INTERVENE

Access to a biobank data set through implementation of local biobank APIs and cloud resources to support federated consortium access

Deliverable 1.4

Dissemination level: Public

Part of:

WP1:

Project summary	
Project acronym:	INTERVENE
Project full title:	International consortium for integrative genomics prediction
Project Coordinator	Institute for Molecular Medicine Finland FIMM, University of Helsinki; Prof. Samuli Ripatti and Dr. Andrea Ganna
Project start date:	1.1.2021
Project end date:	31.12.2025
Project duration:	60 months
Action type:	RIA
Call identifier:	H2020-SC1-FA-DTS-2018-2020 (Trusted digital solutions and Cybersecurity in Health and Care)
Grant number	101016775
Document descriptors	
Deliverable No.	1.4
Work package	WP1
Deliverable lead	CSC – IT Center for Science
Contributors	CSC, HUS, EMBL, UTARTU
Dissemination level	Public
Expected delivery date	31/10/2022
Submission date	31/10/2022

Change history log

Version	Changes made	Prepared by (name & organization)	Reviewed by (name and organization)	Date
0.1	**First draft**	**Kimmo Mattila (CSC)**	**Otto Manninen (HUS), Sudeep Das (HUS), Julius Anckar (UH-FIMM)**	**20.10.2022**
1.0	**Incorporation of change requests, minor technical edits**	**Kimmo Mattila (CSC)**	**Julius Anckar (UH-FIMM)**	**31.10.2022**

Contents

Introduction

The overall aim of the INTERVENE project is to integrate longitudinal health and genomics data to generate advanced Artificial Intelligence (AI) algorithms which would facilitate identification of individuals at high risk of common diseases. To be able to utilize data from several biobanks, INTERVENE needs to adopt a federated computational architecture where the patient data is held locally by the individual biobanks and algorithms are optimized at these local sites.

Currently, INTERVENE researchers interact directly with one or more INTERVENE analysts that have access to the biobanks and repositories. This means that the work is based on a network of personal connections and usage of tools and protocols that may vary in a case-by-case manner. The tools described here provide a solution that simplifies the communication between researchers and biobanks by providing a single contact point that connects INTERVENE researchers to INTERVENE biobanks.

An optimal solution would be a job submission tool that communicates directly with fully automatic APIs (Application Programming Interface) running in the computing platforms of the biobanks. However, because of strict security and regulatory requirements, many computing environments are not directly accessible from external servers. Instead, a manual intervention of the biobank analyst is needed, who processes the request of an INTERVENE analyst to collect the biobank specific genetic and phenotypic data, after which the task is executed in the local computing environment. Submitting the analysis results back to the INTERVENE analyst will also often require a manual step where the biobank analyst retrieves the results back from the protected computing environment. As manual task preparation cannot be totally avoided, building a fully automatic workflow that could be implemented in all the biobanks was not deemed feasible.

The solution described here allows INTERVENE researchers to upload analysis requests to a central task repository in the Data Coordination Center (DCC). The DCC, hosted by CSC, uses cloud storage infrastructure called SD-Connect[1] to provide a hub which facilitates the storage and transfer of job requests, input data and containerized analysis pipelines that are needed to execute the tasks. The designated biobank analysts can receive the job requests from the DCC as self-sufficient packages to their local computer. These packages include containerized workflows and input data that can be easily moved to the local biobank analysis environment for the actual job execution. When the analysis is complete, the analyst uploads the results to the INTERVENE DCC. Once all biobanks have provided their results to the researcher, they can be downloaded from the INTERVENE DCC for further analysis.

The federated approach allows researchers to submit just one request instead of contacting each biobank separately. At the same time, biobanks need to communicate with just one data hub when processing the analysis requests from different researchers. For this deliverable, we have applied the federated analysis across two biobanks, namely, Helsinki Biobank (HUS) and Estonian Biobank (University of Tartu). We describe the tools and process of task submission, data management and result retrieval process that can be applied by all biobanks for processing INTERVENE tasks. This process relates to the project task T1.3 (Federated compute infrastructure supporting AI analysis on the consortium data) as described in the description of action. The process, documented here, provides the first demonstration of the job processing procedure and is likely to improve as new technologies like Federated EGA and GA4GH passports become available. A schematic of these processes is outlined in Figure 1.

[1] https://docs.csc.fi/data/sensitive-data/sd_connect/

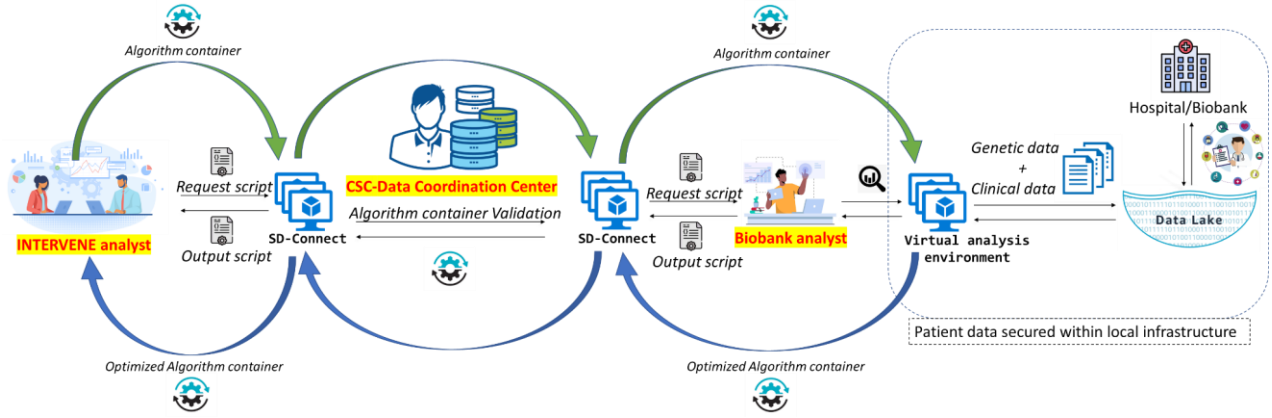

Figure 1. A typical job processing workflow from an INTERVENE analyst to a biobank analyst with DCC as a central task repository. Also shown here are the virtual interfaces used by the analysts such as the SD-connect.

Methods

In a previous deliverable (D1.5), the usage of algorithms (PGSC_calc and PRSpipe) were tested in the high-performance computing and sensitive data (SD) computing environments of CSC. During those tests, synthetic data stored in the SD-Connect was utilized. In the current test, we transferred the synthetic data to biobanks to run the algorithms in the biobanks locally. The process has been tested in the Helsinki Biobank and the Estonian Biobank as described. In the current test scenario, the synthetic genetic data and algorithms are made available through SD-Connect but in actual implementation, only the algorithms and not the genetic data will be shared through a central repository as reasoned earlier.

The analysis environments utilized by Estonian Biobank and Helsinki Biobank serve as examples of computational infrastructures currently used in biomedical research. The tests in Helsinki Biobank used environment, called HUS Acamedic[2]. This is a secure and audited operating environment that meets the requirements of both the Finnish Act on the Secondary Use of Social and Health Data as well as the Finnish Biobank act. Here it serves as an example of a modern cloud based and GDPR-compliant scalable virtual analysis environment used at European biobanks in the INTERVENE consortium.

In Estonian Biobank the test was executed in the Rocket computing cluster[3] of the University of Tartu. This cluster is a traditional HPC cluster that Estonian Biobank uses for computationally intensive tasks.

The demonstrator procedure described here utilizes two software components that interact with the central data repository hosted by INTERVENE DCC.

1. **iv-request.py** (INTERVENE requests). A tool that a researcher can use to upload an analysis request to the DCC to be processed and retrieve the results once they are ready.
2. **iv-analyst.py** (INTERVENE analyst). A tool that is used by the biobank analysts to download the requests for processing and to upload the results to the DCC.

[2] https://sway.office.com/869Qspuc7e2jUOYT?ref=Link
[3] https://hpc.ut.ee/services/HPC-services/Rocket

The iv-request.py and iv-analyst.py are command line python tools. They can be used by any computer that has Python3 installed with the following python libraries: `json`, `datetime`, `time`, `re`, `os`, `swiftclient` and `getpass`. iv-request.py and iv-analyst.py are available in the github repository of INTERVENE:

- https://github.com/intervene-EU-H2020/intervene-job-manager

The two software tools enable the task management between the INTERVENE analysts, the DCC central task repository and the biobank analysts which can be categorized into three procedural steps as described below (Figure 1).

1. Task submission and result retrieval by INTERVENE researcher using iv-request.py

When the iv-request is started, it asks for the CSC username and password to authenticate the user. This means that the user must have a CSC user account, which can be provided to all INTERVENE analysts.

After that the user can choose one of the four operations:

1. Listing submitted requests
2. Submitting a new analysis request
3. Downloading a processed request
4. Deleting a request

The analysis request submission is the main functionality of this tool. During submission, the user must define following items:

- task name
- requestor e-mail
- biobanks included in the analysis
- analysis type

At the moment, the iv-request includes only two predefined analysis types: test tasks for *PRSpipe* and *PGSC_calc* pipelines. In the case of predefined analysis, DCC provides the rest of the information needed for executing the task. If a user wants to submit some other type of analysis workflow, a singularity container that contains the software tools needed to run the workflow, must be included in input files. In these cases, users must also define:

- input files
- output files
- document containing executions instructions for biobanks
- computational resources needed (number of cores,memory, runtime and disk space).

Based on the information given, iv-request constructs a job description file that is uploaded to the DCC data repository together with the input files. The job description file is a JSON file that follows the GA4GH-TES standard[4] when applicable. However, as this use case includes manual processing steps, the job description file cannot always follow the GA4GH-TES standard.

[4] https://github.com/ga4gh/task-execution-schemas

In addition to the information that the submitter has provided the job description file includes the following task specific information:

- job specific ID code (generated based in the user account and submission time)
- locations of input data that is uploaded to DCC data repository

A sample job description file can be found in Appendix 1.

A researcher can have several jobs submitted for processing. She can use the task listing functionality of iv-request.py to check the status of all her tasks in all biobanks (waiting, processing, or ready). When all the requested biobanks have uploaded the results to the central repository, the researcher uses the download function to retrieve the results to her local computer (appendix 2). The result files will be stored into task and biobank specific folders for further analysis.

2. Central job management at the INTERVENE Data Coordination Center

2.1. SD-connect object storage

The job description file is stored in the SD-connect service at CSC into a storage area owned by the DCC. All the data is stored into a single data bucket within SD Connect. This data can be accessed only by the INTERVENE members that have been granted the permission to use this storage service (researchers and biobank analysts). At the moment data is stored without encryption, but crypt4gh based encryption can be implemented if sensitive information needs to be stored.

2.2 Data management with pseudofolders

Inside the object storage bucket the data management is based on job specific pseudo folders[5]. When iv-request.py uploads a new job, the data is stored to (pseudo)folder:

/bucket/jobs/*requestor-username/task-id*

The task-id based folder contains a subfolder for input data and biobank specific folders containing a biobank specific task description file (task.json). The biobank folder also contains subfolders for **status** and **output**.

Thus for example job with ID *task-1664892957-sarah*, submitted by username: *sarah* to be executed in Helsinki Biobank (HUS) and Estonia Biobank would create following data structure:

```
/bucket/jobs/sarah/task-1664892957-sarah/task.json
                                        /input/input.txt
                                        /HUS/task.json
                                            /status/submitted
                                        /Estonia_Biobank/task.json
                                                        /status/submitted
```

In addition a note-object, named after the task-id is added to a biobank specific folder. This object contains the location of the task file. In the example case above following note objects are created:

```
/bucket/Estonia_Biobank/requests/task-1664892957-sarah
/bucket/HUS/requests/task-1664892957-sarah
```

2.3 Script compatibility check

In the case of commonly used tasks, software containers needed to execute the analysis pipeline can be preloaded to the central repository. DCC will test the correct functionality of these containers in collaboration with the biobank analysts. As it is possible that different biobanks need different technical solutions to be able to run the pipeline, this can be addressed in the procedure by uploading biobank specific software containers to the central repository.

[5] Object storage services don't have a real directory hierarchy. In many cases object names are set so that they resemble file paths in file systems to help to manage the stored objects. As the folders are only included in the object names, the approach is called a pseudo folder structure.

3. Task processing by biobank analyst using iv-analyst.py

Biobank analysts use the *iv-analyst.py* tool in their local environments to interact with the tasks loaded to the DCC central job repository. When the script is started, it asks for the CSC username and password to authenticate to the DCC repository. After that the analyst selects her biobank and the task to be executed. The tasks are:

- show biobank specific task list including status of each task
- display job description file of a task
- downloading a task to start the analysis
- upload the results of the task to the data repository

When a biobank analyst selects the task to start the analysis, the interface downloads the pipeline container together with other input data needed as well as an instruction document. Typically this is a text document which describes the data to be collected from the biobank and the list of commands needed to run the analysis on that data, but in principle, this could also be a separate command script.

If the task will be executed in an environment that has internet access, the iv-analyst.py script can be used to copy the data directly to the analysis environment. In some cases, the actual analysis environment of a biobank is not connected to the internet, so the biobank analyst needs to first load the input data to some intermediate server or local hard disk drive and then move the data from there to the computing platform.

In the same way, the results are first exported from the biobank analysis environment to an intermediate platform where the iv-analyst.py tool can be used to upload the results to the central data repository.

Results

The role of the INTERVENE analyst was played by CSC personnel who created the task request script and the workflows (PRSPIPE and PGSC_CALC). These were made available at the DCC using the SD-connect. CSC was able to provide login credentials with access to project specific directories. The biobank analyst at Helsinki biobank and Estonian biobank were able to access and download the task request script and the workflows using SD-Connect. The file sizes for the workflows were moderately large (~40 GB) as it contained both the data and the algorithms, yet they could be easily downloaded from the secure server of DCC. As some INTERVENE analysis environments are without internet access, all the necessary dependencies and data files are needed to be prepackaged into the container. The workflow containers were first checked on the local computer for self-sufficiency and any missing packages were noted through the logs. The containers were packaged with these missing packages, which in our test case were certain conda packages and data files storing gene sequence variations. The test workflows were self-sufficient and did not require collecting local biobank specific dataset to be used in running the task. Thus, these workflows differ from the actual use cases.

The Rocket cluster, used by the Estonian Biobank, allows internet connections, which made running the test easy and straightforward. The Acamedic environment used by HUS is an example of isolated analysis environment where internet connections are blocked. As the Acamedic environment is fairly new and in beta phase, manual intervention by developers is required to upload these containers to the Acamedic.

The workflows can then be run by the biobank analyst on Acamedic, without administrator privilege or the need for the internet.

In summary, the basic functionalities of the process were tested by submitting PRSPIPE test job requests to HUS and Estonia Biobank. Sample commands to submit, analyze and retrieve the results are shown in appendix 2. These tests demonstrate that the job submission process works and that the Singularity containers allow INTERVENE to distribute pre–defined analysis pipelines to biobanks as a part of job requests.

However, already the first tests showed that the containerized analysis pipelines don't always work in the same way in all computing environments. Underlying hardware, operating system and other resources can have an effect. Thus the applicability of a new pipeline container should be checked in the computing environments of all biobanks where it will be used. In some cases, several versions of the pipeline containers may need to be created. Once working pipelines are available in DCC, the job management system can take care of providing the right pipeline container to each biobank.

Discussion and next steps

Currently, INTERVENE researchers have used common general purpose messaging and data sharing methods (e.g. e-mail, Slack, Google Docs, Figshare) to create ad-hoc solutions for running their analysis. That has been sufficient as the total amount of analysis requests has been low. However, when the amount of analysis tasks increases, more structured and well-defined task management methods will be needed. The task handling procedures discussed in this deliverable describe a preliminary demonstration of achieving a structured task management under a central task coordination center. Implementing a common well-defined procedure, where automation is applied when possible, will ease the workloads of both the INTERVENE researchers and the biobank analysts. Further, methods for systematically transporting and storing the results are at the moment missing and no centralized logging and result storage is applied. This could be improved in future iterations of the job management system.

In this test we demonstrated the possibilities of a centralized INTERVENE job management to the user community and were able to deduce key insights which will be vital for future development. There are several future developments that could be applied to make the procedure more practical including:

- developing a graphical user interface for request.py tool
- setting up a supporting service that collects log information of task execution and actively informs researchers and biobanks about the status of relevant jobs
- applying authentication methods that are not dependent on CSC user accounts only
- enabling encryption so that sensitive data can be included in the input or in the output
- creating a result archiving system

All the above development prospects are feasible, but before further development, feedback from the INTERVENE user community will be required to build additional features and match the needs of INTERVENE researchers.

Appendix 1. Sample of task description file:


```
{
  "ID": "1666270693-kkmattil"
  "name": "PRSPIPE test task for HUS",
  "description": "Predefined PRSPIPE fucntionality test.",
  "csc-user": "kkmattil",
  "requestor": "kimmo.mattila@csc.fi",
  "date": "2022-10-19 14:28:34.285234",
  "biobanks": ["HUS"],
  "requirements": [
        "singularity"
  ],
  "resources": [
        {
        "cpuCores": 4,
        "ramGb": 8.0,
        "diskGb": 100.0,
        "timeHours": 2,
        "preemptible": false
        }
  ],
  "storageserver": "allas",
  "inputs": [
        {
        "name": "infile",
        "description": "tar-package containing all data needed",
        "bucket": "2004504-prspipe",
         "object": "prspipe_8.7.22.tar",
         "url": "/2004504-prspipe/prspipe_8.7.22.tar",
         "path": "./prspipe_8.7.22.tar",
         "type": "FILE"
        }
  ],
  "outputs": [
        {
        "bucket": "project_2004504-intervene-tasks",
        "object": "jobs/kkmattil/1666270693-kkmattil/HUS/results/prspipe_results.zip",
        "url": "project_2004504-intervene-tasks/jobs/kkmattil/1666270693-
kkmattil/HUS/results/prspipe_results.zip",
        "path": "./prspipe_results.zip"
        }
  ],
  "instructions": [
        {
        "bucket": "2004504-prspipe",
        "object": "PRSPIPE_test_instuctions.docx"
        }
  ]
}
```

Appendix 2. Job processing example

The text below shows the command history of different steps of the sample job submission process. The command and selections that the user gives are displayed in blue bold face letters.

1. Job submission by a researcher

The INTERVENE researcher sends a new analysis request by launching command python iv-request.py in her local computer.

```
python iv-request.py

CSC username:
kkmattil

CSC password:
xxxxxxxxxxxxxx

Collecting task information.

Select a task:
1) list
2) submit
3) download
4) delete
5) update task list
6) quit

task: 2

Selected task: submit

Give a name for your task: Second PRSPIPE test for HUS
Give contact email address: xxxxx.xxxxx@csc.fi

Select target biobanks:
Include biobank HUS(y/n)? y
Include biobank Estonia_Biobank(y/n)? n
Include biobank HUNT(y/n)? n
Include biobank FIMM(y/n)? n

Selected biobanks:
['HUS']

Is this section OK(y/n)? y

Select analysis pipeline:
1) prspipe
2) pgsc_calc
3) other

Select analysis pipeline: 1

Selected Select analysis pipeline: prspipe

Uploading job 1666182816-kkmattil to project_2004504-intervene-tasks for HUS

Collecting task information.
Select a task:
```

```
1) list
2) submit
3) download
4) delete
5) update task list
6) quit

task: 1

Selected task: list
Tasks submitted by user kkmattil
Task:1666182816-kkmattil
        HUS: submitted

Select a task:
1) list
2) submit
3) download
4) delete
5) update task list
6) quit

task: 6
```

2. Job processing by biobank analyst

The biobank analyst starts the analysis by running command: python iv-analyst.py on a local server in the biobank network.

```
python iv-analyst.py

CSC username:
kimmokulju

CSC password:
xxxxxxxxxxxxxxx

Select a Select your Biobank.:
1) Estonia Biobank
2) HUS

Select your Biobank.: 2

Selected Select your Biobank.: HUS

Collecting task information.

Select a operation:
1) Show task list
2) Show the job request file
3) Download task for processing
4) Upload results
5) Quit

operation: 1

Selected operation: list

Collecting task information.

------------------------------------------------------------
Ready tasks:
------------------------------------------------------------
Tasks that have already been downloaded for processing:
------------------------------------------------------------
Tasks that are waiting to be downloaded for processing:
1666182816-kkmattil Second PRSPIPE test for HUS
------------------------------------------------------------

Select a operation:
1) Show task list
2) Show the job request file
3) Download task for processing
4) Upload results
5) Quit

operation: 3

Selected operation: download
Collecting task information.

Select a Task to be downloaded for processing.:
1) 1666182816-kkmattil

Task to be downloaded for processing.: 1
```

```
Selected Task to be downloaded for processing.:1666182816-kkmattil
Dir test
Get inputs

Downloading /2004504-prspipe/prspipe_8.7.22.tar to 1666182816-kkmattil/./prspipe_8.7.22.tar

prspipe_8.7.22.tar

get instructions
PRSPIPE_test_instuctions.docx
write json
Collecting task information.

Job downloaded to local directory 1666182816-kkmattil
Task specific instructions are available in: 1666182816-kkmattil/PRSPIPE_test_instuctions.docx

Select a operation:
1) Show task list
2) Show the job request file
3) Download task for processing
4) Upload results
5) Quit

5
```

The commands above create a new local directory named after the job id (1666182816-kkmattil). In this case the job directory contains files:

- prspipe_8.7.22.tar
- PRSPIPE_test_instuctions.docx
- task.json

Here the *prspipe_8.7.22.tar* contains the dataset that needs to be transported to the biobank computing environment. *PRSPIPE_test_instuctions.docx* contains the instructions on how the analysis is executed.

When the processing in the protected computing environment is read. The analyst move the result files, in this case *prspipe_results.zip*, to the job directory and launches the iv-anayst.py script again to upload the results.

```
python iv-analyst.py

CSC username:
kimmokulju

CSC password:
xxxxxxxxxxxxxxxx

Select a Select your Biobank.:
1) Estonia Biobank
2) HUS

Select your Biobank.: 2

Selected Select your Biobank.: HUS
Collecting task information.

Select a operation:
1) Show task list
```

```
2) Show the job request file
3) Download task for processing
4) Upload results
5) Quit

operation: 4
Selected operation: upload

Task to upload:
1666182816-kkmattil

Checking result files.

Result file ./prspipe_results.zip found.

All result files found. Starting upload.

Uploading: project_2004504-intervene-tasks/jobs/kkmattil/1666182816-kkmattil/CSC testing
only/results/prspipe_results.zip->./prspipe_results.zip
jobs/kkmattil/1666182816-kkmattil/HUS/results/./prspipe_results.zip

Collecting task information.

Select a operation:
1) Show task list
2) Show the job request file
3) Download task for processing
4) Upload results
5) Quit

operation: 5

Selected operation: quit
```

3. Result retrieval by a researcher

The INTERVENE researcher checks the status of jobs and downloads the ready jobs by launching the *python iv-request.py* in her local computer.

```
python iv-request.py

CSC username:
kkmattil

CSC password:
xxxxxxxxxxxx

Collecting task information.

Select a task:
1) list
2) submit
3) download
4) delete
5) update task list
6) quit

task: 1
Selected task: list

Tasks submitted by user kkmattil

Task:1666182816-kkmattil
        HUS: ready

Select a task:
1) list
2) submit
3) download
4) delete
5) update task list
6) quit

task: 3
Selected task: download

Listing ready tasks
Collecting task information.

1666182816-kkmattil ready

Select a task:

1) 1666182816-kkmattill
2) none

task: 1

Selected task: 1666182816-kkmattil

Downloading project_2004504-intervene-tasks/jobs/kkmattil/1666182816-
kkmattil/HUS/results/prspipe_results.zip to 1666182816-kkmattil/HUS/./prspipe_results.zip
jobs/kkmattil/1666182816-kkmattil/CSC testing only/results/prspipe_results.zip

Select a task:

1) list
2) submit
3) download
4) delete
```

```
5) update task list
6) quit
task: 6

Selected task: quit
```

The process above downloads the resulting output file of the requested procedure (prspipe_results.zip) to a job and biobank specific subdirectory. In this case: 1666182816-kkmattil/HUS/./prspipe_results.zip