dissemination / engagement activities linked to this deliverable

6.3 Publications in preparation OR submitted

None

6.4 Intellectual property rights resulting from this deliverable

None

7 Detailed report on the Deliverable

7.1 Abbreviations

BBB BigBlueButton

C Code

CA Consortium Agreement
CCB Change control board
CO Confidential
DECICE Device-Edge-Cloud intelligent collaboration framework
DEM Demonstrator
O Other
P Prototype
PU Public, fully open
R Document, report
R+C Report and code
WP Work package
WPL Work package leader

7.2 Project profile

Project name and ID	DECICE project, proposal ID 101092582
Call	HORIZON-CL4-2022-DATA-01
Principal	Prof. Dr. Julian Kunkel
Participants	Georg-August-Universität Göttingen Stiftung Öffentlichen Rechts (UGOE) Gesellschaft für Wissenschaftliche Datenverarbeitung MbH Göttingen (GWDG) E4 Computer Engineering Spa (E4) Kungliga Tekniska Hoegskolan (KTH) Universität Stuttgart (USTUTT) Huawei Technologies Düsseldorf GmbH (HWDU) Forschung Burgenland GmbH (FBU) SYNYO GmbH (SYNYO) Marmara University (MARUN) Consorzio Top-ix - Torino E Piemonte Exchange Point (TOP-IX) Alma Mater Studiorum - Universita Di Bologna (UNIBO) The Numerical Algorithms Group Limited (NAG) BIGTRI Bilisim Anonim Sirketi (BIGTRI)
Project lead	UGOE
Research area	AI-enabled computing continuum from Cloud to Edge
Duration	36 month
Start date	2022-12-01
End date	2025-11-30
EU contribution	5,627,250.00 €
Aim	AI-based, open and portable cloud management framework

Milestones/ Deliverables

Deliverable (number)	Deliverable name	Work package number	Short name of lead participant	Type	Dissemination level	Delivery date (in month)
D1.1	Project Handbook	WP1	UGOE	R	PU	4
D1.2	Data Management Plan	WP1	GWDG	R	PU	6
R1	Periodic Report	WP1	UGOE	R	PU	20
R2	Final Report	WP1	UGOE	R	PU	38
D2.1	Specification of the Optimization Scope	WP2	NAG	R	PU	9
D2.2	Digital Twin	WP2	UNIBO	O	PU	12
D2.3	AI-Scheduler Prototypes for Storage and Compute	WP2	FBU	R	PU	12
D2.4	Integrated AI-Scheduler Prototype	WP2	FBU	R	PU	24
D2.5	Final Scheduler and Digital Twin	WP2	NAG	C	PU	36
D3.1	Synthetic Test Environment	WP3	UGOE	C	PU	12
D3.2	Final Architecture and Interfaces	WP3	GWDG	R	PU	24
D3.3	Final Implementation	WP3	GWDG	R+C	PU	36
D3.4	Security and Trustworthiness	WP3	KTH	R	PU	24
D4.1	Implementation Report of CI/CD Environment	WP4	USTUTT	R+C	PU	6
D4.2	Integration of Monitoring Framework	WP4	USTUTT	R+C	PU	12
D4.3	Final Integration of DECICE APIs	WP4	USTUTT	R+C	PU	36
D4.4	Final Integration of HPC and AI Services	WP4	USTUTT	R+C	PU	36
D5.1	Use Case Requirements	WP5	TOP-IX	R	PU	9
D5.2	Development Environment Specification	WP5	E4	R	PU	12
D5.3	Project Development Environment Deployed for Phase 1 and 2	WP5	E4	DEM	PU	18

Deliverable (number)	Deliverable name	Work package number	Short name of lead participant	Type	Dissemination level	Delivery date (in month)
D5.4	Project Development Environment Deployed for Phase 3	WP5	E4	DEM	PU	24
D5.5	Performance Evaluation Report	WP5	E4	R	PU	36
D6.1	Dissemination & Communication Plan	WP6	SYNYO	R	PU	6
D6.2	Exploitation Strategy	WP6	GWDG	R	CO	18
D6.3	Online and Media Presence	WP6	SYNYO	R	PU	6
D6.4	Engagement Summary Report	WP6	SYNYO	R	PU	36

Table 2: List of deliverables

Deliverable (number)	Deliverable name	Short name of lead participant	Reviewer
D1.1	Project Handbook	UGOE	Vanessa End (GWDG), Sabri Pillana (FBU)

Table 3: List of reviewers

Milestone number	Milestone name	Related work package(s)	Due date (in month)	Means of verification
M1	Requirements Analysis Documentation	WP2, WP4, WP5	M6	Workload/infrastructure/metric specifications and use cases defined for the internal design and between WPs is gathered. Drafts of technical documentation is released
M2	Architecture and Interfaces Specified	WP2-WP4	M9	Documentation of interfaces and description of architecture is release
M3	Virtual Cloud Environment Prototype	WP2-WP3	M12	Implementation of core components running in a virtual environment for AI training and verification
M4	Test Deployment	WP2-WP5	M18	Integrated prototype running on real system environment
M5	System Evaluation Started	WP2-WP5	M24	First evaluation on in-situ measurements
M6	Final Deployment	WP4-WP5	M32	Final version running on project development environment

Table 4: List of milestones

7.3 Introduction

7.3.1 Objective

This handbook has two objectives. On the one hand, it serves as a collection of project-specific knowledge about processes, tools and techniques. Thus, this manual serves as a reference work for all project participants, whether they have been involved from the beginning or have joined the project during its runtime. On the other hand, this manual is part of the deliverables for WP1.

7.3.2 Approach

The project partner responsible for project management made the initial proposals for the tools to be used and the resulting processes. The processes and structure are derived from the experience of the authors and from tailoring existing management approaches, esp. Prince2 [EMPa], to the needs of the project. The outcomes were presented to the project partners during the kick-off meeting and then implemented. Subsequent changes will be presented in our monthly meeting. The project partners are given the opportunity to discuss these proposals. The procedures are constantly reviewed during the project runtime. If approaches are identified that have prospect to improve our

state of the practice, they are considered and discussed. The achieved agreements are recorded in this document.

7.3.3 Responsibilities

UGOE is the partner responsible for project management. As such, it is the responsible party for proposing and documenting suitable project management structures and insisting on their compliance. All project partners are required to cooperate, perform and fulfil their duties according to the project plan. This includes adhering to agreed management structures.

UGOE is responsible for disseminating this hand book as part of the WP1 deliverables.

7.3.4 Supporting tools

This document was created with an instance of the collaborative, online LaTeX editor ShareLaTeX hosted by the GWDG and can be found in our Git repository.

7.3.5 Processes

Updating the project handbook This is a living document and meant to be updated constantly throughout the project phase.

- Each project member may participate in keeping the information in this handbook up to date.
- As soon as information changes, the project member first becoming aware of this may change the corresponding piece of information in this handbook.
- Any changes made by project members who are not part of the project office, must undergo review.
 - Create a new feature branch as working branch.
 - Commit changes to this feature branch and push them to origin.
 - Create a merge request using the GitLab web interface. Source branch must be the newly created feature branch, target branch master.
 - Notify the project office of the merge request via mail or chat.
 - A member of the project office will review the features, clarify any open questions and merge the changes back into master once approved.

7.4 Overarching tool suite

The following paragraphs describe how to access the wider collaborative tool suite that require prior registration of new participants. We request that all contributors follow these steps to register to these services. To access IT services offered by GWDG, a universal user account called AcademicID must be created first. This is done through the single sign on provider **Academic Cloud**. Also, for various services a first login is necessary to provision your service-specific account, then the project office can add you to the resources.

Academic Cloud (Cloud service suite) <https://academiccloud.de/>

Academic Cloud serves as a single sign on solution for all services hosted by GWDG. To register to Academic Cloud, visit <https://academiccloud.de/>.

1. Below “Don’t have an Academic ID?” click on “Sign Up”.
2. Enter your email address.
3. If you get the message “Sorry, your institute could not be determined automatically.”, click on “I didn’t find my institute”. Fill in the form “Create new account”, for “invite token (optional)” enter “Julian Kunkel/DECICE Project”.

BigBlueBlutton (Video conferencing tool) <https://.....¹>

BBB is the main tool for video conferencing and online presentations within the project. Project meetings (WPL or WP meetings) shall utilize this room. This allows to record the meetings in a central place to ensure information is available to absent peers. For specific topics, face-to-face discussions or overlaps in the schedule, further rooms can be created by project members on demand.

- Do not share the URL with externals as all recordings are available to anyone with the room URL.
- To use the BBB instance of GWDG, no login is required. To join a meeting, simply click on the link and enter your name. Use a Chrome-based web browser for BBB to work properly.
- Recorded sessions are listed, once rendered, on the start page of the room.
- The meeting room dedicated for the DECICE project is open to all project members and can be used anytime. This includes work package specific meetings where the project is not involved directly.

GitHub (Code repository) <https://github.com/>

Publicly facing code (under an open source license) will be published on GitHub. A dedicated repository for DECICE will be created once required.

GitLab (Collaborative document management) <https://gitlab-ce.gwdg.de>

At <https://gitlab-ce.gwdg.de/decice> there are separate repositories for project management and individual WPs. Further repositories can be created as required. The project management repository in particular is used to archive meeting notes and presentations of project-wide meetings, as a reference point for templates and checklists for deliverable preparation, and for reports, deliverables and publications. It can be found under <https://gitlab-ce.gwdg.de/decice/decice-management>. Project code should go into a work-package specific, separate repository. GitLab issues for individual tasks can be created as needed in the work-package specific repositories. Those issues must be updated regularly such that they hold all information needed to work on the given task, e.g. additional

¹The URL is private to prevent access as it is an internal project resource.

information gathered in personal communication, for example via chat. Individual repositories are not listed here, as they are subject to change. The full list of all available repositories can be found at <https://gitlab-ce.gwdg.de/decice>.

1. If not already done, register at Academic Cloud (see Section 7.4). To use GitLab, you need an Academic ID.
2. Log in to Academic Cloud.
3. Log in once to GitLab <https://gitlab-ce.gwdg.de> to activate your account for GitLab.
4. Contact the project office.
5. The project team will add you to the GitLab-DECICE group.
6. You now have access to all repositories related to the DECICE project.

Mailing lists There are specific mailing lists for the different groups of recipients:

- `decice-project-all` → for all project members
- `decice-project-wpl` → for WPLs
- `decice-news` → for external
- `decice-project-wp1` → for project members contributing to WP1
- `decice-project-wp2` → for project members contributing to WP2
- `decice-project-wp3` → for project members contributing to WP3
- `decice-project-wp4` → for project members contributing to WP4
- `decice-project-wp5` → for project members contributing to WP5
- `decice-project-wp6` → for project members contributing to WP6

Members can subscribe to the mailing lists by themselves. The procedure is described in Section 7.5.5.

OpenProject (Project management) <https://projects.academiccloud.de>

1. If not already done, register at Academic Cloud (see Section 7.4). To use OpenProject, you need an Academic ID.
2. Log in to Academic Cloud.
3. Log in once to OpenProject <https://projects.academiccloud.de/projects/22-decice/> to activate your account for OpenProject.
4. Contact the project office.
5. The project team will add you to the DECICE project.

6. You now have access to the project.

Individual work packages, milestones and deliverables defined in the project proposal have been migrated to OpenProject and are provided with a deadline. During project work, artefacts in OpenProject will be assigned to respective project member(s) set to the specific task. The status will be updated as work progresses. Relevant statuses are “New” for unassigned project goals, “In progress” for goals that are currently worked on, and “Closed” for completed goals.

OwnCloud (Storage Service) <https://owncloud.gwdg.de>

The OwnCloud service is used to host and share files as well as documents which are more suitable to be worked on in a live environment in a collaborative manner. Excel sheets as well as social media content will be updated on a regularly basis and can be found under: <https://owncloud.gwdg.de/index.php/s/NC24GQ5rAnfvqyZ>. The file sharing service also works with the single sign on and can be used with an Academic ID.

1. If not already done, register at Academic Cloud (see Section 7.4). To use OwnCloud, you need an Academic ID.
2. Log in to Academic Cloud.
3. Visit <https://owncloud.gwdg.de> to upload or work on files.

Pad (Collaborative note-taking) <https://pad.gwdg.de>

1. If not already done, register at Academic Cloud (see Section 7.4). To use hedgedoc, you need an Academic ID.
2. Log in to Academic Cloud.
3. Visit <https://pad.gwdg.de> to create a new pad or re-open an existing pad.
4. Share the link to the pad for collaborative work on the document. Anyone with the link can edit the shared document with no further login needed.
5. There is also an option to publish a pad. The link of the published version does not allow further editing of the page.
6. Published or not, pads do not exist permanently and are therefore not suitable for long-term archiving of information.
7. Pads for specific meetings are listed in decice-management repository on the “notes” page.

RocketChat (Chat service) <https://chat.gwdg.de/>

1. If not already done, register at Academic Cloud (see Section 7.4). To use RocketChat, you need an Academic ID.
2. Log in to Academic Cloud.

3. Log in once to RocketChat <https://chat.gwdg.de/> to activate your account for RocketChat.
4. Contact the project office.
5. The project team will add you to the DECICE channel where conversation with all project members occur.
6. You now have access to the channel and can post to the channel, i.e. to all project members or start private conversations with individuals or small groups as needed.

ShareLaTeX (LaTeX platform) <https://sharelatex.gwdg.de>

1. If not already done, register at Academic Cloud (see Section 7.4). To use RocketChat, you need an Academic ID.
2. Log in to Academic Cloud.
3. Visit <https://sharelatex.gwdg.de> to start a new project or re-open an existing one.
4. Upload the report or presentation template from the DECICE management GitLab.
5. To work collaboratively on a document, click on “Share”, then click on “Turn on link-sharing”. Anyone with the invitation link can edit the shared document with no further login needed.

Zenodo (General purpose repository) <https://zenodo.org/communities/decice-project/>
There is a dedicated space ('community') to deposit all written project results at Zenodo.

7.5 Project Communication

7.5.1 Objective

Clear communication, document workflows and structures facilitate the collaboration and organization of large scale projects, reduces over-communication between non-related tasks/WP and therefore reduces noise and increases efficiency. To ensure an understanding of communication within the project, the tools and processes for communication are defined in this section.

7.5.2 Approach

To ensure good communication, a few, clear processes are defined for the project. The following communication scenarios and processes should be heeded throughout the project.

Communication is coordinated hierarchically within each level of the project as follows: steering group (contains one representative per institution and all work package leaders), the work package, the task leader and contributors to a task. The respective work package leader or task leader have to ensure all information are communicated up- and downstream.

7.5.3 Responsibilities

The partners shall utilize the tools and processes for project communication. In addition, it is the responsibility of the individual partners to comply with the agreed guidelines.

Work packages organize themselves utilizing the here listed tools and processes. Work package leaders play a central role in this regard, as they ensure that the common framework is heeded in the work package they lead. They may designate task leaders for specific tasks in their work package to support them with this. Work package and task leaders organize the work, report the status back to other work package leaders and the project office regularly and, crucially, oversee that the timeframe for their work package is kept.

UGOE proposes tools and processes for project meetings, internal technical and administrative discussions, note-taking and subsequent dissemination. UGOE and GWDG set up these tools and make them available to the project partners.

SYNYO creates and implements tools and processes for external communication. SYNYO also takes care of public outreach.

Uploading individual results to Zenodo after approval is the responsibility of the respective WPL. Any result deemed to be ready to be published will have undergone several feedback loops, including one with the project office (cf. Section 7.6.5 on how to prepare deliverables).

7.5.4 Supporting tools

BBB Platform for virtual meetings (cf. Section 7.4)

GitLab Software development and file storage (cf. Section 7.4)

Mailing lists (cf. Section 7.4)

OwnCloud Storage service (cf. Section 7.4)

Pad Collaborative markdown notes (cf. Section 7.4)

RocketChat Messaging service (cf. Section 7.4)

Zenodo General purpose repository (cf. Section 7.4)

7.5.5 Processes

Work packages utilize communication channels, such as meetings, chat and mailing lists to their specific needs. General purposes of different communication channels and consequential recommendations for their usage are listed in the following processes.

Handling new/leaving contributors During a long-running project, contributors will be entering and leaving the project. This has to be communicated to the team in order to ensure logins and permissions are granted and everyone is informed.

New contributors When a new team member wants to participate in the project development, perform the following steps:

1. The new contributor creates an academic ID under <https://academiccloud.de/>
2. The new contributor has to log in to GitLab once <https://gitlab-ce.gwdg.de/>
3. The new contributor has to log in to OpenProject once <https://projects.academiccloud.de/>
4. The new contributor has to log in to RocketChat once <https://chat.gwdg.de>
5. The team lead writes an email to the project office or a named deputy with the subject “DECICE: new contributor” and name, email and academic ID of the new contributor as content. If the team lead is unavailable, the new contributor or another, existing member of the project may write this email but has to provide a reference in CC.
6. The responsible person in the project office or his/her deputy ensures that the new contributor has access to the used tools in the project. He/she can act only after the email has been written. If the mail was written by an established member, access can be granted immediately. If the mail request was written by the new contributor, the responsible person has to check with the given reference in CC first. Should for some reason no reference has been given, the responsible person has to ask the new contributor for a reference and does not grant any permissions before the request is confirmed.

Leaving contributor If a contributor leaves the project, his permissions must be withdrawn.

1. The leaving contributor or the according team leader writes an email to the responsible person with subject “DECICE: deny contributor” or “DECICE: leaving contributor”. The body has to contain the name, email and academic ID of the leaving contributor.
2. The responsible person checks if the mail was written by a valid person – either team lead or the contributor him/herself and withdraws the contributor’s permissions from the tools.
3. If the user leaves the organization, then the account in the Academic Cloud will be deleted by the GWDG.

A denied contributor will continue to have access to tools that utilize a private link for access control such as pads and the video recordings of meetings. This is considered an acceptable risk — an according entry in the risk register (cf. Section 7.8) has already been created.

Disseminating content The workflow for content creation can be found here <https://.....>². Besides the workflow, the document also contains the checklists for authors as well as for reviewers. The workflow for creating content should consist of:

1. Familiarise yourself with the requirements for the document.
2. Write the content

²The URL is private to prevent access as it is an internal project resource.

3. Send the document for peer-review
4. Document is being reviewed
5. Consolidate and integrate comments and suggestions as needed
6. Send the improved document to PI/supervisor's for review/confirmation
7. Consolidate and integrate remaining comments and suggestions
8. Upload the document to GitLab and inform the project office for acceptor's review

Requirements regarding the content, organization and readability for documents as well as specific checklists for reviewers and authors which describe the procedures in detail can be found in the pad above.

For internal usage, deposit content in the appropriate GitLab repository. For external usage, upload content such as project reports, presentations or open source publications to the DECICE community at Zenodo after undergoing the review. Therefore, proceed as follows:

- Login to Zenodo (you must create an account)
- Visit <https://zenodo.org/deposit/new?c=decice-project> to upload your content directly into the DECICE community. If you upload your content via the generic upload dialogue, ensure to add it to the DECICE community manually by selecting "DECICE" in "communities section" in the Zenodo upload dialogue.
- Use the title as it is in the project (if deliverable or milestone)
- Add the appropriate metadata, e.g., all the involved authors.
- Use the (general public understandable) synopsis as a description.
- Update the dissemination records in OwnCloud by using the corresponding dissemination reporting document (cf. Section 7.4).

Handling conflicts Only with an open discussion culture, we can resolve contradictory opinions and achieve a sustainable result. Please bring all arguments to the table and resolve the dissent.

A conflict is a serious disagreement or argument that cannot be resolved in a timely fashion. If someone feels emotionally too much involved in the discussion that it impairs the result, the issue shall be escalated.

WPLs moderate all arising, internal conflicts on the task level. If a WPL reaches their limit or is personally involved, conflicts should be escalated to the project coordinator.

Mails Mails are always an option to communicate in a more conservative way. Be it bi-lateral or via the project mailing lists. Mails are especially useful to ensure confidentiality or validating the source (signing or encryption) of information.

The normal mail etiquette applies to avoid cluttering and loss of information. The sender of an email is responsible:

1. To select the minimum/correct recipients/ mailing list of the message:
 - (a) “To:” recipients – persons who should read the mail or may have tasks/questions in the mail.
 - (b) “CC:” recipients – interpreted as FYI, should but does not necessarily need to read. No tasks/questions for these persons. Text could be interesting or they could contribute to the discussion.
 - (c) “BCC:” recipients – pure FYI
2. Select a suitable subject. If something is urgent and needs an immediate clarification, add “URGENT” at beginning of the subject line.
3. Write the body of the mail. One email should usually only cover one topic, do not add further or unrelated topics. The topic should cover the subject. This ensures that mails can be sorted in a good manner.

The initiator of an email is responsible to transfer information or decisions from the mail they started to the according meeting notes or GitLab issues (cf. Section 7.5.5).

Things to avoid:

- Including everyone in the recipients list. The use of mailing lists is recommended in that case.
- Thread hijacking. A practice to add topics not covered by the subject to an existing mail discussion. The subject stays the subject of a mail.

Internal communication Short questions and anything that needs a direct answer can go through RocketChat (preferred) or mails. Decisions made or know-how generated in such conversations such as via RocketChat need to be recorded in the running meeting notes in the pad, in GitLab or in the project handbook. It should be avoided to add information to the notes of past meetings. If this has to be done for some reasons, the relevant information should be added to the exported meeting notes as well.

- Longer-term discussions should be resolved in topic-oriented meetings.
- Technical discussions on specific WPs should take place via GitLab issues.
- Information kept in RocketChat or mails is considered to be lost for the project.

Meetings Meetings are, unless stated otherwise, virtual meetings. Project-wide meetings are chaired by the coordinator, otherwise by the responsible person. In order to ensure that meetings utilize the scheduled time efficiently and produce an outcome, the organization and execution has to follow a process which includes preparation, execution and follow-up.

Meetings are usually scheduled regularly but additional meetings may be conducted as needed. Each WP follows its own regular rotation. Each meeting, its respective schedule and meeting notes can be found in the GitLab repository (<https://gitlab-ce.gwdg.de/decice/decice-management/-/tree/main/notes>). The BBB meeting room should be used for this to allow recording of the sessions for prospect attendees that could not attend. Meetings just stating a status without need for discussion should be

avoided. This kind of communication should be done by utilizing status reports. To keep track of the existing meetings, corresponding topics and the link to the according pad used for the minutes of the meeting is provided in the decice-management repository in folder “notes”.

Preparing a meeting The responsible person for a meeting has to prepare the meeting ahead of time, and inform all participants beforehand to ensure everyone is prepared. This includes:

1. Select the pad for minutes of the meeting — the same pad can be reused/extended for the same meetings. There should be one pad for the WPL/steering committee meeting and one per WP that contains chronologically all the information for quick access. Pads for topic-specific detailed/extensive meetings can be created but must be referenced by the WP pad – the WP pads are also referenced by the WPL/steering committee meeting pad.
 - (a) Check the “notes” README page of decice-mangement repository for existing meeting pads.
 - (b) If a pad for the according meeting exists, use that pad in further steps, otherwise create a pad and link it with the topic/subject/purpose on the main meeting pad. Check the minutes of the meeting template in the repository.
2. Ensure that there is enough space in the pad. The maximum character count, which can be found on the bottom right of the editor, is 100,000. If the pad is around 90,000 old meeting notes but the last 3 can be deleted.
3. Create a new section for the meeting in the pad.
4. Fill in the attendee list for mandatory attendees.
5. By default, the organizer is the moderator – s/he can define a deputy, though.
6. Create the agenda. The agenda should be doable in the estimated timespan. Items from previous meetings that are found in an “Open items” section have to be considered for the agenda. The following standard items should be on the meeting notes:
 - Attendees – everyone can add the institution and their name by themselves, mandatory people should be added by the organizer in advance, also apologized invitees should be listed.
 - Status – every partner should give an update of the progress of work.
 - Pending actions – the moderator shall check together with the attendees the due-tasks from the agreed Actions/ToDo list of the past agenda.
 - ... Pending items copied over ... – any old item should usually be addressed first.
 - ... Your agenda here...
 - Urgent/Other – Urgent last minute matters and anything minor/noteworthy.
 - Actions – agreed / committed actions by the attendees with the names and their deadlines. If no deadline is specified, it is assumed that the action is worked on until the next meeting.

7. Schedule a date/time and invite all participants. The regular WP and WPL meetings are jour fixe.
8. Check and adjust the agenda before the meeting to process possible inputs by the participants. Assess if the topics can be dealt within the given timeframe. If not a triage has to be done and the exceeding topics marked. Take a 5-minute buffer for summarizing tasks into account. Always end the meeting in time – best 5 minutes before the announced end time.

Participants have to

1. Check the agenda beforehand, add some information regarding your status. If subjects to discuss are missing in the agenda, they have to add an according item. To make it clear who added the item, it must be delimited with their name.
2. Confirm/reject their participation in the calendar invite. You may also add yourself to the attendee list as apologized.
3. Define a deputy for you if your attendance is required.

Executing a meeting The moderator guides through the agenda and ensures that the time frame is adhered to. A dedicated note-taker should be identified – however, all attendees should contribute to the meeting notes if possible. Specialist discussion that only interest a small subset of attendees should be avoided and shifted to a Face-to-Face conversation. The note-taker is ultimately responsible for taking notes of the meeting. Everyone should support and extend the notes to ensure that important items are not lost. Tasks have to be clearly marked with Markdown syntax “* [] << name >> << target date >>” where name denotes the name of the responsible person.

Adding to the agenda during the meeting is only allowed if the attendees (and moderator) agree. Discussion items missing should be added to the “Urgent/Other” section (including the name of the author). At the end of the meeting, the tasks are collected in the Actions section and have to be agreed on.

Follow-up a meeting Traditionally, a notetaker:

1. Creates a summary of the meeting.
 - (a) Summarizes the outcome of the meeting.
 - (b) Collects all not discussed items/topics from the agenda as open items and moves them to the next meeting.
 - Checks with the according author of the topic.
 - Move the item to a dump section of the meeting pad with an according comment. This avoids loss of information about decisions.
 - Adds the item to the “Open Items” list to be assessed before the next meeting if not dumped.
2. Sends the summary to the organizer of the meeting.

However, we integrate this process into the meeting itself, i.e., at the end of the meeting, it should be summarized in the then existing meeting notes, non attended topics are copied to the agenda of the next meeting. If this cannot be done in a timely fashion, then please wrap it up following the traditional approach.

The organizer:

1. Checks, extends and corrects the summary from the notetaker
2. Copies the summary and the meeting minutes to the according section in the GitLab repository "decice-management" (cf. Section 7.4).

The participants:

1. Check and asses their respective tasks.
2. Create issues for the tasks in the according GitLab repository including a deadline and all necessary information and assign the issue to themselves.
3. Add a link to the according issue to the task in the ToDo section to ensure crosslinking.
4. Mark the task as done by checking the checkbox. This makes it possible to track which tasks have already been completed

Creating issues in the repositories is crucial to allow a single source of truth with regard to task tracking. GitLab Issues have been designed to do that, pad tasks not.

Outreach to the general public A project homepage <https://decice.eu> has been set up and is maintained by SYNYO. In addition, DECICE is present on Twitter (https://twitter.com/DECICE_EU) and LinkedIn (<https://www.linkedin.com/in/decice-project-b0b55b25a/>). There will also be a newsletter.

All digital channels are managed by SYNYO and used at the suggestion of the project partners. To this end, SYNYO has set up spreadsheets and made them available to project partners. This content can be found in the DECICE GitLab repository in the WP6 subproject (cf. Section 7.4).

- The aim is to post two items per week. SYNYO sets up a rotating system for content suggestions from all project partners and ensures compliance.
- All project members are invited to suggest content. These suggestions should be placed with the respective team/work package leader. After internal approval, a content suggestion can be added to the respective spreadsheet, e.g. a social media post to the "social media spreadsheet". Spreadsheets are editable online at <https://owncloud.gwdg.de/index.php/s/NC24GQ5rAnfvqyZ>
- Relevant content includes but is not limited to project related press releases, events and publications.
- SYNYO collects input from the spreadsheets and follows up with the project partner who made the contribution by email for further clarifications.
- After a common decision on the content is reached, SYNYO uploads the content to the corresponding channel.

Subscribing to mailing lists It is the responsibility of the individual project member to subscribe to all relevant mailing lists. Typically, this will be the generic list 'decice-project-all' and mailing lists specific to the work package(s) the person is contributing to.

To subscribe to a list, visit the webpage prefixed with

<https://listserv.gwdg.de/mailman/listinfo/>

To subscribe to the aforementioned mailing lists, visit:

- <https://listserv.gwdg.de/mailman/listinfo/decice-news/>
- [³](https://.....)
- [⁴](https://.....)
- [⁵](https://.....)
- [⁶](https://.....)
- [⁷](https://.....)
- [⁸](https://.....)
- [⁹](https://.....)

Consortium Agreement The purpose and task of the Consortium Agreement (CA) is to specify the relationship among parties within the project, in particular concerning the organisation of the work between parties, the management of the project and the rights and obligations of the parties concerning liability, access rights and dispute resolution. The organisational structure of the consortium is comprised of the following Consortium Bodies:

1. The General Assembly - decision-making body of the consortium
2. The Coordinator - legal entity acting as the intermediary between the parties and the granting authority

Furthermore the General Assembly consists of one representative of each party. These members are deemed to be duly authorised to deliberate, negotiate and decide on matters regarding the CA. The coordinator chairs all meetings of the General Assembly, unless decided otherwise by the General Assembly. Each party agrees to abide by all decisions of the General Assembly. However this does not prevent the parties from exercising their veto rights or from submitting a dispute for resolution in accordance with the provisions of settlement of disputes as stated in the CA. Further information regarding the CA can be found in the DECICE repository: <https://gitlab-ce.gwdg.de/decice/decice-management/>.

³The URL is private to prevent access as it is an internal project resource.

⁴The URL is private to prevent access as it is an internal project resource.

⁵The URL is private to prevent access as it is an internal project resource.

⁶The URL is private to prevent access as it is an internal project resource.

⁷The URL is private to prevent access as it is an internal project resource.

⁸The URL is private to prevent access as it is an internal project resource.

⁹The URL is private to prevent access as it is an internal project resource.

7.6 Deliverables/ Milestone Handling

7.6.1 Objective

A common understanding of the way of working is the basis for achieving the objectives set in the project plan. The aim of this section is to describe basic aspects of cooperation and thus giving individual team members all tools needed to contribute to completing the project goals. For future reference, results to be achieved are listed in Table 2.

7.6.2 Approach

The project coordinator will propose procedures and methods to work efficiently towards the set goals and to overcome common obstacles. This includes, among other things, questions of work organization and time management. To further support project members in their work, the project office has prepared checklists for key aspects.

7.6.3 Responsibilities

The goals set are already allocated to the individual project partners (cf. Table 2). For each work package and the goals defined in it, one of the project partners was designated as the main responsible partner. For each goal, responsibilities are structured hierarchically as follows:

1. The project partner leading a work package appoints internally responsible persons who distribute the upcoming tasks and monitors their timely completion. Upcoming tasks include, but are not limited to, milestones and deliverables defined in the project plan.
2. Monitoring is technically supported by GitLab issues and tasks and milestones in OpenProject. Project members responsible for a given task ensure that the status of the corresponding artefacts in OpenProject are up-to-date. With regard to GitLab issues, they oversee that the issue contains all information necessary to work on it.
3. Work package leaders are in close contact with the steering board (implemented via WPL meeting) and project coordinator and exchange information about the current progress of the project once a month or as required.
4. The project coordinator keeps track of the overall progress, also with regard to possible dependencies between different work packages.
5. The project coordinator is responsible for collecting and submitting reports and deliverables to the Granting Authority.

7.6.4 Supporting tools

GitHub Code repository (cf. Section 7.4)

GitLab Software development and file storage (cf. Section 7.4)

OpenProject Project management software (cf. Section 7.4)

Zenodo General purpose repository (cf. Section 7.4)

7.6.5 Processes

Deliverable preparation

1. Individual project members work on the tasks assigned to them. They inform themselves about the exact requirements and the promised scope of work in the project proposal and work towards fulfilling those requirements. They also inform themselves about the workflow for documents, which is part of the corresponding checklist at the onset of their work.
2. Corresponding artefacts in GitLab (additional information on the task) and OpenProject (task status) are constantly updated, by arrangement either from the individual project member or the corresponding task lead.
3. Individual project members are in close contact with their task or work package leader and inform him/her about task progression and difficulties encountered, if any.
4. Work package leaders update each other and the project coordinator regularly about the progress, paying special attention to the promised scope of work and any deviations.
5. Acceptance of the content is carried out by and in close coordination with the project partners and the project office.
6. Actual deliverables including reports and code are prepared using the provided templates and checklists to ensure formal aspects such as uniform appearance of the documents. Documents need to be written and reviewed with the promised scope of work and deviations to it in mind.
7. Review process is performed according to the workflow for document preparation.

Delivering results

1. First and foremost, established quality assurance guidelines must be followed thoroughly.
2. Only the project coordinator can define a result as ready. Only then it can be published.
3. Publishing a result involves several steps, but at least the submission of an appropriate report to the stakeholders and subsequent upload to Zenodo.
4. Project coordinators uploads report to EU portal and informs EU project office.
5. If the project result is code, it should also be uploaded to GitHub.
6. If the nature and scope of the result is suitable for a scientific publication, a suitable journal should be determined and a draft created. The draft should then be circulated amongst all project partners. The consortium agreement specifies a notice period for planned publications.

Information regarding progress and delay The project coordinator must be informed as soon as possible, i.e. as soon as it becomes apparent that deadlines might not be holdable. The project coordinator will then evaluate with the responsible WPL what the reasons for the delays are and initiate appropriate countermeasures. These can include but are not limited to reallocating tasks and man-power, redefining the minimum viable product and changing prioritization.

Keeping track of time WPs themselves as well as all milestones and deliverables are planned in OpenProject. Since each individual task is marked with its submission date, OpenProject shows at a glance whether the project is on track. A WPL may use this tool to plan person months in their respective WP accordingly. At the very least, individual tasks need to be assigned to the corresponding person(s) and the status must be updated regularly. Should it become apparent that a task cannot be completed in time, see Section 7.6.5 for further information.

Technical questions GitLab provides an effective means of linking technical discussions directly to the code base through so-called issues. Questions of content should be solely addressed in GitLab issues.

7.7 Quality management

7.7.1 Objective

To consistently ensure high quality results, quality management is needed. All project partners will be briefed about the importance of keeping up a high quality product to achieve consistency in tasks and overarching activities. Furthermore, quality management routines will increase the efficiency in processes and improve the use of time and other resources.

7.7.2 Approach

Quality management takes place at various levels and throughout the whole duration of the project. Documents and software will be reviewed and tested continuously to ensure a constant improvement on two levels, administration as well as code quality. Aspects of quality management range from quality planning over controlling milestones and deliverables to evaluating results against the project plan and organisational obligations against the project board.

Milestones and deliverables of this project include both code and text. For document creation, accompanying checklists are stored in GitLab. They detail internal review processes preceding dissemination, as well as guidelines to ensure a consistent layout. Following the checklists and proposed review processes, a document is checked for completeness, accessibility and consistency before it is disseminated.

Code is a separate entity where additional and specific tools such as code reviews and tests are used for quality assurance. Additional emphasis for code lies on reliability and efficiency. Only after internal acceptance, software should be released and committed to a publicly available repository on GitHub. Quality assurance of software is deemed of utmost importance for the overall success of the project. To this end, one complete work package (WP5) is dedicated to testing, with associated tasks for compiling lists of use cases and test applications to be coded against and for designing a project development environment.

For all results, a “Definition of Done” (DoD) is given in advance and added as an issue with the label *QM* to the according work-package specific repository in GitLab. For example, the Definition of Done for WP2 goes into the *decice-wp2* repository.

The DoD should contain clear criteria which define when a product is considered to be verified against underlying criteria. These criteria have to be negotiated between the WPL, additional (task) leaders within the WP and the WPL of other WPs that are dependent on this particular WP. Including the WPL of dependent WP in this round will ensure that all expectations and downstream requirements will be fulfilled. Only if a result has been checked against the DoD it can be considered done and validated. This has several advantages:

- Using GitLab issues allows for easy filtering and tracking of DoD and their status.
- As GitLab issues are also used for risk management, this will ensure a smooth and frictionless management process.
- All kinds of changes including changes in responsibilities and changes to the DoD itself are tracked automatically and thus made transparent. Commenting on an issue allows keeping track of the underlying discussions that lead to a change.
- Having an agreed quality definition for products ensures that contributors consider the requirements in a very early stage. This will ensure that requirements are understood and will avoid later discussions.
- Without DoD criteria there would be no real quality assurance for deliverables as the status of “done” would be subject to interpretation, which contradicts the idea of quality management.

7.7.3 Responsibilities

It is the responsibility of work package leaders to supervise that the defined QM approach is followed through. This begins with not only assigning an individual task as derived from the project proposal to someone but also the task of reviewing the associated result. WPL have to factor in the time needed for the review process, testing (in case of code) and documentation in their planning of person month. They ensure that their staff work not only on the actual, publishable result but also on testing and documentation. WPL are to mentor quality reviews. This includes overseeing that quality reviews happen regularly and that checklists and guidelines are followed. They propose a result for publication only after it went through an internal review process and was checked against the corresponding Definition of Done. WPL have to define the DoD of their WP. Doing this, they have to be in close contact with the project manager and WPL of related WPs.

Individual project members have a central role in the quality management process in this project. They are to inform themselves about relevant DoD and work towards them continuously. This includes but is not limited to consistently using common project structures and templates. In addition, they schedule in time to document and test their task and for the review cycle. They are to demand time for this if necessary.

7.7.4 Supporting tools

GitLab Software development and file storage

GitLab is the central reference point for issues, templates and checklists (cf. Section 7.4).

OpenProject Project management software

All deliverables and milestones are listed in OpenProject and assigned to a person in charge. (cf. Section 7.4).

7.7.5 Processes

Creating a document There are templates, checklists and workflows for writing documents such as reports and presentations in GitLab (cf. Section 7.4).

- Read the checklist and contained workflow in its entirety, the part specific for authors and the part for reviewers.
- Follow the workflow thoroughly and keep the checklist in mind when creating the document. This way, the guidelines for document organization, content and readability can be taken into account directly during the writing process. It is imperative to use the LaTeX template provided in the "decice-management" GitLab repository.
- Send the document for review and address any open issues.
- Once the document has been finalized, the TeX file must be uploaded into GitLab with a file name following our file naming scheme, consisting of the date in YYYY-MM-DD format and the deliverable number. The project coordinator should be informed of the upload.

Software development For each deliverable result, a Definition of Done has been uploaded to GitLab (cf. Section 7.4). All code that has been written has to be checked against the respective DoD.

- Each field and programming language has its own syntax, naming and documentation conventions. As far as this project is concerned, they are taken as standard and baseline for our work. For example, PEP8 is regarded as the state-of-the-art style guide for Python and hence must be followed when writing python code.
- Tests such as unit tests, systems tests and integration tests are defined at the level of the individual work packages.
- Write engineering documentation covering features and functionality of the code, details for maintenance and a troubleshooting section.
- Divide tasks into individual subtasks, this allows for multiple members to work in parallel. The priority should be to produce correct, well-documented and tested code. Only later code may be prettified, optimized and sped up (dubbed as "Make it work, make it fast, make it pretty").

Develop an efficient git workflow.

- Configure git authorship such that name and email are set correctly.
- Commit incremental, small changes. A single commit should contain only related work.
- Use atomic commit messages and provide additional information in a second paragraph. The first line should not exceed 50 characters and continue the following sentence *“If applied, this commit will. . .”*.
- Use multiple branches. While the exact setup may be adjusted in individual work packages, the following should serve as a baseline:
 - Main branch is reserved for code ready to be deployed.
 - Dev(elop) branch is the main working branch and serves as an integration branch for features.
 - Feature branches are for developing individual features. They are to be merged into the development branch but not directly into main.
 - GitLab actions may be set up for automated tests, e.g. triggered after every push.

7.8 Risk management

7.8.1 Objective

The goal is to provide a common approach on how to handle, mitigate and report risks or possible risks to ensure that the project can succeed. Therefore, this section provides information about risk management including identification, assessment and control of risks.

7.8.2 Approach

Risk management follows the Prince2 approach and uses according terminology and functionalities. Main sources for this section are [EMPa] and [EMPb]. Risks are managed utilizing the GitLab repository “decice-management”. Risks are discussed and treated as issues in the GitLab. Utilizing the GitLab feature ensures that the history and decision with regard to handling risks is preserved and comprehensible. In addition, processing risks is easy and straight forward, as GitLab is used thoroughly during the duration of the project and no further tools are required. A corresponding risk summary matrix is set up in the wiki of the decice-management repository (Risk Summary Matrix)

7.8.3 Responsibilities

Responsibilities for the aspect *risk* of the projects are:

- Responsible: project coordinator
- Owner: project office

Every member of the project can identify risks and is liable to report them.

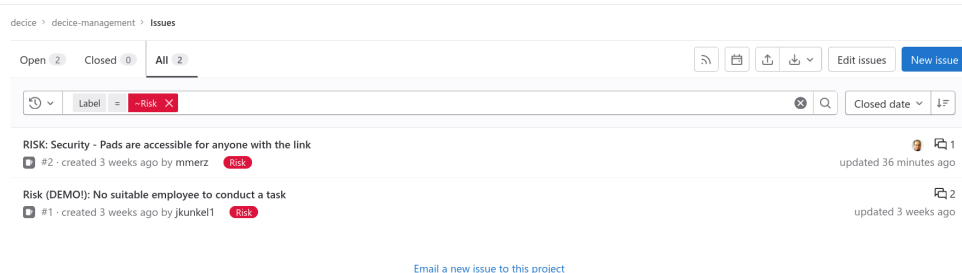


Figure 1: View risk register

7.8.4 Supporting tools

GitLab Software development and file storage

GitLab is the reference point for issues, and wiki (cf. Section 7.4).

7.8.5 Processes

View risk register and risks Everyone can view the risk register, e.g. GitLab issues labeled “risk” and the risk summary matrix.

1. In the decide-management repository, navigate to “Issues” in the sidebar
2. Select the “all” tab
3. Click on the filter line. A drop-down menu opens.
4. Select “Label” in the drop-down menu.
5. Type “Risk” in the text input field next to “Label” → the filtered list will show all issues that include a risk label (e.g. Figure 1).
6. Any risk listed can be viewed by simply following (clicking) the according link in the list.

Identify and describe a risk A risk is an uncertain event that will impact the outcome of parts or the whole project if it occurs. The impact can be positive or negative, meaning that a risk is either a threat or an opportunity.

If a risk is identified, the project member should:

1. Check if the risk is already known and reported by checking the risk register (cf. “View risk register and risks” in section 7.8.5). If the risk has not been covered, proceed with the following steps:
2. Write a summary which contains a description of the risk, the context in which it can appear, an estimate of its likelihood and the possible impact.
3. Report the risk (cf. “Reporting a risk” in 7.8.5)

Reporting a risk Every member can report a risk.

1. Identify and describe a possible risk (cf. Section 7.8.5)
2. Open an issue with a corresponding label in the decice-management repository (Select “New issue” in the issues list)
3. Fill in the form:
 - (a) Title: select a unique and clear title of the risk starting with: “Risk: ”
 - (b) Type: select “issue” as type
 - (c) Select the risk template for “Description: Choose Template”
 - (d) Description: fill in the summary created during risk identification, use the following template:


```
# Risk description:
# Affected WPs:
# Likelihood:
# Severity:
# Mitigation strategy:
```

Details are described below.
 - (e) Label: select “Risk”
 - (f) Assignee: unassigned.
4. Create the issue for the risk by pressing “Create issue”

If a risk is critical to short term operation, the responsible person (i.e. the project coordinator) has to be informed immediately. The notification has to contain a link to the newly created issue.

Details for filling out the form The project member reporting a risk (“reporter” in the following) should try to fill out the form as thoroughly as possible. Leaving out information is possible but should be avoided as the reporter should already have an idea about the details and is likely an expert in this field. Being as detailed as possible allows for an apt and fast assessment of the reported risk. Description of the fields to fill out:

- Risk description: A detailed description of the risk. What is the risk? What is the possible impact? In which context may a risk emerge?
- Affected WP: Which WPs are affected?
- Likelihood: Estimates the probability of the risk to emerge: *Very High, High, Medium, Low, Very Low*
- Severity: Describes the impact of the risk if it occurs: *Very High, High, Medium, Low, Very Low*
- Mitigation strategy: Describe how to response to a risk. Mitigation falls into different response categories (cf. Section 7.8.5). Select one of the categories first and then describe what to do.

New Issue

Title (required)

RISK: Security - Pads are accessible for anyone with the link

Type

Issue

Description

risk

WritePreview

BBIU~~S~~≡<>🔗⋮⋮⋮📎📅🖨️↻

Risk description:
The pads and therefore all information is available for everyone with a link to the pad. Data/Content in the pads can be modified arbitrarily
This applies also to contributors who left the project.

Affected WPs:
All

Likelihood:
Very High

Severity:
Very Low

Mitigation strategy:
Accept - the project and related information is funded by public money, is public and has a limited runtime. The severity is very low, as the pads are used to allow fast, frictionless communication but the information is always transferred to the gitlab repo, e.g. the according, wiki as defined in the project communication. Access to the repo is restricted by standard access control - only members of the team can access it. Illegal data/information loss/manipulation is therefor not expected.

Supports Markdown. For quick actions, type

☐ This issue is confidential and should only be visible to team members with at least Reporter access.

Assignee

Unassigned

Assign to me

Due date

Select due date

Milestone

Select milestone

Labels

Risk

Create issueCancel

Figure 2: Example of creating a risk in GitLab

Response Categories This section is taken as is from <https://prince2.wiki/theme/risk/> section “Responses to Threats” and “Responses to Opportunities”.

Threads Avoid Objective: Take action so the threat no longer has impact or can no longer happen.

Reduce Here, actions are taken to reduce the probability of the risk. Reduce the impact if the risk does occur. To help understand this, I will give an example of both reduce probability and reduce impact. Reduce response is the most common way of dealing with risks.

Fallback Fallback is also referred to as contingency. See fallback as a fallback plan of actions that would be done if the risk occurs and would become an issue. These actions would help to reduce the impact of the threat.

Transfer Here you can transfer the financial risk to another party. For example, using an insurance policy, you could recover the costs if the threat does happen.

Accept Here, a decision is taken to accept the risk. It just may cost too much money to do something about it or it may not be possible to do anything about it. However, you do keep the status of this risk open and continue to monitor it.

Share Share is both a response for threats and opportunities. Share is very common in customer/supplier projects where both parties share the gain if the costs are less than the planned costs, and share the loss if the costs are exceeded.

Opportunities Share of threats responses. I already covered “Share” when discussing the planning responses to threats. It’s where you share the profits and losses with another party.

Exploit Exploit is where, if the risk does happen, you would take advantage of it and use it.

Enhance Enhance is where you take actions to improve the likelihood of the event occurring and you enhance the impact if the opportunity should occur. This is not the same as “Exploit,” but doing certain things will give a greater chance for the opportunity to happen.

Reject This is where you identify an opportunity and decide not to take any action on this opportunity. There can be many reasons not to do this. For example, it could cause you to lose focus on your main objective, or the return on this opportunity could be low.

Assessment of risks Risk assessment is performed on a regular basis. Risks that are likely to occur are an exception. They have to be assessed more frequently. Newly filed, highly critical risks must be dealt with immediately.

For the assessment, a meeting of the project office and selected members is scheduled on a regular basis. Together, they form the “Change Control Board” (CCB).

During a meeting of the CCB the members have to:

1. Check the decision-management repository for **open** issues labelled “Risk” and assignee “None” (cf. Figure 3).
2. For every issue stating a possible risk:
 - (a) Understand the description/context of the risk

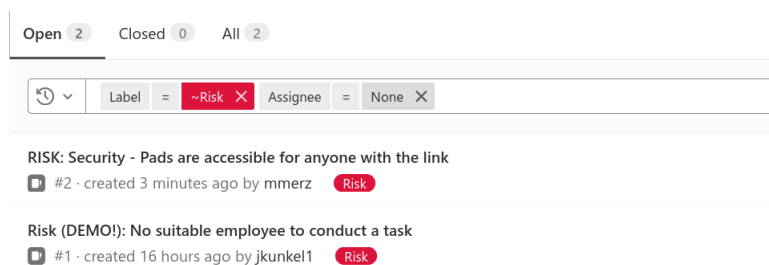


Figure 3: List of risks to asses with filter

Probability/Impact	Very Low	Low	Medium	High	Very High
Very High	2			1254	234 777
High				4563	4563
Medium					3456
Low					
Very Low					

Figure 4: Risk Summary Matrix – risk 2 has been added. Red risks are critical to the project.

(b) Assess all fields filled out (cf. Section 7.8.5) and contact the reporter to clarify any open questions.

3. Decide how to handle a risk

- Schedule a meeting to gather experts
 - Decide on preventive counter measurements
 - Decide on the reaction if the risk occurs
 - Decide who is responsible for handling the risk and who is to be informed
 - Update the description of the issue in the risk issue to contain that information
 - Update the responsible person and response type in the risk register
4. Assign the risk in GitLab to the risk owner by selecting the assignee in the “edit mode” in the issues “Assignee” section (cf. red eclipse in Figure 5).
 5. Add the risk using the data from the issue (issue number, title, priority, impact, response type, responsible person) to the risk register.
 6. Add risk with issue number to the “Summary Risk Matrix” in the wiki page Risk Summary Matrix) → risk (cf. Figure 4)

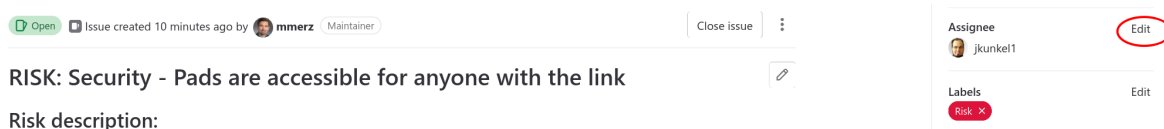


Figure 5: Edit assignee

All actions should be noted in the issues activities starting with the line "CCB Risk - << List Names of the people involved >>"

Re-assessment of risks The risk register has to be re-assessed, e.g. checked and cleaned up, on a regular basis.

For every risk in the issue list with state "open":

1. Check if still valid or applicable. If the risk is no longer valid, the underlying decision has to be explained in a comment and the risk issue can be closed.
2. Check, update and change as necessary
 - Severity
 - Likelihood
 - Affected WPs
 - Mitigation strategy
 - Assignee (risk owner)
3. Every change has to be explained in a comment to keep track of the changes and why they happened. One comment for all changes is sufficient.
4. Changes have to be promoted to the risk summary matrix and the risk register to avoid deviation of sources!

Initializing the risk register The project coordinator creates the wiki page for the table of risk (risk register) at the beginning of the project.

All members may start the "Reporting a Risk" process.

Exporting risks To export the risks as CSV to be used in the risk register, reports etc.:

1. Follow instructions how to view the risk register (cf. Section 7.8.5)
2. Select the option "Export as CSV" on the top button row (cf. Figure 6).
3. Confirm the dialogue by clicking on "Export issues". An email with the CSV attached will be sent out.

Controlling of risks

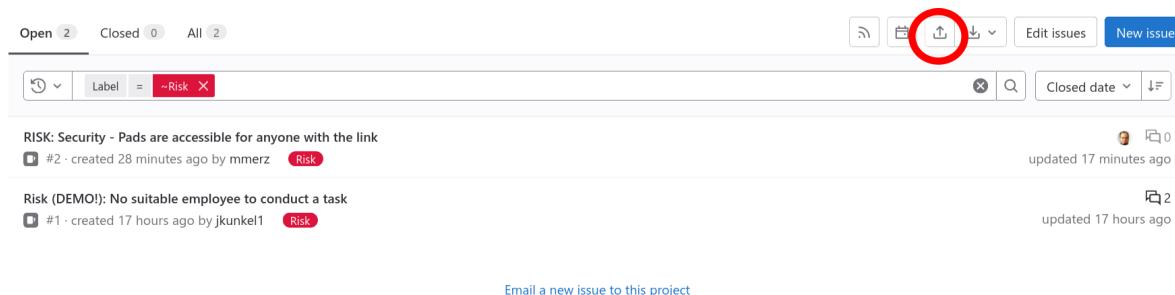


Figure 6: Export the listed risks by “Export as CSV” option (red circle)

Report risk event If a risk event occurs, the person responsible for the risk (the Assignee) looks up in the issue description whom to inform and what to do and informs the relevant persons. All actions taken must be reported in the form of a comment in the issue.

Handling a risk event The Assignee starts all actions defined in the issue description, if he/she has the capabilities to do so. All persons informed in the risk reporting step detailed above follow the measures defined in the issue description. All actions taken must be reported in the form of a comment in the issue.

7.9 Governance

7.9.1 Objective

Project governance defines how decisions are made and which responsibilities, policies and processes are applied to increase the chance of succeeding. Laying out governance structures ensures that responsibilities and processes are clearly defined and enforced. Project members have clearly defined processes they can follow and rely on. By providing them with the necessary tools and processes, they are empowered to handle specific topics and to participate in the decision process.

7.9.2 Approach

The project is managed by an exception based approach. An exception arises whenever constraints of a phase or the project are broken or tolerances are exceeded. Constraints and tolerances include, but are not limited to, the timeframe, risks and scope of a (sub-)task. Thus, whenever a situation arises that exceeds the competence level or responsibility of any given project member, it is reported as an exception to the next management level. For example, as soon as an impending decision affects the work package as a whole, the WPL must be involved and the topic must be discussed in the next regular (typically monthly) WP meeting. The other WPL should be informed in turn in the next monthly WPL meeting. The decision should be recorded in writing immediately, e.g. in a task-related GitLab issue. For faster dissemination, the decision should be announced in RocketChat as well.

The escalation approach ensures that the leads of work packages can operate freely and handle arising challenges within given limits regarding quality aspects, risk and scope by themselves. Thus, most issues are dealt with within the field of expertise for a specific topic or work package. This

reduces the load on higher management levels and utilizes domain specific expertise optimally at the same time.

Exceptions can be passed up through the various levels of management. By not skipping any level, it is possible to ensure that all levels are involved.

The following aspects may be subject to exceptions:

- **Scope** Scope or functionality of a deliverable
- **Time** Timeframe to deliver a deliverable
- **Effort** Work invested into a deliverable
- **Quality** Expected functionality and quality of the deliverable
- **Risk** Manifestation of an impending risk or change in a risk attribute, e.g. its likelihood

Further aspects of governance, e.g. risk management or communication, have been handled in their according section of this document.

7.9.3 Responsibilities

Every member has to report an exception immediately as soon as they become aware of it.

Roles involved in exception handling are:

- Work package leaders
- Project coordinator
- Advisory board
- Granting authority

The user perspective is incorporated through interaction with (scientific) community.

For the present project, the management lies with the project coordinator.

Leaders of different work packages are defined in the project proposal. Sub-tasks may have been delegated internally to designated task leaders.

7.9.4 Supporting tools

GitLab Software development and file storage

GitLab is the reference point for issues (cf. Section 7.4).

Mailing lists Mailing lists allow for targeted communication with different interest groups (cf. Section 7.4).

OpenProject Project management software

All deliverables and milestones are listed in OpenProject. (cf. Section 7.4).

RocketChat Messaging service (cf. Section 7.4).
Messaging allows for direct, low-threshold communication.

7.9.5 Processes

Defining tolerance criteria In order to know when to raise an exception, each deliverable needs to be comprehensibly defined. This is done in a meeting between all directly and indirectly (e.g. depended on the result) involved parties.

- **Scope** Expected functionality of the deliverable
- **Dependencies** Dependencies on other deliverables and deliverables dependent on this result
- **Quality** When can the deliverable be considered done according to requirements (cf. Section 7.7) and tolerable deviations from plan
- **Time** Due date of the deliverable
- **Risk** Risks to be taken into account (cf. Section 7.8).

Discussing the above defined criteria in a joint meeting ensures a common understanding of all parties involved.

In order to define the tolerances:

1. The work package leader responsible for a given deliverable collects basic information regarding scope, dependencies etc. of the deliverable from the project proposal.
2. WPL organizes and hosts a meeting (cf. Section 7.4) with all involved partners and the manager(s) above his/her level of competence.
3. During the meeting, an agreement on the above named attributes has to be reached and logged.
4. Risks and quality criteria are handled according to the according sections.
5. The remaining, newly defined attributes of the deliverable and links to the related risk and quality management issues are added to the deliverable description in OpenProject.

Identifying an exception With tolerance criteria being clarified, identifying when to report an exception is straightforward. If one of the defined tolerance criteria is exceeded or it is foreseeable that it will be exceeded, an exception has occurred. An exception to this rule are risk criteria. Here, an exception occurs if the risk becomes an event or if a risk attribute, for example the likelihood of the risk to occur, changes. In the latter case, the next management level has to be informed to ensure that the risks are re-assessed. As soon as an exception is identified, it has to be reported.

Reporting an exception Every project member can report an exception. As soon the pre-defined tolerances of a task are exceeded or constraints are about to be broken, the project member first noticing it has to report to the next management level. Usually, this should be the task or work package leader. The person noticing the exception has to:

1. Create a GitLab issue in the repository of the affected work package containing an exception report. The report has to define the context, e.g. affected work package, task and deliverable, the actual exception with its reason and cause as well as possible measures how to handle the problem.
2. Assign the newly created issue to the corresponding work package leader and labels the issue with the “Exception” label.
3. Informs the person on the next management level about the exception via both mail and chat. The subject should start with “EXCEPTION: WP X task Y” and the message has to contain the link to the newly created issue.

Handling an exception The person receiving an exception report has to immediately assess the report and decide how to continue.

1. Discuss the report with the reporter to clarify possible open questions and to adjust or extend the report if necessary.
2. Depending on whether the issue can be decided within his/her responsibility, the further procedure differs. If the exception falls within the level of competences of the person receiving it, he/she continues solving the exception. Should this require further coordination, the person receiving an exception organizes a meeting with all involved parties to discuss and decide on the course of action. Should the exception exceed the level of competence of the original receiver, he/she reports it to the next higher management level.
3. Add any findings and decisions reached with regard to handling the exception as a comment to the original report.
4. Should the original exception lead to another, downstream exception, the person receiving the original exception reports the resulting one (cf. Section 7.9.5) and links it to the original exception.
5. The person responsible for handling the exception, whether this was the original receiver or someone on a higher management level implements the agreed upon course of action. All countermeasures taken are logged as further comments to the GitLab issue.
6. As soon as the problem is solved, the responsible person closes the issue.

Viewing exceptions Everyone can view the exception register, e.g. GitLab issues with label “Exception”.

1. In the decice-management repository, navigate to “Issues” in the sidebar

2. Select the “all” tab
3. Click on the filter line. A drop-down menu opens.
4. Select “Label” in the drop-down menu.
5. Type “Exception” in the text input field next to “Label” → the list shows all exceptions.
6. Any exception listed can be viewed by simply following (clicking) the according link in the list.

7.10 Reporting

7.10.1 Objective

Reports describe and summarize the current state of the project, i.e. its progress, problems and challenges. They provide a high-level overview of the project’s status. Thus, reports are crucial in bringing the different project partners and the project coordinator up to date with the latest developments in individual working groups. Allowing the project coordinator and the granting authority to regularly assess the project, reports are the foundation to make informed decisions on the future direction of the project.

7.10.2 Approach

The reporting system is divided into multiple levels. Some types of reports are initially of a purely internal nature and primarily serve to keep the project partners up to date with each other’s progress. These reports include regular status reports as well as reports on problems that arise or become apparent, such as risk and exception reports. We consider the latter to be of such a high importance that we dedicate separate chapters to each of these (cf. Section 7.8 and Section 7.9). The aforementioned reports ultimately cumulate into the reports to be made to the granting authority.

7.10.3 Responsibilities

Reporting follows a bottom up approach. Each project member reports on his/her progress and on foreseeable deviations to the project plan in their respective area of operations. This information is merged at the level of work package leaders, which in turn report to each other and the project coordinator. Work package leaders not only distribute individual tasks to their staff (cf. Section 7.6), but also report the status of their work package for a specific period or project phase to the project coordinator.

Each project partner must prepare and submit regularly the financial reports for its own institute to the granting authority. This also includes reporting of working times. The project office coordinates and supervises this activity and supports it by providing templates and guidance.

The project coordinator administers finances and distributes them according to the project plan. Furthermore, the project coordinator oversees both financial and content reports and submits them to the granting authority.

7.10.4 Supporting tools

GitLab Software development and file storage

GitLab is the reference point for issues, templates and checklists (cf. Section 7.4).

OpenProject Project management software

All deliverables and milestones are listed in OpenProject and assigned to a person in charge. (cf. Section 7.4).

7.10.5 Processes

Financial reporting The financial reporting is a continuous process and is done by each partner individually. Before a report is being handed in a coordinator is obliged to verify the integrity and completeness of each submission. The first report is due 18 months after the initial project start, internal financial reportings are mandatory every 6 months stating expenses per month as this is relevant for budget transfers. The EU portal requires total cost per WP. This includes personnel and material costs. If the latter is more than 15% above personnel costs, a breakdown of costs into individual items such as travel expenses have to be provided. A template with an appropriate level of cost breakdown will be provided.

Financing The DECICE project and its members will receive a financial coverage of up to 90% after the first financial report (18 month after the initial start) given the project is on track and interim evaluation is passed successfully. The remaining 10% of the budget will be transferred after the project has ended and the final report has been completed, which might take up to one year after project end. Budget shifts are possible in principle and without amendment if project members run into financial difficulties. Partners are entitled to transfer money directly between each other and no intermediary step that includes UGOE is necessary. An amendment is needed only for changes of content.

Reporting working times Every member of the project is required to report their working hours for documentation and liability purposes. A template with an appropriate level of detail for documentation will be provided.

Interim and final reports Cf. Section 7.6

Status reports Status reports are either done regularly based on a schedule or negotiated to cover a specific phase, which tries to achieve a specific goal.

Reports should contain which tasks have been completed, which remain open, if issues were encountered and how they were fixed as well as any open issues including potential solutions.

Reports are stored in the according report section of the decice-management repository.

Exception reports Cf. Section 7.9

Risk reports Cf. Section 7.8

8 References

- [EMPa] EMPH Group BV. *PRINCE2 wiki Risk*. URL: <https://prince2.wiki/theme/risk/> (visited on 01/31/2023).
- [EMPb] EMPH Group BV. *PRINCE2 wiki Risk Management*. URL: <https://prince2.wiki/management-products/risk-management-approach/> (visited on 01/31/2023).