D3.7 – i4Q GUIDELINES FOR BUILDING DATA REPOSITORIES FOR INDUSTRY 4.0

WP3 – BUILD: Manufacturing Data Quality

i4Q has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 958205.

Document Information

GRANT AGREEMENT NUMBER	958205	ACRONYM	i4Q
FULL TITLE	Industrial Data Services for Quality Control in Smart Manufacturing		
START DATE	01-01-2021	DURATION	36 months
PROJECT URL	https://www.i4q-project.eu/		
DELIVERABLE	D3.7 – i4Q Guidelines for Building Data Repositories for Industry 4.0		
WORK PACKAGE	WP3 – BUILD: Manufacturing Data Quality		
DATE OF DELIVERY	CONTRACTUAL: June 2022	ACTUAL	June 2022
NATURE	Report	DISSEMINATION LEVEL	Public
LEAD BENEFICIARY	ITI		
RESPONSIBLE AUTHOR	Jordi Arjona (ITI), Santiago Gálvez (ITI), Emili Miedes (ITI), Sonia Santiago (ITI)		
CONTRIBUTIONS FROM	1-CERTH, 2-ENG, 5-KBZ, 11-UNI,		
TARGET AUDIENCE	1) i4Q Project partners; 2) industrial community; 3) other H2020 funded projects; 4) scientific community		
DELIVERABLE CONTEXT/ DEPENDENCIES	This document is a public report developed within the context of "Task 3.5 - Manufacturing Data Storage and Use", that presents a guide for building Data Repositories for Industry 4.0, which corresponds to the i4Q^{DRG} solution. This document has no preceding documents but will have a next version by M24, namely *D3.15 – Guidelines for building Data Repositories for Industry 4.0 v2*. It also provides feedback to deliverable *D3.8 - i4Q Data Repository*.		
EXTERNAL ANNEXES/ SUPPORTING DOCUMENTS	Appendix I of this deliverable provides further details on a questionnary distributed to pilots and solutions providers to gather specific and technical requirements. Appendix II of this document contains the PDF version of the web i4Q^{DRG} documentation.		
READING NOTES	None		

ABSTRACT	This deliverable presents an overview of the role and importance of data repositories in Industry 4.0 contexts, such as this project. Furthermore, this document explains the challenges and requirements arising when developing data repositories and provides some recommendations on how to address them, using the i4Q^{DR} as an illustrative example.

Document History

VERSION	ISSUE DATE	STAGE	DESCRIPTION	CONTRIBUTOR
0.1	13-Apr-2022	ToC	First Version of Table of Contents	ITI
0.2	30-May-2022	1st Draft	Version submitted for internal review	ITI
0.3	13-Jun-2022	Internal Review	Internal review	FACTOR, KBZ
0.4	20-Jun-2022	2nd Draft	Address comments by reviewers	ITI
1.0	30-Jun-2022	Final Draft	Final quality check and issue of final document	CERTH

Disclaimer

Copyright message

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS/ACRONYMS

ANSI	American National Standards Institute
API	Application Programming Interface
BDA	Big Data Analytics
CNC	Computer Numerical Control
CRUD	Create, Read, Update, and Delete
DR	Data Repository
DRG	Data Repository Guidelines
HA	High Availability
HA+Sec	High Availability with Security
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol (Secure)
IM	Infrastructure Monitoring
IT/OT	Information Technology/Operational Technology
JSON	JavaScript Object Notation
OS	Operating System
PA	Perspective Analysis
PDF	Portable Document Format
PQ	Process Qualification
REST	Representational State Transfer
REST API	RESTful Application Programming Interface
SQL	Structured Query Language
SS	Single Server
SS+Sec	Single Server with Security
TCP	Transfer Control Protocol
TLS	Transport Layer Security

Executive summary

D3.7 is the first version of a guide devoted to the building of Data Repositories for Industry 4.0, which also tackles the estimation of storage capabilities and the building of technological systems that provide easy ways to access manufacturing data and to communicate with industrial platform components and microservice applications**.** These guidelines correspond to the **i4Q Data Repository Guideline** (i4Q^{DRG}) solution.

In this document, firstly the role of data repositories in Industry 4.0 contexts is explained and is presented the importance of the requirements they fulfil. Using illustrative examples, this role explanation is matched with the **i4Q** project and its solutions, especially with the **i4Q Data Repository** (i4Q^{DR}) solution.

Furthermore, this deliverable explains the challenges arising when building a data repository for Industry 4.0 and provides an overview on how to approach its development by gathering a set of recommendations addressing these challenges, focusing especially on the analysis and design phases, but also covering some aspects related to the implementation phase.

These guidelines have driven the development of the **i4Q Data Repository** (i4Q^{DR}) Solution, which is presented in "*D3.8 - i4Q Data Repository*". Furthermore, this deliverable will have a second version, at M24, namely *D3.15 – Guidelines for building Data Repositories for Industry 4.0 v2.*

Document structure

Section 1: provides an overview on the features provided by data repositories in Industry 4.0 contexts, explaining the key role they play. This explanation is illustrated using the i4Q project and its solutions (especially the i4Q^{DR}) as running examples.

Section 2: gathers some of the most important challenges and requirements arising when developing data repositories for Industry 4.0. Moreover, it proposes some recommendations and procedures addressing them.

Section 3: explains in detail how the recommendations and guidelines provided in Section 2 have been applied to define the design and the development *strategy* of the i4Q^{DR}. Note, however, that this section does not provide specific details on the implementation of the i4Q^{DR}. This issue is out of the scope of this deliverable but will be covered by "*D3.8 - i4Q Data Repository*" (due on M18).

Section 4: provides the conclusions of this document, summarising its main contributions and providing a brief explanation on the future version of this deliverable.

Appendix I: provides information on a questionnaire used to gather specific needs from pilots and other i4Q solutions to must be covered by the i4Q^{DR}.

Appendix II: includes the PDF version of the i**4Q** **Data Repository Guideline** (i4Q^{DRG}) web documentation, which can be accessed online at: **http://i4q.upv.es/7_i4Q_DRG/index.html**

1. Introduction

One of the main characteristics of the so-called industry 4.0 is the application of technical solutions to retrieve data from industrial processes so that it can be analysed and processed afterwards. The goal of such exercise is to obtain relevant knowledge, that can be applied to improve these industrial processes and make them more efficient. For instance, by reducing the percentage of defects, or by detecting them in earlier stages of the production process, which typically reduces the economic costs of those defects.

In a typical Industry 4.0 scenario, most part of the data is *generated* by manufacturing devices acting as *sensors* and is dumped into a storage technology or tool. Then, there are several *data analysis and processing tools* that consume this data for different purposes. For instance, there are tools that perform data cleaning processes to remove useless cases, or to extract only the parts of data that really matters for a specific purpose. Other tools, such as dashboards, show the stored information in a more graphical and intuitive way, to facilitate some decision-making processes. Other important data consumers are big data analytics tools, that apply different algorithms to the data and allow, for example, to discover correlations and relationships among different variables of the analysed data.

In the case of the i4Q project, the *i4Q Data Repository* (i4Q^{DR}) is the solution covering all the data storage and retrieval requirements of the other solutions, which is presented in more detail in deliverable D3.8. Then, there are many other i4Q solutions that fall into the category of *data analysis and processing* tools mentioned, such as the i4Q Data Integration and Transformation Services (i4Q^{DIT}), the i4Q Services for Data Analytics (i4Q^{DA}), and i4Q Big Data Analytics Suite (i4Q^{BDA}), which are presented in deliverables D4.1, D4.2 and D4.3, respectively. These solutions take data previously stored in the i4Q^{DR} as input and produce new one that can be stored into it as well. Another source of data for the i4Q^{DR} is the i4Q Trusted Networks with Wireless and Wired Industrial Interfaces (i4Q^{TN}) solution (see deliverable D3.4), which oversees collecting data generated by actuators, sensors, and other IT/OT systems and give it as input for the i4Q^{DR} and the i4Q Analytics Dashboard (i4Q^{AD}) solution (described in deliverable D4.4). Figure 1 shows an overview of connection and interactions of the i4Q^{DR}, including other i4Q solutions. More specifically, elements in the box labelled as *"Takes input from"* are the ones requiring the storage

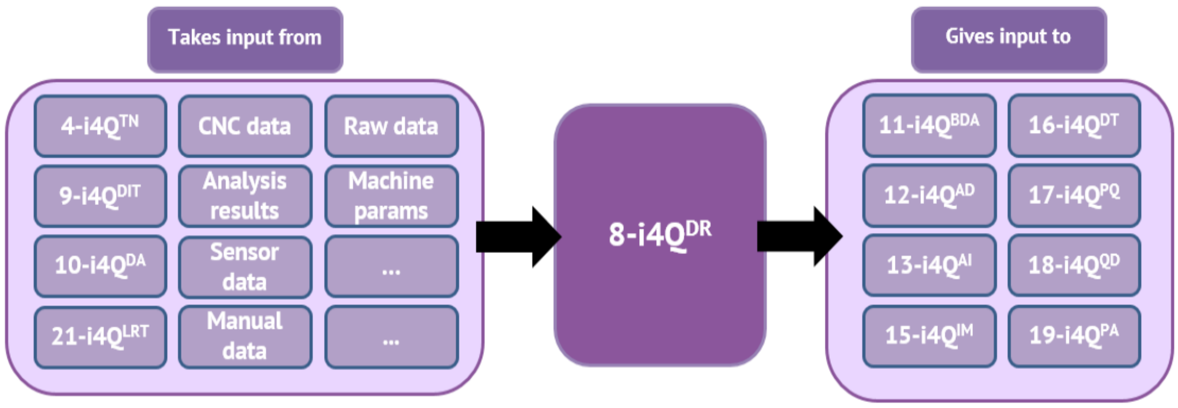


Figure 1. Connections and interactions of i4Q^{DR}

of data into the i4Q^{DR}, whereas the ones located within the box labelled as *"Gives input to"* correspond to solutions retrieving data from the i4Q^{DR}.

Note that the i4Q^{DR} in some cases can be the final destination of the stored data, but in some other cases, the stored data will be provided as input for another solution that will further process it. This is well illustrated by the pipelines designed for the i4Q pilots. For instance, in Pilot 1 (see pipeline in Figure 3) data stored into the i4Q^{DR} will not be retrieved by any other solution, whereas in Pilot 3 (see pipeline in Figure 2) such data will be sent via the Message Broker to other solutions, and directly retrieved by the i4Q Prescriptive Analysis Tools (i4Q^{PA})[1]. As discussed above, storing, and retrieving data is a basic but crucial feature in an Industry 4.0 scenario. Therefore, the development of technical solutions providing that functionality, such as the i4Q^{DR}, must be addressed very carefully so that this feature is provided in the most appropriate way for each different component that interacts with it.

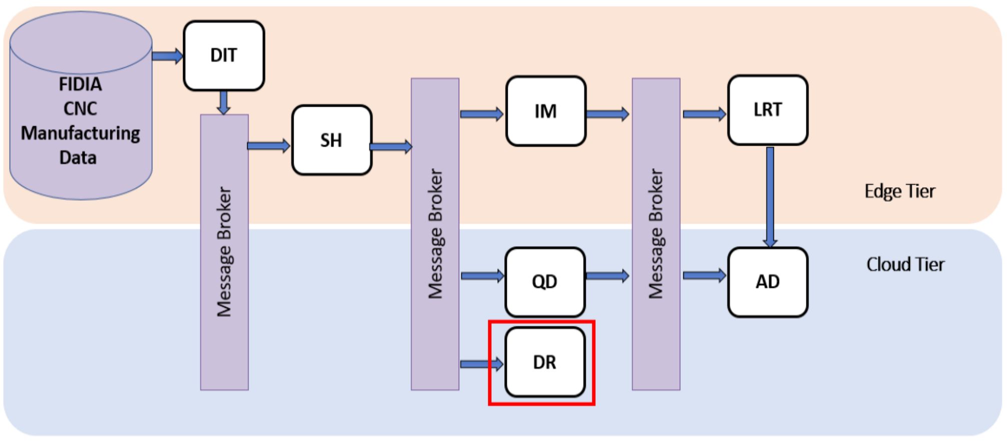


Figure 3. Pipeline of Pilot 1 (FIDIA)

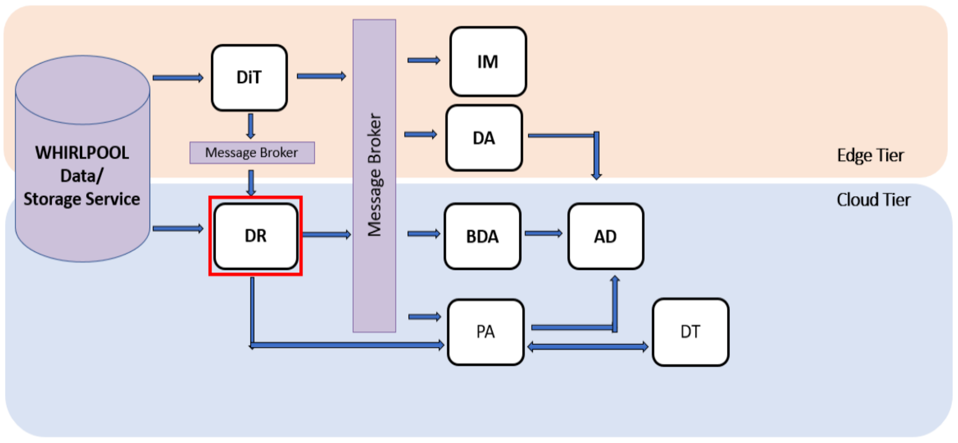


Figure 2. Pipeline of Pilot 3 (Whirlpool)

[1] In both, Figure 3 and in Figure 2, solutions are denoted by the acronyms of the solution's code without the word "i4Q". For example, the i4Q Data Repository, whose code is i4Q^{DR}, is denoted by "DR".

Indeed, the interaction with other solutions is the main challenge for the development of the i4Q^{DR}. In this sense, some complexity arises just due to the number of different solutions interacting with the i4Q^{DR}. However, the highest degree of complexity is due to the fact that solutions retrieving or storing data from or into the i4Q^{DR} may need different functionalities from it, and, specially, have different technical requirements. For instance, some solutions may only need to store data, whereas other may need importing/exporting data from/to other data sources. Moreover, some solutions may require an SQL-style relational database, while others may require storing files.

This document provides the first version of the i4Q Data Repository Guidelines solution (i4Q^{DRG}), a guide devoted to the building of Data Repositories for Industry 4.0, which also tackles the estimation of storage capabilities and the building of technological systems that provide easy ways to access industry data and to communicate with industrial platform components and microservice applications. More specifically, this document focuses on the challenges that may arise during the development of a data repository, and some recommendations to address them.

The recommendations gathered in this document have guided the development of the i4Q^{DR} so far, up to M18. Therefore, this solution will be used as an example to illustrate some of the concepts and recommendations explained in this document.

2. Guideline on how to tackle data repositories for Industry 4.0

Data repositories may be thought of as low-complex technical solutions, since they provide a basic functionality, namely the storage and retrieval of data. However, its development is quite challenging since, as explained in Section 1, they usually interact with many other components with different purposes. Consequently, in this context, the requirements of a data repository, especially the technical, are quite *heterogenous,* since the ones for a given component can be quite different from the ones for another component.

In this sense, a good example is the type of storage technology, which will vary depending on the type of data to be stored. However, it could also be the case that two components have contradictory requirements. For instance, some components running on premises with constrained resources will require light mechanisms to store just a few bytes of data, whereas other components may require more complex and powerful mechanisms to store large amounts of data.

Therefore, the development of data repositories in that context requires a balance between two somehow opposite concepts. In the one hand, data repositories need to have a *flexible design*, so that they can fulfil heterogenous requirements coming from different dependent solutions. In the other hand, to reduce the complexity of the development, and the maintenance of the data repository once it is deployed, it is necessary to *provide* the required *functionality* in the *most centralised fashion possible*, by extracting as many common functionalities as possible and implement them in the most harmonised possible way. For instance, a data repository should try to offer only one interface to receive data from other solutions to store it, independently of the type of data. In this way, there is only one implementation for that feature, instead of one per storage technology.

The first step to address these challenges when developing a data repository is to identify which components interact with it, either to provide data that needs to be stored, or to retrieve data that has previously been stored. Then, for each solution interacting with the data repository, it is necessary to gather the specific needs regarding several aspects, such as:

- The functionality required (e.g., only CRUD, or something else).
- The volume of data.
- Possible performance constraints, such as the expected number of read or write operations per second.
- The operation mode. For instance, whether the data will be provided in an interactive mode, or in batch, and if will be done on-demand or as a scheduled task.
- The need of replicated instances of the data repository.
- The type of communications and expected interfaces and data formats.
- Who and what components are authorised to access the data repository.

This information must be carefully analysed to make the decisions that will define the main aspects of the data repository's design, namely:

- The most appropriate tools and technologies for the type of data that will be managed.
- The architecture of the data repository, which must be flexible to accommodate different requirements but, at the same time, should try to fulfil them in a centralised fashion

whenever possible, as mentioned above.

- The necessary storage configuration options considering the functional requirements and technical details such as the amount of data that will be handled, and the computing resources that will be available once the data repository is deployed.

For making these decisions it might be helpful to categorise needs and functionalities into those that are specific only to few solutions (or only one) and those that are common to several of them. Addressing the first group of needs might require using concrete tools. However, whenever possible, one should try to use tools that can work for other cases, too. Common needs and functionalities should be addressed in the most general and harmonised way. For instance, using the same technology or tool, whenever possible.

At the implementation level, the most important strategy is to simplify and reduce the complexity of the developments. This can be achieved, for instance, by providing only one implementation of a common functionality, independently of the solution requiring it. This can be the case, for instance, of access control. If the implementation differs depending on the solution the data repository is interacting with, it is recommended to analyse whether there are some sub-functionalities that can be implemented in the same way. For instance, if a solution needs to store data in a relational database, and another one needs to store a file in an object storage tool, the concrete mechanism to do so will be different in both cases. However, as mentioned above, the data repository can offer a common interface to receive the data to be stored, and then run the corresponding storage mechanism.

When different implementations for the same feature are necessary, the recommendation is to follow a similar structure and style to facilitate the identification of the functionality that is being implemented and the update of the code if changes are necessary in the future. An example of this is using similar class or method names.

Finally, another way to simplify and decrease the complexity of developments consists of reducing the number of dependencies and using tools and libraries that do not have too specific requirements. For instance, Docker is compatible with most common operating systems. Furthermore, Docker containers allow to abstract the (virtual) execution of a tool from the operating system (OS) in which Docker is running. This means that a tool running inside a Docker container executed on a Windows OS behaves *exactly the same* as when the tool is run in a Docker container executed on a Linux OS, assuming both containers have the same configuration.

3. Design specifications of the i4Q Data Repository

This section describes how the design of the i4Q^{DR} and the definition of the *strategy* for its development have been approached so far (M18), following the recommendations and guidelines provided in Section 2 to address the challenges and requirements arising when developing data repositories for Industry 4.0. Further details on the implementation of the i4Q^{DR} will be explained in deliverable "*D3.8 - i4Q Data Repository*" (due on M18).

Regarding the interactions of the i4Q^{DR} with other components, the i4Q project includes a number of pilots and solutions that manage data one way or another and thus need mechanisms to handle such data. More specifically, the i4Q^{DR}, takes as input data from other i4Q solutions as well as raw data from other industrial components (e.g. CNC data or data gathered from sensors), and even data manually provided by a human user. Furthermore, the i4Q^{DR} provides data to other i4Q solutions, such as the i4Q^{BDA}, the i4Q^{IM}, the i4Q^{PQ}, etc. As explained in Section 2, Figure 1 gathers the interaction and connections of the i4Q^{DR} with other components and solutions. Further information on this topic can be found in deliverable *"D1.9 –Requirements Analysis and Functional Specification v2"* (Section 4.8).

With respect to the features and functionalities that the i4Q^{DR} is required to provide, the main needs common to all pilots and solutions are basic functionalities like storing data and retrieving the stored data. However, there are some additional generic requirements which are quite common in any storage technology, including:

- *Data persistence*: the data is saved to a persistent storage and remains intact until it is altered or deleted on purpose.
- *Availability:* the data is available to any of its legitimate users.
- *Privacy:* the data is protected against unauthorized access.
- *Security:* the data is protected against software and hardware failures.
- *Efficiency:* with a proper use of the available resources.

Moreover, each pilot and solution have specific needs that involve a number of factors. One of the most important criteria is the *type of the data* to be stored, either relational, document-based, graph-oriented, blob-based, file-based, etc. This criterion on its turn affects other criteria. For instance, the use of enormous blobs or files may make a significant impact on the performance numbers. On the other hand, the same solution may have different needs depending on which pilot it is used. Thus, the i4Q^{DR} must be able to offer its services to its clients, under a wide range of needs, scenarios, and settings.

As a second step, needs related to the *interoperability* of the data have been identified. For instance, a pilot or solution generates data with different structures and/or syntax (e. g. relational and document-based data) or data that, for some reason, has to be handled by different storage tools. Another example may be the case of a solution that handles different sets of data and each one has different needs (formats, syntax, access patterns, performance requirements, etc.) and needs to use them in a unified way.

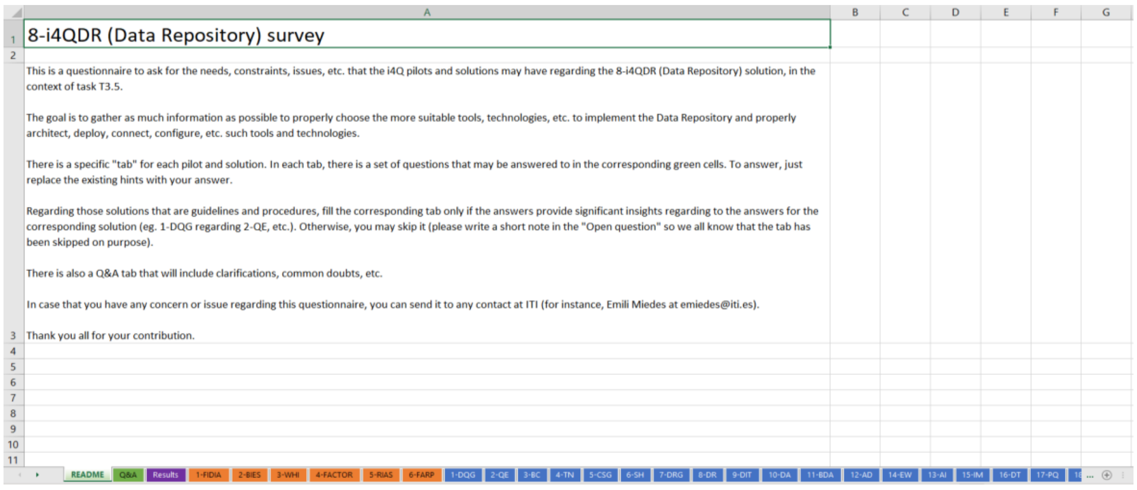
The i4Q^{DR} includes a number of off-the-shelf open-source tools. The selection of the current tools has been performed according to the needs of both the pilot and solution partners of the project. The procedure followed to produce such selection includes the following steps.

1. Prepare a questionnaire to ask about the needs a pilot or solution has regarding the i4Q^{DR}.
2. Receive the answers of the questionnaire and summarize them.
3. Identify which tools are specifically required by the partners.
4. Identify other tools that can satisfy the requirements and needs of the partners.

The questionnaire is a Microsoft Excel document available in a private repository[2] (see Appendix I for further details). As shown in Figure 4, which provides an overview of such document, it includes a page for each pilot and solution partner. These pages contain a set of questions to be answered by each partner. The questions include these issues:

- The types of storage mechanism needed by the pilot or solution (relational, structured, JSON-based, key-value-oriented, etc.).
- Any information available so far regarding the volume of the data to be handled (for instance, in orders of magnitude).
- The type of interaction that may be expected (interactive, batch, stream-oriented, etc.).
- The type of communication interfaces that may be used (HTTP, HTTPS, TCP, etc.).
- Any other requirement related to communication.
- Any requirement or need related to data replication.
- Any requirement regarding data privacy and security (access control, anonymization, etc.) that may apply.
- Any specific deployment needs if any (on bare machines, Docker, Kubernetes, etc. and also on-premises vs cloud).
- Any requirement or need regarding the performance.

[2] Survey to solution providers and pilots to gather i4Q^{DR} requirements: https://knowledgebiz.sharepoint.com/:x:/r/sites/i4Q/_layouts/15/Doc.aspx?sourcedoc=%7B4A34149D-B1D1-4146-B4EF-0E44056E200E%7D&file=Survey_8-i4QDR.xlsx&cid=79408e25-3af5-4d1e-a129-6e25abf22f3e


8-i4QDR (Data Repository) survey

This is a questionnaire to ask for the needs, constraints, issues, etc. that the i4Q pilots and solutions may have regarding the 8-i4QDR (Data Repository) solution, in the context of task T3.5.

The goal is to gather as much information as possible to properly choose the more suitable tools, technologies, etc. to implement the Data Repository and properly architect, deploy, connect, configure, etc. such tools and technologies.

There is a specific "tab" for each pilot and solution. In each tab, there is a set of questions that may be answered to in the corresponding green cells. To answer, just replace the existing hints with your answer.

Regarding those solutions that are guidelines and procedures, fill the corresponding tab only if the answers provide significant insights regarding to the answers for the corresponding solution (eg. 1-DQG regarding 2-QE, etc.). Otherwise, you may skip it (please write a short note in the "Open question" so we all know that the tab has been skipped on purpose).

There is also a Q&A tab that will include clarifications, common doubts, etc.

In case that you have any concern or issue regarding this questionnaire, you can send it to any contact at ITI (for instance, Emili Miedes at emiedes@iti.es).

Thank you all for your contribution.

Figure 4. Overview of questionnaire to gather needs to be covered by the i4Q^{DR}

After analysing the survey's answers and carefully evaluating the needs of the partners, the following tools were selected:

- *Cassandra*[3], a wide-column NoSQL distributed database server, appropriate for managing massive amounts of data.
- *MinIO*[4], which offers a high-performance, S3 compatible object storage. It is used to store files and is compatible with any public cloud.
- *MongoDB*[5], a JSON document-oriented database server.
- *MySQL*[6], a SQL relational database server.
- *Neo4J*[7], a graph database server that allows storing data relationships.
- *Redis*[8], an in-memory data structure store, used as a distributed, in-memory key–value database, cache, and message broker, with optional durability.

In addition, the following tools have been included later, to satisfy additional needs and requirements brought out by some partners:

- *MariaDB*[9], a SQL relational database server, very similar to MySQL.
- *PostgreSQL*[10], a SQL relational database server.

[3] Cassandra. https://cassandra.apache.org
[4] MinIO. https://www.min.io
[5] MongoDB. https://www.mongodb.com
[6] MySQL. https://www.mysql.com
[7] Neo4J. https://www.neo4j.com
[8] Redis. https://redis.io
[9] MariaDB. https://mariadb.org
[10] PostgreSQL. https://www.postgresql.org

Both tools offer similar features, in the sense that they are SQL relational databases. However, it is not possible to use only one of them or replace them by MySQL to have only one tool for relational data storage because they were especially requested by some pilots to facilitate the integration with other technological systems.

Moreover, according to the needs of the pilot and solution partners, it has been decided to offer a variety of configurations or *scenarios* of each tool. These scenarios can be seen as different ways of deploying the tools to meet some requirements related to aspects such as performance or security. Generally speaking, for each tool, the following scenarios are considered:

- A first basic *single server (SS)* scenario, which offers the tool in its most basic version. It leverages a single instance of the tool, ready to be used. It is worth noting that such single instances do not consider any security issue beyond its default configuration. Thus, an "SS" scenario is only recommended for the development tasks of a pilot or solution.
- A *single server with TLS security (SS+Sec)* scenario, that offers the tool with a security configuration based on the use of TLS (with x509 certificates). This scenario is suitable for production settings in which a single instance of the tool suffices to provide a secure and stable service.
- A *high availability (HA)* scenario, which offers the tool in a high availability mode. This typically involves defining some *cluster* or *replica set* environment, composed by a number of instances of the tool that work in a cooperative mode (for instance, a cluster of replicas of a database server). Again, these instances do not offer any security mechanism beyond those that are offered by default and are, thus, only recommended for development purposes.
- Finally, a high *availability with TLS security (HA+Sec)* scenario, which extends the "HA" scenario with a security configuration based on the use of TLS. This is the recommend option to offer a secure, fault-tolerant, highly available service.

For each scenario, a number of *software artifacts* are provided, including configuration files, Docker Compose[11] orchestration files, and shell-scripts for Bash. Moreover, additional common configuration files and shell scripts are provided. The purpose of such artifacts is to deploy the selected storage tools according to the features of one of the four scenarios described above. Deliverable *"D3.8 - i4Q Data Repository"* (section 3.1 "Current implementation") provides additional details about these artifacts. Moreover, it provides a summary about the status of the development of the scenarios of each tool.

Furthermore, the i4Q^{DR} includes another tool, Trino[12], which is a highly parallel and distributed ANSI SQL-compliant open-source query engine. These features enable a very efficient performance, even in the case of many simultaneous queries. Trino offers a relational-like view of a number of data storage tools including Cassandra, MariaDB, MongoDB, MySQL, PostgreSQL, Redis and others.

From an architectural perspective, Trino will be deployed as a layer on top of the different artifacts mentioned above, as illustrated in Figure 5. Note that Neo4J does not appear below the Trino

[11] Docker Compose documentation. https://docs.docker.com/compose/

[12] Trino. https://trino.io

layer because this storage tool is not currently supported by Trino. Therefore, in this case, access to the storage tool will not be performed via Trino and, instead, a dedicated mechanism might probably be necessary.

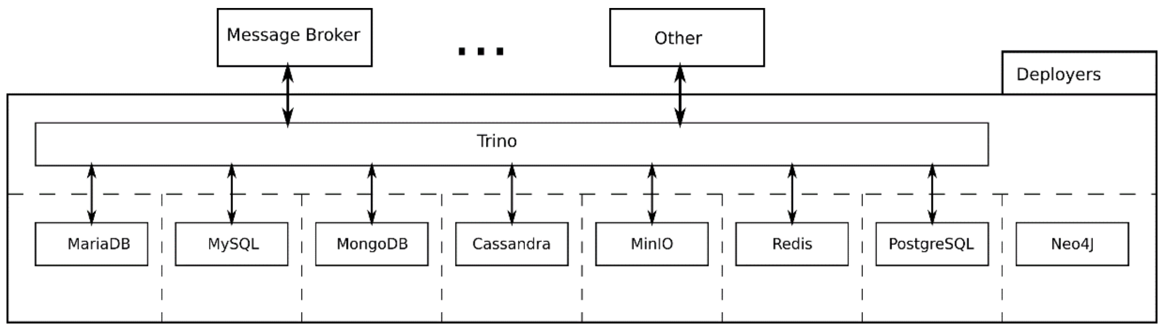


Figure 5. Architectural overview of i4Q^{DR}

In the context of the i4Q^{DR}, besides its efficient performance, the most important benefits of using Trino are related to interoperability aspects. Firstly, it enhances interoperability at an internal level. In this regard, Trino adds a level of abstraction that allows the pilots and solutions to have a relational SQL-based view of different data storage tools, no matter what type of data they handle. This also allows to build *federated views* of data that is managed by a heterogeneous set of tools, which in turns allows to build more complex and interoperable software components.

Secondly, Trino also enhances the interoperability of the i4Q^{DR} at an external level since it facilitates its interaction with other i4Q solutions. More specifically, Trino allows the i4Q^{DR} to offer a common communication interface for all the selected storage tools. For instance, Trino offers a REST API which will allow the interaction of the i4Q^{DR} with other solutions by means of the HTTP protocol.

4. Conclusions

This document provided an overview of the importance of data repositories in Industry 4.0 contexts. In Section 1 this explanation was illustrated with the i4Q^{DR}, the technical solution that will cover the data management requirements of any solution or pilot in the context of the i4Q project.

Moreover, this deliverable gathers the main problems and challenges arising when developing such data repositories. In this regard, a non-exhaustive list of recommendations addressing these problems and challenges has been proposed with the purpose of serving as a guideline for the development of data repositories. Indeed, Section 3, as an illustrative example, describes in detail how these recommendations have been applied in practice during the design of the i4Q^{DR} solution, and the definition of its development strategy.

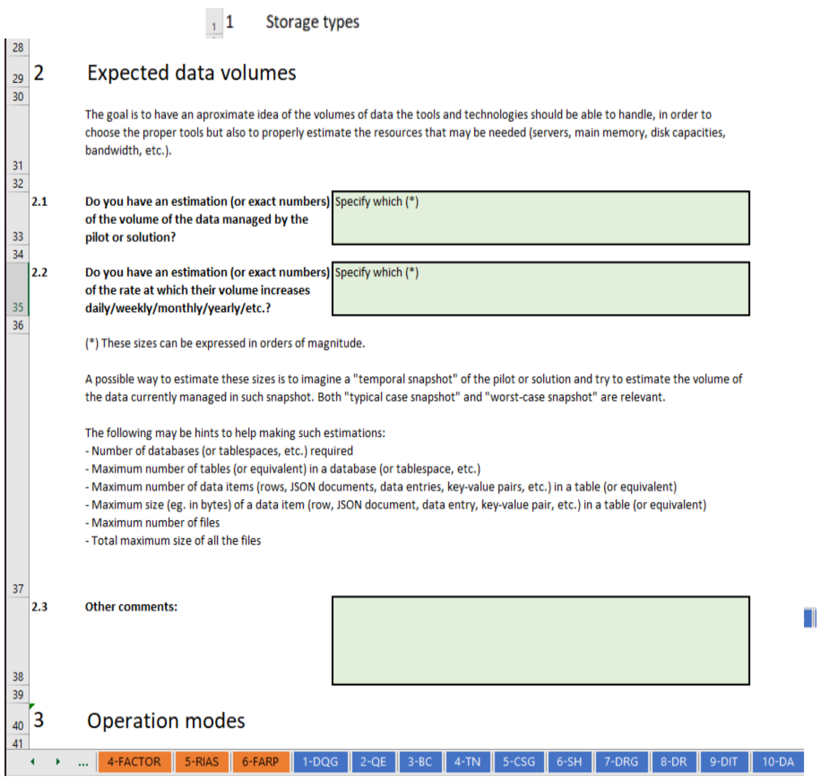
This document provides feedback to deliverable *"D3.8 - i4Q Data Repository"*, which presents the i4Q^{DR} solution, and to its future version, *"D3.16 - i4Q Data Repository v2"*, due on M24. In fact, deliverable D3.7 will also have a new version on M24, namely deliverable *"D3.15 - i4Q Guidelines for Building Data Repositories for Industry 4.0 v2"*, due on M24 as well. This new version will probably extend the content of the present document by explaining challenges arising in later phases of the development (e.g., deployment), and the corresponding recommendations addressing them, or considering other technical aspects not included here.

Appendix I: questionnaire on data storage needs

This appendix provides further information on the questionnaire distributed to the responsibles of pilots and other i4Q solutions. The purpose of this questionnaire was gathering more specific and technical details on their data storage needs. The information provided by the partners that filled in this questionnaire was used to select the most suitable tools and technologies to implement the i4Q^{DR}.

The questionnaire is a Microsoft Excel document containing different sheets. Besides some sheets providing an overview on the survey and useful information on how to fill in it, there is a sheet for each pilot and solution. These sheets contain a set of questions on different issues, as briefly explained in Section 3. Since this questionnaire is stored in a private repository, several figures have been included in this section to show the concrete questions that were asked to the partners. More specifically:

- Figure 6 shows the questions about the type of storage that is necessary.
- Figure 7 gathers the questions on the expected volume of data to be handled
- Figure 8 reflects the questions on the expected type of interaction with the i4Q^{DR} (interactive, batch, stream-oriented, etc.), and the communication interfaces with the solution that may be used (HTTP, HTTPS, TCP, etc.).
- Figure 9 shows questions regarding other communication issues, and the needs related to data replication.
- Figure 10 gathers the questions in relation to authorization issues, deployment of the solution, and its expected performance.
- Figure 11 shows he questions asked to collect information on whether the partners have specific needs regarding tools or technologies to use, possible constraints and preferences about the premises where the solution will be deployed, or other concrete needs, such as data anonymisation or any other proposed by the partners.



1 Storage types

2 Expected data volumes

The goal is to have an aproximate idea of the volumes of data the tools and technologies should be able to handle, in order to choose the proper tools but also to properly estimate the resources that may be needed (servers, main memory, disk capacities, bandwidth, etc.).

2.1	**Do you have an estimation (or exact numbers) of the volume of the data managed by the pilot or solution?**	Specify which (*)
2.2	**Do you have an estimation (or exact numbers) of the rate at which their volume increases daily/weekly/monthly/yearly/etc.?**	Specify which (*)

(*) These sizes can be expressed in orders of magnitude.

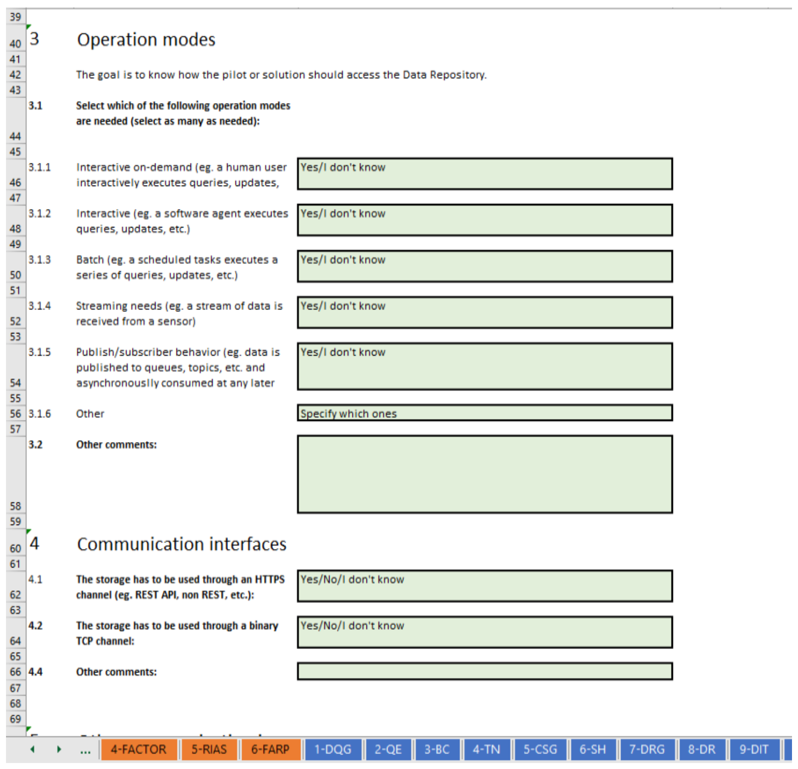
A possible way to estimate these sizes is to imagine a "temporal snapshot" of the pilot or solution and try to estimate the volume of the data currently managed in such snapshot. Both "typical case snapshot" and "worst-case snapshot" are relevant.

The following may be hints to help making such estimations:
- Number of databases (or tablespaces, etc.) required
- Maximum number of tables (or equivalent) in a database (or tablespace, etc.)
- Maximum number of data items (rows, JSON documents, data entries, key-value pairs, etc.) in a table (or equivalent)
- Maximum size (eg. in bytes) of a data item (row, JSON document, data entry, key-value pair, etc.) in a table (or equivalent)
- Maximum number of files
- Total maximum size of all the files

2.3	**Other comments:**	

3 Operation modes

Figure 7. Questions on expected data volumes



3 Operation modes

The goal is to know how the pilot or solution should access the Data Repository.

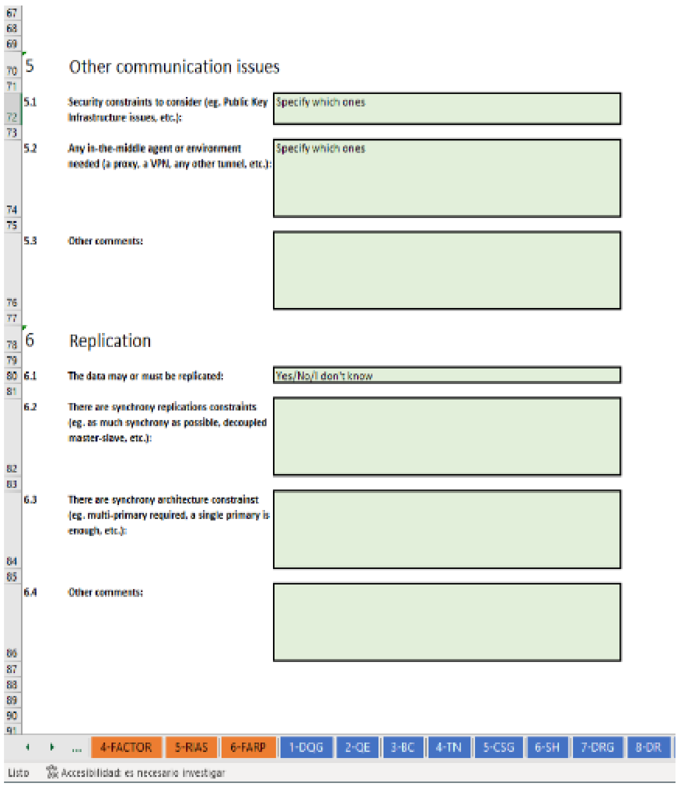
3.1 **Select which of the following operation modes are needed (select as many as needed):**

3.1.1	Interactive on-demand (eg. a human user interactively executes queries, updates,	Yes/I don't know
3.1.2	Interactive (eg. a software agent executes queries, updates, etc.)	Yes/I don't know
3.1.3	Batch (eg. a scheduled tasks executes a series of queries, updates, etc.)	Yes/I don't know
3.1.4	Streaming needs (eg. a stream of data is received from a sensor)	Yes/I don't know
3.1.5	Publish/subscriber behavior (eg. data is published to queues, topics, etc. and asynchronouslly consumed at any later	Yes/I don't know
3.1.6	Other	Specify which ones
3.2	**Other comments:**	

4 Communication interfaces

4.1	**The storage has to be used through an HTTPS channel (eg. REST API, non REST, etc.):**	Yes/No/I don't know
4.2	**The storage has to be used through a binary TCP channel:**	Yes/No/I don't know
4.4	**Other comments:**	

Figure 8. Questions on type of interaction and communication interfaces



5 Other communication issues

5.1	Security constraints to consider (eg. Public Key Infrastructure issues, etc.):	Specify which ones
5.2	Any in-the-middle agent or environment needed (a proxy, a VPN, any other tunnel, etc.):	Specify which ones
5.3	Other comments:	

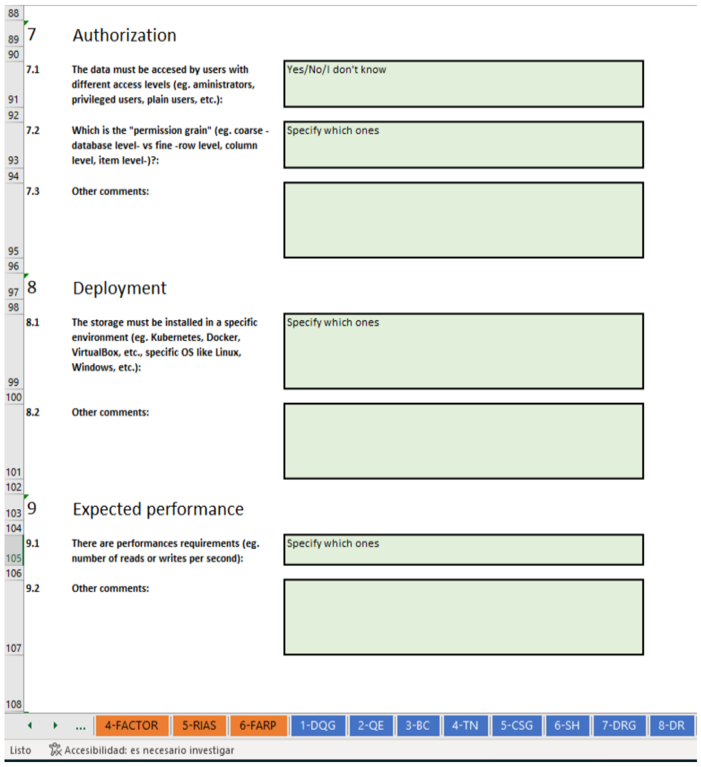
6 Replication

6.1	The data may or must be replicated:	Yes/No/I don't know
6.2	There are synchrony replications constraints (eg. as much synchrony as possible, decoupled master-slave, etc.):	
6.3	There are synchrony architecture constrainst (eg. multi-primary required, a single primary is enough, etc.):	
6.4	Other comments:	

4-FACTOR 5-RIAS 6-FARP 1-DQG 2-QE 3-BC 4-TN 5-CSG 6-SH 7-DRG 8-DR

Listo Accesibilidad: es necesario investigar

Figure 9. Questions on other communication issues and data replication needs



7 Authorization

7.1	The data must be accesed by users with different access levels (eg. administrators, privileged users, plain users, etc.):	Yes/No/I don't know
7.2	Which is the "permission grain" (eg. coarse -database level- vs fine -row level, column level, item level-)?:	Specify which ones
7.3	Other comments:	

8 Deployment

8.1	The storage must be installed in a specific environment (eg. Kubernetes, Docker, VirtualBox, etc., specific OS like Linux, Windows, etc.):	Specify which ones
8.2	Other comments:	

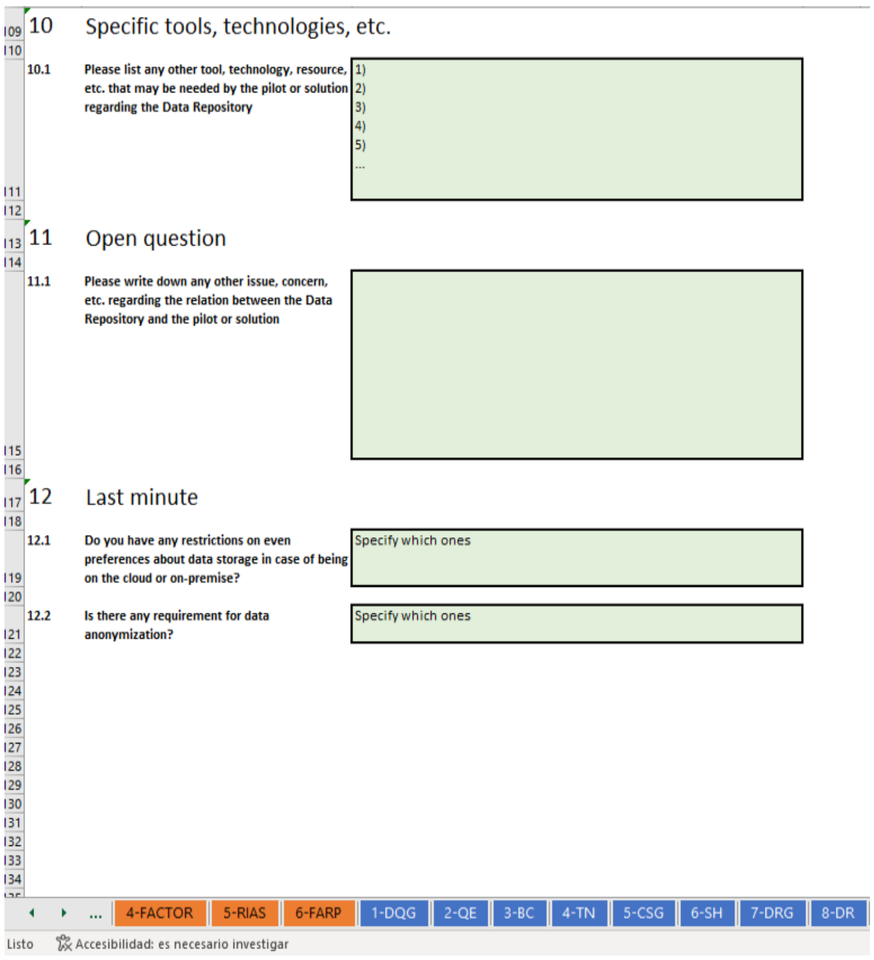
9 Expected performance

9.1	There are performances requirements (eg. number of reads or writes per second):	Specify which ones
9.2	Other comments:	

4-FACTOR 5-RIAS 6-FARP 1-DQG 2-QE 3-BC 4-TN 5-CSG 6-SH 7-DRG 8-DR

Listo Accesibilidad: es necesario investigar

Figure 10. Questions about authorization, deployment, and performance



10 Specific tools, technologies, etc.

10.1	Please list any other tool, technology, resource, etc. that may be needed by the pilot or solution regarding the Data Repository	1) 2) 3) 4) 5) ...

11 Open question

11.1	Please write down any other issue, concern, etc. regarding the relation between the Data Repository and the pilot or solution	

12 Last minute

12.1	Do you have any restrictions on even preferences about data storage in case of being on the cloud or on-premise?	Specify which ones
12.2	Is there any requirement for data anonymization?	Specify which ones

Figure 11. Questions about other specific needs and possible constraints and preferences

Appendix II: web documentation

This appendix shows the PDF version of the **i4Q Data Repository Guideline** (i4Q^{DRG}) web documentation, which can be accessed online at: **http://i4q.upv.es/7_i4Q_DRG/index.html**.

i4Q Data Repository Guidelines (i4Q^{DRG})

General Description

This solution is a guide for building Data Repositories for Industry 4.0, for estimating the storage capabilities and for building systems providing easy ways to access industry data and to communicate with industrial platform components and microservice applications.

The i4Q^{DRG} is not technical solution. Actually, it consists of two documents: deliverables *"D3.7 – Guidelines for building Data Repositories for Industry 4.0"* (due on M18), and *"D3.15 – Guidelines for building Data Repositories for Industry 4.0 v2"* (due on M24).

In D3.7 we first explain the role of data repositories in Industry 4.0 contexts and motivate the importance of the requirements they fulfill. With illustrative purposes, we relate this explanation with the i4Q project and its solutions, especially with the i4Q Data Repository (i4Q^{DR}) solution. Furthermore, this deliverable explains the challenges arising when building a data repository for Industry 4.0 and provides an overview on how to approach its development by gathering a set of recommendations addressing these challenges, focusing especially on the analysis and design phases, but also covering some aspects related to the implementation phase. These guidelines have driven the development of the i4Q Data Repository (i4Q^{DR}) Solution, presented in "D3.8 - i4Q Data Repository". The web documentation of the i4Q^{DR} solution is available here.

Features

Overview on data repositories in Industry 4.0: the i4Q^{DRG} explains the role of data repositores in the context of Industry 4.0, explaining the challanges and requirements arising when developing this type of solutions.

Description of i4Q^{DRG} design process: these guidelines describe how the design of the i4Q^{DR} has been approached.

Commercial Information

Authors

Company	Website	Logo
ITI	https://www.iti.es	
Uninova	https://www.uninova.pt	
Engineering	https://www.eng.it	
CERTH	https://www.certh.gr	
Knowledgebiz	https://knowledgebiz.pt/	

License

A free-software license

Pricing

Subject	Value
Payment Model	One-off
Price	0 €

Associated i4Q Solutions

Required

None.

Optional

These guidelines have driven the development of the i4Q^DR^.

Guidelines Summary

The development of data repositories in Industry 4.0 is very challenging, since this type of solution typically interacts with many other components, each one for different purposes. Consequently, the requirements of a data repository, especially the technical ones, are quite heterogenous, since the ones for a given component can be quite different from the ones for another component.

Therefore, the development of data repositories in that context requires a balance between two somehow opposite concepts. In the one hand, data repositories need to have a flexible design, so that they can fulfil heterogenous requirements coming from different dependent solutions. In the other hand, to reduce the complexity of the development, and the maintenance of the data repository once it is deployed, it is necessary to provide the required functionality in the most centralised fashion possible, by extracting as many common functionalities as possible and implement them in the most harmonised possible way.

Our recommendations to address the challenges arising when developing data repositories can be briefly summarised as follows. During the **analysis and design phase**:

1. Identify the components that will interact with the data repository.
2. For each component identified in step 1, gather the specific needs regarding:
   - The functionality required (e.g., only CRUD, or something else).
   - The volume of data.
   - Possible performance constraints, such as the expected number of read or write operations per second.
   - The operation mode. For instance, whether the data will be provided in an interactive mode, or in batch, and if will be done on-demand or as a scheduled task.
   - The need of replicated instances of the data repository.
   - The type of communications and expected interfaces and data formats.
   - Who and what components are authorised to access the data repository.
3. Analyse and summarise results, categorising needs and functionalities into those that are specific only to few solutions (or only one) and those that are common to several of them. Addressing the first group of needs might require using concrete tools. However, whenever possible, one should try to use tools that can work for other cases, too.
4. Address common needs and functionalities in the most general and harmonised way.

During the **implementation** phase the most important recommendations are:

1. Harmonise developments of common functionalities as much as possible
2. Follow similar structure and architecture

3. Simplify dependencies: use systems, tools, and libraries that do not have too specific requirements (e.g. Docker)

Other resources

- Deliverable *"D3.7 – Guidelines for building Data Repositories for Industry 4.0"* explains in detail the guidelines briefly presented in this web documentation.

Resource	Location
Last document release (v.1.0.0)	Link to D3.7 (private Repository)

- Deliverable *"D3.8 - i4Q Data Repository"* provides a technical description of the i4Q^{DR} solution, and explains its implementation status.

Resource	Location
Last document release (v.1.0.0)	Link to D3.8 (private Repository)

- The *web documentation of the i4Q^{DR}*, available here, contains further technical information on how to configure and use the i4Q^{DR}.