



# Deliverable 1.6

## Cross-platform technical compatibility Demonstrated –Part 1

Grant Agreement Number: 101136962

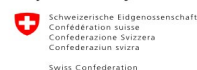


Funded by  
 the European Union



The British associated partners of NextGen were funded by UK Research and Innovation (UKRI) under the UK government's Horizon Europe funding guarantee [grant agreements No. 10098097, No. 10104323]

Project funded by



Federal Department of Economic Affairs,  
 Education and Research EAER  
 State Secretariat for Education,  
 Research and Innovation SERI

NextGen	
Project full title	Next Generation Tools for Genome-Centric Multimodal Data Integration In Personalised Cardiovascular Medicine
Call identifier	HORIZON-HLTH-2023-TOOL-05-04
Type of action	RIA
Start date	01/ 01/ 2024
End date	31/12/2027
Grant agreement no	101136962

Funding of associated partners
<p>The Swiss associated partners of the NextGen project were funded by the Swiss State Secretariat for Education, Research and Innovation (SERI).</p> <p>The British associated partners of NextGen were funded by UK Research and Innovation (UKRI) under the UK government's Horizon Europe funding guarantee [grant agreements No 10098097, No 10104323]</p>

Author(s)	R.Mitwicki, Ph.Page
Editor	-
Participating partners	#6- Human Colossus Foundation (lead) , HIRO, QMUL, UMCU
Version	1.0
Status	Submitted for review
Deliverable date	December 18 2025
Dissemination Level	Document (incl.videos): PU Public Code: SEN Sensitive
Official date	2026.01.16
Actual date	2026.01.16

## Disclaimer

This document contains material, which is the copyright of certain **NextGen** contractors, and may not be reproduced or copied without permission. All **NextGen** consortium partners have agreed to the full publication of this document if not declared “Confidential”. The commercial use of any information contained in this document may require a licence from the proprietor of that information. The reproduction of this document or of parts of it requires an agreement with the proprietor of that information., according to the provisions of the Grant Agreement nr. 101136962 and the Consortium Agreement.

Furthermore, a disclaimer will be included, indicating that: “Any communication or publication related to the action, whether produced collectively by the beneficiaries or individually in any format and through any medium, represents solely the views of the author, and the Commission bears no responsibility for any use that may be made of the information contained therein.”

## The NEXTGEN consortium consists of the following partners:

No	PARTNER ORGANISATION NAME	ABBREVIATION	COUNTRY
1	UNIVERSITAIR MEDISCH CENTRUM UTRECHT	UMCU	NL
2	HIRO MICRODATACENTERS B.V.	HIRO	NL
3	EURECOM GIE	EURE	FR
4	JOHANN WOLFGANG GOETHE-UNIVERSITAET FRANKFURT AM MAIN	GUF	DE
5	KAROLINSKA INSTITUTET	KI	SE
6	HUS-YHTYMA	HUS	FI
7	UNIVERSITY OF VIRGINIA	UVA	US
8	KLINIKUM RECHTS DER ISAR DER TECHNISCHEN UNIVERSITAT MUNCHEN	TUM-Med	DE
9	HL7 INTERNATIONAL FOUNDATION	HL7	BE
11	DATAPOWER SRL	DPOW	IT
12	SOCIETE EUROPEENNE DE CARDIOLOGIE	ESC	FR
13	WELLSPAN HEALTH	WSPAN	US
14	LIKE HEALTHCARE RESEARCH GMBH	LIKE	DE
15	NEBS SRL	NEBS	BE
16	THE HUMAN COLOSSUS FOUNDATION	HCF	CH
17	SCUOLA UNIVERSITARIA PROFESSIONALE DELLA SVIZZERA ITALIANA	SUPSI	CH
18	DRUG INFORMATION ASSOCIATION	DIA	CH
19	DPO ASSOCIATES SARL	DPOA	CH
20	QUEEN MARY UNIVERSITY OF LONDON	QMUL	UK
21	EARLHAM INSTITUTE	ERLH	UK
22	ASSOCIACAO DO INSTITUTO SUPERIOR TECNICO PARA A INVESTIGACAO E O DESENVOLVIMENTO	IST-ID	PT

## Document Revision History

DATE	VERSION	DESCRIPTION	CONTRIBUTIONS
2025.12.17	V09rc	Release Candidate to Coordination team	All
2026.01.16	V1.0	Delivered for submission	

## Authors

AUTHOR	ORGANISATION
Mitwicki, Robert	Human Colossus Foundation
Page, Philippe	Human Colossus Foundation

## Reviewers

REVIEWER	ORGANISATION
Benjamin, Rafy	HIRO
Lee, Aaron	QMUL
van Setten, Jessica	UMCU
Haitjema, Saskia	UMCU

## List of terms and abbreviations

ABBREVIATION	DESCRIPTION
ACDC	Authentic Chain Data Container
AI	Artificial Intelligence
CDM	Common Data Model
.CSV	Comma Separated Value a format for tabular data
DAG	Directed Acyclic Graph
DOA	Data Oriented Architecture
DICOM®	Digital Imaging and COMunications
DKMI	Decentralised Key Management Infrastructure
DKMS®	Decentralised Key Management System
DSL	Domain Specific Language
EHDS	European Health Data Space
EHR	Electronic Health Records
EMA	European Medical Agency
FAIR	Findable, Accessible, Interoperable, Reusable
FDA	USA Food and Drug Administration
FHIR®	Fast Healthcare Interoperability Resources
GA4GH	Global Alliance for Genomic and Health
GATK®	Genome Analysis Toolkit
GDPR	EU General Data Protection Regulation
HIPPA	USA Health Insurance Portability and Accountability Act
ICD	International Classification of Diseases
MDR	EU Medical Device Regulation
ML	Machine Learning
NLP	Natural Processing Language

OCA <sup>®</sup>	Overlays Capture Architecture
OMOP <sup>®</sup>	Observational Medical Outcome Partnership
PET	Privacy Enhancing Technology
PPT	Photoplethysmography
RTK	NextGen Research ToolKit
SAID	Self-Addressing IDentifier
SNOMED CT <sup>®</sup>	Systematized Nomenclature of Medicine Clinical Terms
.VCF	Variant Call Format. Standard data format defined by GA4GH

## Glossary

In this document, we use the following definition for specific terms.

Term	Short Definition
<b>data type</b>	Data type is the shape of information—its internal structure, its rules, and the kinds of operations you can perform on it. See also <i>structured data</i> , <i>semi-structured data</i> , and <i>unstructured data</i> .
<b>decentralised key management system (DKMS)</b>	A Decentralised Key Management System (DKMS) is a cryptographic key management framework that provides all the components needed to build authentication systems without relying on central authorities. It distributes the creation, storage, and control of keys to the owners themselves, uses decentralised authentication with self-certifying identifiers and immutable provenance logs, and enables secure, scalable, interoperable authentication across different platforms and networks.
<b>decentralised key management infrastructure (DKMI)</b>	DKMI is a set of interoperable protocols, data models, and network services that collectively enable decentralised key management systems to operate consistently and securely by providing common mechanisms for key discovery, authentication, provenance tracking, trust verification, and governance in decentralised environments.
<b>domain specific language</b>	A DSL is a specialized programming language designed for a specific domain, using its terminology to simplify and clarify tasks
<b>data lifecycle</b>	The sequence of stages through which health data passes, beginning with its collection and processing and extending to its reuse for specific purpose. For example, the EU EHDS framework starts with primary use, and extends to secondary reuse (e.g. research, innovation, policy-making). The lifecycle concludes with the validation of aggregated outputs and the eventual erasure of the data from the processing environment to ensure privacy and compliance

	with the EHDS Regulation and GDPR. (see [THEDAS2023a, section3]). A ten stage data lifecycle can be found in the glossary of [THEDAS2024b])
<b>GATK pipeline</b>	A widely used open-source software package developed by the Broad Institute for analyzing high-throughput sequencing data, particularly for variant discovery and genotyping. GATK is commonly used in genomic pipelines for tasks such as: read alignment preprocessing, variant calling (SNPs, indels), variant filtering and refinement. It supports various data types including whole-genome, exome, and targeted sequencing data.
<b>layered semantic architecture</b>	A layered semantic architecture organizes system components into separate layers, each with defined semantics and tasks (responsibilities), enabling modularity, reusability, and clear abstraction boundaries.
<b>Overlays Capture Architecture (OCA)</b>	OCA presents a schema as a multi-dimensional object consisting of a stable Capture Base and interoperable OCA Overlays.
<b>OCA Bundle</b>	An "OCA Bundle" refers to a collection of OCA Overlays and a Capture Base that conform to Overlays Capture Architecture (OCA) specification. OCA bundle is a snapshot without history of the OCA Objects whose integrity can be cryptographically verified. It is used as a consumable unit for capturing and representing data semantics.
<b>OCA File</b>	A text-based specification file written in the OCA File Domain-Specific Language (DSL) that defines and constructs an OCA Bundle by describing Capture Base structures, metadata, and overlay operations in a deterministic, layered form.
<b>OCA Overlays</b>	OCA Overlays are task-specific objects that have deterministic relationships with other objects. These "Overlays" define individual tasks, which, when combined, provide additional context to the object.
<b>OCA Repository</b>	An OCA repository is a structured, addressable storage and distribution system for Overlays Capture Architecture (OCA) artifacts. It enables the management, storage, and sharing of OCA Objects like OCA Bundles, Capture Bases, and Overlays
<b>Overlays Registry</b>	The Overlays Repository is a governed registry for OCA overlay definitions, enabling ecosystems to publish, discover, and apply task-specific overlays according to shared rules.
<b>PPT signal</b>	Is a non-invasive optical technique commonly used in wearable devices (like smartwatches and fitness trackers) to estimate heart rate, heart rate variability (HRV), blood oxygen saturation (SpO <sub>2</sub> ), and sometimes respiration rate.
<b>schema</b>	is the formal, versioned blueprint that describes the structure of data, the types and constraints of every field, and the relationships and rules governing how data is stored and validated. It makes data usable, interpretable, and enforceable across analytical and operational systems.
<b>self-addressing identifiers (SAID)</b>	A Self-Addressing Identifier (SAID) is a content-derived, cryptographically verifiable identifier whose value is deterministically computed from the data it identifies.
<b>semantic compatibility</b>	is the extent to which two or more information sources convey meanings that are mutually interpretable, enabling them to be combined, compared, or used together without misinterpretation.

<b>semantic alignment</b>	<p>The consistent representation and interpretation of meaning across datasets, systems, and standards — enabling transparency of data elements reference between senders and receivers, regardless of their source, format, or terminology. This includes alignment of vocabularies, ontologies, units of measure, and contextual metadata to preserve the intended meaning of data during exchange and reuse.</p>
<b>syntactic alignment</b>	<p>Technical harmonization of data structure, format, and encoding across disparate standards and platforms — ensuring datasets can be reliably exchanged, parsed, and processed by different systems without structural failure or manual reformatting. This includes alignment of file formats (e.g., CSV, JSON, FHIR bundles), schema definitions (e.g., OMOP CDM tables, FHIR resources), data types, field naming conventions, and serialization rules.</p>
<b>VCF –Variant Call Format</b>	<p>A standard text file format developed in 2010 for the 1000 Genomes project as a common output format for variant calling software.</p>

## Table of contents

<b>EXECUTIVE SUMMARY .....</b>	<b>12</b>
<b>CROSS-PLATFORM TECHNOLOGY COMPATIBILITY DEMONSTRATION .....</b>	<b>12</b>
<b>-PART 1.....</b>	<b>12</b>
<b>1. CROSS-PLATFORM TECHNICAL COMPATIBILITY IN PERSONALISED CARDIOVASCULAR MEDICINE. WHY IS THIS SO HARD TO ACHIEVE?</b>	<b>15</b>
1.1. GOAL AND RESEARCH SCOPE	15
1.2. WHY IS TECHNICAL COMPATIBILITY SO CHALLENGING?	15
1.3. NEXTGEN APPROACH TO ADDRESS THE CROSS-PLATFORM COMPATIBILITY PROBLEM	16
<b>2. SEMANTIC COMPATIBILITY</b>	<b>19</b>
2.1 A DETERMINISTIC ARCHITECTURE FOR SEMANTIC COMPATIBILITY	19
2.1.1 Structured Data vs. Unstructured Data	19
2.2 Introducing the Overlays Capture Architecture -OCA	20
2.2.1 OCA core concepts	21
2.2.2 OCA Overlays	21
2.2.3 OCA Bundles	25
2.2.4 OCA Repository	25
2.2.5 OCA Ecosystem	26
2.3 Semantic Flow with OCA	26
2.3.1 Why create a semantic	26
2.3.2 Creation of Semantic using OCA	27
2.3.2.1 Creation of the Overlay Files	27
2.3.2.2 Dissemination of Overlays: The Overlay Registry	28
2.3.2.3 Creation of the OCA File	28
2.3.2.4 Dissemination of OCA File: OCA Repository & OCA Bundle	30
2.4 SDK components for semantic	31
<b>3. DATA-LEVEL COMPATIBILITY</b>	<b>32</b>
3.2 Enhancing Data-Level Compatibility in NextGen: The Role of the Multi-Modal Integration Object (MMIO)	32
3.2.1 MMIO in NextGen	32
3.3 Creation of MMIOs	34
3.3.2 Encapsulation of modalities	35
3.3.3 MMIO's library	35
3.4 MMIO properties	35
3.4.4 MMIO indexing	36
3.4.5 Portability: MMIO as an accurate data container	37

3.4.6	Selective Disclosure	37
3.4.7	Chained Linkability	37
3.4.8	Advanced authentication	38
3.4.9	MMIO for Pathfinder Functionalities	39
3.4.9.1	Pathfinder Adaptor	39
3.4.9.2	Provenance & Information Discovery in NextGen	39
3.4.9.3	Discovery Mechanism: indexing of MMIO	40
<b>4.</b>	<b>OTHER DIMENSIONS OF CROSS-PLATFORM TECHNICAL COMPATIBILITY</b>	<b>41</b>
4.1	Hardware and Runtime Compatibility .....	41
4.2	Model-level Compatibility .....	41
4.3	Regulatory and Workflow compatibility .....	42
4.4	Status as of M24 .....	42
<b>5.</b>	<b>CONCLUSIONS</b>	<b>43</b>
<b>6.</b>	<b>REFERENCES</b>	<b>44</b>
<b>APPENDICES</b> .....		<b>46</b>
4	Examples	46
2.0	LIST OF OVERLAYS	47

## Executive Summary

### Cross-Platform Technology Compatibility Demonstration –Part 1

This document presents the first demonstration of technologies designed to enable semantic and data compatibility, within the broader context of data integration challenges in research workflows — including genomics — and their application to the NextGen initiative. The demonstration comprises two executable components:

- **oca** semantic construction
- **m2io** Multi-Modal Integration Object (MMIO) construction

Two components (oca and m2io) , tagged as SEN, are made available in the private NextGen GitHub repository for training NextGen participants, and for development and testing of tools intended for integration into the Pathfinder platform. These two components enable a cross-platform search capability that is described in the D1.4 deliverable.

The repository will also receive test data used in relation to NextGen use cases and pilots to support the developments of data management tooling. Part 2, scheduled for delivery in December 2026, will focus on the application of the data integration tooling within NextGen relevant applications .

Short instructional videos are included to support tool usage and validation.

#### Problem Statement:

Healthcare generates enormous amount of data, often sensitive, and ranks among the top 5 data producers — yet this data remains largely siloed, locked behind proprietary platforms and incompatible systems. To address this, the European Union’s digital strategy introduced the concept of data spaces: governed ecosystems enabling secure, sovereign, and interoperable data exchange among producers, consumers, and regulators.

This vision culminated in the European Health Data Space (EHDS) — the world’s most advanced domain-specific data space. Within this framework, the NextGen project is developing Pathfinder, an experimental ecosystem designed to integrate genomic data into cardiovascular personalized medicine through novel, scalable mechanisms.

However, unlocking this potential requires overcoming formidable compatibility challenges — not only technological, but also economic, legal, and regulatory. These must be addressed within a multi-jurisdictional context that respects the sovereignty of all actors involved.

The NextGen project included in its scope the need to face the challenges of multi-jurisdictional barriers in data integration. As a result, to test and validate from a researcher perspective the technologies developed, NextGen is an international consortium — spanning EU Member States, the USA, and the UK.

#### NextGen’s Two Distinctive Approaches to Technology Compatibility:

##### 1. Multi-Stakeholder by Design

The NextGen project brings together an ecosystem of research organisations, software, AI, and ML development teams to develop integration tools, data protection lawyers, and civil society representatives. This collaborative approach provides a forum of expertise to approach **cross-platform compatibility across the entire data lifecycle — from a holistic perspective —** while remaining focused on cardiovascular personalized medicine.

Compatibility across the data life-cycle is addressed through five dimensions:

- Semantic Compatibility
- Data Compatibility
- Hardware & Runtime Compatibility
- Model Compatibility
- Regulatory & Workflow Compatibility

These dimensions are domain-agnostic and transferable to other health or non-health data spaces.

## 2. A New Paradigm for Data Management: Integrity & Authenticity

**Integrity:** NextGen is developing decentralized data management technologies and introduce new tooling for data harmonisation in multi-stakeholder environments. At its core is the Overlays Capture Architecture® (OCA) — a semantic layer that enables digital artifacts (datasets, algorithms, credentials, certificates) to self-describe their meaning, purpose, and integrate governance rules — directly encoded into the data object itself. We introduce new technology concepts that provide a paradigm shift in metadata handling — where data and its meaning are intrinsically linked. This provide an essential, machine-readable definition of the meaning of data across different sites. The new core concepts are:

- **Overlays:** – encoder of purpose, governance, and semantics;
- **OCA Bundles** — the next-generation data schema — cryptographically binding task specific **Overlays** in a verifiable structure over a **Capture Base:** the pure, structural data model.

**Authenticity:** In parallel, NextGen leverages a decentralised authentication layer, DKMI® enabling a decentralised authentication (i.e. digital identity) of actors, organisation, events and objects. The Multimodal Integration Object (MMIO) and NextGen’s discovery mechanism aim to demonstrate how a decentralized authentication architecture improves data integration into cardiovascular research pipelines.

**Integration & Pilots:** Together, the multi-stakeholder design and decentralized data management paradigm deliver modular, application-layer compatibility mechanisms. These are being piloted in NextGen’s Pathfinder platform, integrated with other core tools:

- Accelerated analysis pipelines
- Privacy-preserving federated machine learning
- Transformer-based models
- Data-oriented platform architecture

### Structure of the document:

Section 1 introduces a five-dimension compatibility framework to describe NextGen’s actions for cross-platform compatibility. Section 2 and 3 focus on **Semantic Compatibility** and **Data Compatibility** as they are at the core of the decentralised data management approach. These sections also includes description of the demonstrated functionalities at M24. Section 4 introduces the three other dimensions that will be the focus of Part 2 (due in December 2026) and Part 3 due in December (2027).

Figure Summary 1: The five compatibility dimensions approach

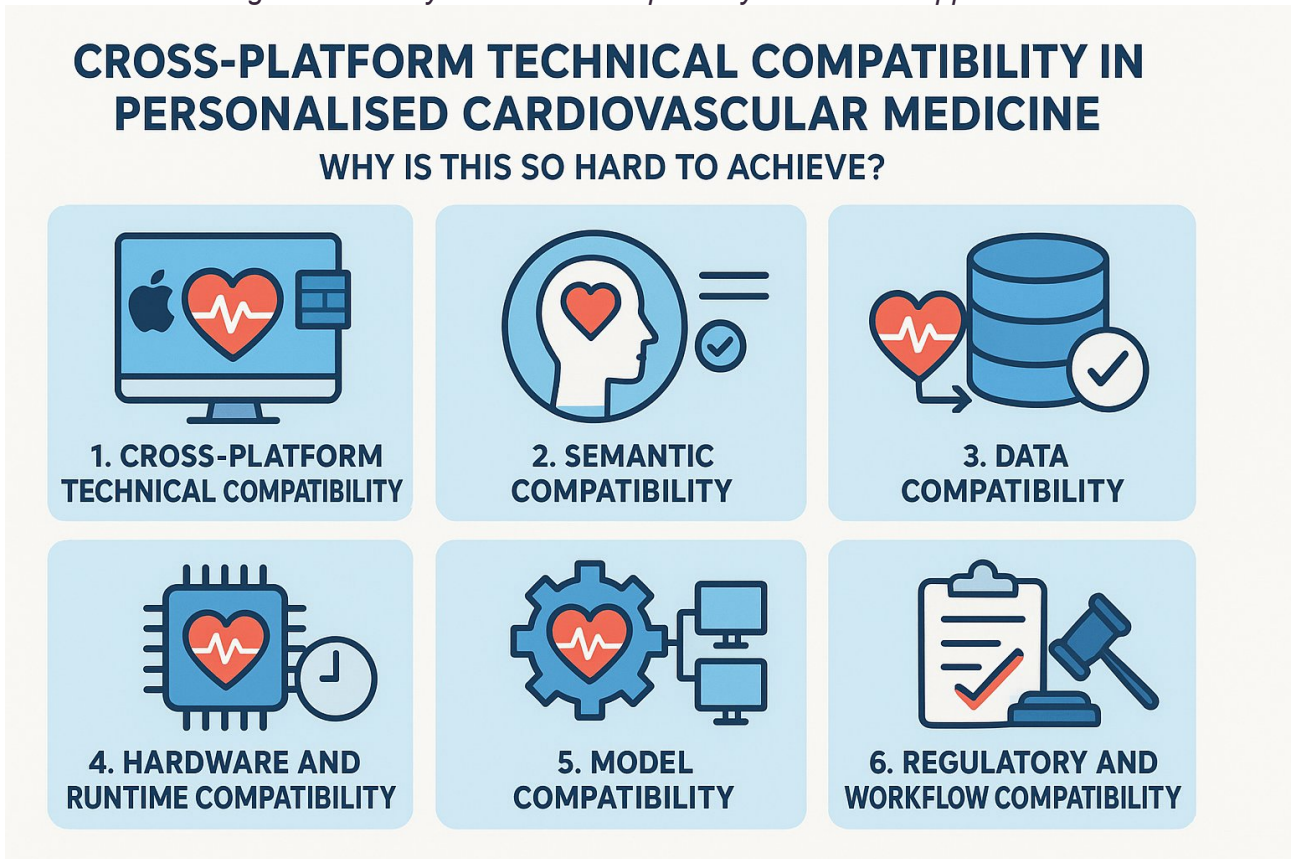


Figure Summary 2: Introducing OCA (visually)

## NextGen

<https://www.nextgentools.eu/> Demonstration of Overlays Capture Architecture for NextGen project

▶ Start watching

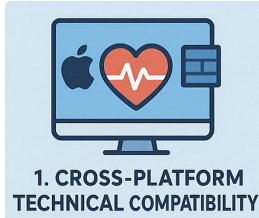
2 videos Autoplay next

Introduction: What is OCA  
HumanColossus

Overlay definition: overlayfile  
HumanColossus

# 1. Cross-Platform Technical Compatibility in Personalised Cardiovascular Medicine. Why is this so hard to achieve?

## 1.1. Goal and Research Scope



The NextGen consortium brings together research partners across the European Union, the United Kingdom, and the United States. Their shared mission is to advance cardiovascular personalized medicine, focusing on conditions including: heart failure, sudden cardiac death, coronary heart disease, myocardial infarction, high-risk cardiomyopathy, atherosclerosis, congenital cardiac disorders.[Page2024].

To develop truly individualized therapies, researchers must gather and integrate vast amounts of patient-specific data — a task where NextGen’s technological infrastructure plays a central role.

### NextGen’s Technological Approach

NextGen’s technology partners are developing solutions in four key areas:

- **Decentralised data management** for data harmonisation and integration of multi-modal datasets (genomic, clinical, imaging, etc.);
- **Data oriented** infrastructure and platform
- **Accelerated, secondary and tertiary genomic analysis** for vendor agnostic, CPU independent, open-source solutions;
- **Secured federated ML and AI tools for genomic computation** for genomic data curation, integration and interpretations.

In 2025, a “hop-on” partner was added to enhance the NextGen toolbox:

- **Transformer-based AI technologies**, further expanding its analytical capabilities.

These tools are selectively applied to specific research questions, with the goal of demonstrating the functionalities in pilots.

## 1.2. Why is technical compatibility so challenging?

Integration of heterogeneous data sources originating in different computational frameworks, hardware architectures, and analytic pipelines faces several hurdles. Illustrative examples are:

- **Discovery challenge:** Different institutions and researchers use inconsistent terminology, making data linkage difficult. Finding relevant, scientifically accurate data across distributed repositories — often with unstructured metadata, requires manual exploration and data curation. In such searches, guessing is not an option. *This is the **semantic incompatibility dimension** of the cross-platform technical compatibility problem.*
- **Data standards mismatch:** Hospitals and research institutions use different EHR standards (e.g. HL7, FHIR, proprietary schemas). Interoperability between standards requires the development of adaptors or interfaces. *This is the **data-level incompatibility dimension** of the cross-platform technical compatibility problem.*
- **Fragile bridges between analytic and data models** The challenges in analytical and modeling approaches go beyond data harmonisation and standards [Chahal2025aa, p331]. The active development of frameworks proposed by large initiatives like the Global Alliance for Genomic and Health (GA4GH)’s phenopackets [GA4GH, Jacobson2022] demonstrate the need for compatibility across models and lead to the standard interfaces (API, schemas, ontologies). But these require adherence to one framework often at the expense of another or loss of flexibility when the use of unstructured data is necessary. This is particularly the case for contextual metadata accompanying public omics data without which secondary analyses are limited. *This is the **model-level incompatibility dimension** of the cross-platform technical compatibility problem.*

- **Research reproducibility across hardware platforms:** Algorithms used in research, particularly AI, needs to provide reproducible output across different hardware platforms, all else being equal. *This is the **hardware and runtime incompatibility dimension** of the cross-platform technical compatibility problem.*
- **Legal & Ethical Access:** Hospitals and research institutions use different anonymization pipelines, different regulatory interpretations. To make data lawfully accessible, compatible technical measures must be in place to ensure compliance with for example generally applicable privacy regulations (e.g., GDPR) and specific data-sharing agreements. *This is the **regulatory and workflow incompatibility dimension** of the problem. This is the **regulatory and workflow incompatibility dimension** of the problem.*
- **Technical Interoperability:** Even when data is available, differences in file formats, standards, and systems prevent seamless integration.

### 1.3. NextGen Approach to address the cross-platform compatibility problem

#### Dimensions of cross-platform technological compatibility

The challenges to cross-platform compatibility introduced in the previous section are organized to provide a structured approach — categorizing the different challenges into addressable groups we define as Compatibility Dimensions. Ultimately, the goal is to establish cross-platform workflows that function reliably: if any single compatibility challenge is not met, the workflow will be affected.

Technologies developed in the NextGen project aim to lower barriers to data integration, thereby improving research in personalized cardiovascular medicine. Guided by a holistic perspective — yet focused on practical application within specific pilot projects - this comprehensive perspective ensures coherent development of the diverse NextGen integration tools (including data management, privacy-preserving federated learning, accelerated genomic pipelines, and transformers) to be applied in algorithms (AI), as contributed by participating research organisations; each focusing on practical application within pilots facilitates coherent development of nextgen tools.

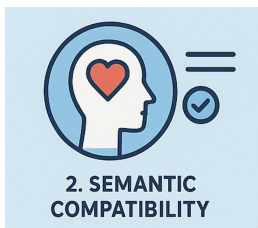
Table 1.1: Dimensions of technical compatibility in cardiovascular personalised medicines

No.	Compatibility Dimension	Short Description	In NextGen
1	Cross-platform technical compatibility: Why is it so hard to achieve		
2	<b>Semantic Compatibility</b>	<p>Even when platforms agree syntactically, they disagree semantically. The meaning of information must be harmonised.</p> <p><i>Example:</i> OMOP was designed for clinical semantics. Multi-omics and high-res phenotyping require finer distinctions than OMOP natively captures. Semantic incompatibility happens when two sites map ostensibly identical clinical or biological events to different OMOP concepts, different vocabularies, or different levels of granularity — producing seemingly harmonized tables that actually encode different meanings.</p> <p><i>Technical Consequences:</i> Cross-platform datasets can't be merged naively because the "same variable" encodes</p>	<p>Decentralised semantics OCA</p> <p>Common standards compatibility (OMOP, FHIR, CDISC)</p> <p>Semantic level discovery mechanism</p>

		different underlying physics, biology, or statistical assumptions. Harmonising the semantic without forcing everyone to use a single standard require mechanisms to ensure verifiable data integrity across different semantic domains, possibly in different jurisdiction.	
3	<b>Data-level Compatibility</b>	<p>Data-level incompatibility is the mismatch between datasets originating from different modalities, platforms, or domains.</p> <p><i>Example:</i> In cardiovascular personalized medicine, data incompatibility from ECGs, MRI scans, genomic sequences, EHR entries, wearable sensors that are represented in modality-specific ways as they represent different types of information.</p> <p><i>Technical Consequences:</i> Domain-specific preprocessing pipelines are required for each modality.</p>	<p>Mechanisms based on MMIOs to facilitate interoperability and information discovery.</p> <p>Synthetic data generation</p> <p>Federated ML</p> <p>Transformers for missing data</p> <p>Research Toolkit for researchers access to multimodal datasets</p>
4	<b>Hardware &amp; Runtime Compatibility</b>	<p>Hardware incompatibility arises when identical software (libraries, executables) can not be deployed executed on different hardware. Runtime incompatibility arises when, in addition, yield non-equivalent results when run on different system.</p> <p><i>Example:</i> The same code base compiles and can be executed in different environment but the outcome differs.</p> <p><i>Technical Consequences:</i> Hardware and runtime combatibility needs to be assessed on a case-by-case basis to establish if issues are present in a given context</p>	<p>Open-Source, CPU independent acceleration of secondary and tertiary analysis in GATK pipeline</p>
5	<b>Model Compatibility</b>	<p>Model compatibility refers to the coherence of the model assumptions used in a research workflows particularly in multi-stakeholder environments.</p> <p><i>Example:</i> In traditional software engineering, compatibility refers to the form of input/output (e.g., types, dimensions, domains of validity). For machine learning (ML) software, this is too limited: the ML models themselves come with their own assumptions about inputs/outputs distributions that the software ignores. Since ML software lacks visibility into these assumptions, ensuring data consistency across pipelines falls to engineers — a manual, error-prone process.</p> <p>Our approach aims to enable ML software to validate probabilistic properties of models and their I/O at runtime by augmenting data objects with metadata (via overlays), allowing programmatic model compatibility checks within the system itself.</p>	<p>Machine verifiable augmented model metadata</p>

		<p><u>Technical Consequences:</u> Requires a technical solution to enrich meta-data with model assumptions and ensure communication how the model is to be used. See for example [SENDAK2020]</p>	
6	<p><b>Regulatory &amp; Workflow Compatibility</b></p>	<p>Regulatory &amp; workflow compatibility is not primarily technical. It comprises governance, traceability, authorization, consent, auditability, and workflow alignment which have an impact on technology choices.</p> <p><u>Example:</u> The EU EHDS regulations lay down the regulatory framework for the most advanced data space in the complex mesh of EU, Member states, health regulators and private sector stakeholders active in healthcare. Compatibility between the regulatory regimes of these actors requires a balance between standardisation for effective process and digital sovereignty to preserve differences. The implementation complexities can be seen in the work of the THEDAS initiative for example.</p> <p><u>Technical Consequences:</u> Technical design choice driven by governance.</p>	<p>Multi-jurisdiction data integration framework.</p> <p>Privacy preserving ML</p> <p>REG and exploitation work packages</p>

## 2. Semantic Compatibility



Semantic Compatibility is the extent to which two or more information sources convey meanings that are mutually interpretable, enabling them to be combined, compared, or used together without misinterpretation.

In data management, semantic compatibility refers to the degree to which data, annotations, or metadata, share consistent meaning—using aligned vocabularies, ontologies, and contextual definitions—so they can be accurately compared, integrated, or interpreted across different datasets or

systems.

In NextGen semantic compatibility put forward as a core element to facilitate data harmonisation when needed in cardiovascular precision medicine. The tooling developed around the Overlays Capture Architecture can be used for standards interoperability, harmonisation of catalogue information or of different legal ontologies used across the different jurisdictions. NextGen must also consider the compatibility of its output with existing EU initiatives and the requirements of the upcoming EHDS regulations.

We introduce a mechanistic, verifiable approach to semantic compatibility using the Overlays Capture Architecture (OCA), enabling precise encoding of meaning within data models through structured, machine-checkable overlays; this framework supports semantic flows that preserve context across transformations, and is accompanied by experimental tooling provided to NextGen participants for training and extension, including schema validation and overlay creation utilities designed for incremental, interoperable local deployment that will serve the integration in the Pathfinder platform without requiring global schema alignment.

### 2.1 A deterministic architecture for semantic compatibility

#### 2.1.1 Structured Data vs. Unstructured Data

The main semantic incompatibility comes from unstructured data that must be supported alongside structured data and unstructured data.

- **Structured Data** - refers to information organized in a predefined, machine-readable format, typically conforming to a schema. It is usually stored in relational databases (RDBMS) or tabular formats (e.g., CSV, Excel), where each data element occupies a fixed field or column, and relationships between records are explicitly defined. Structured data enables efficient querying, indexing, and automated processing.
- **Semi-structured Data** lies between structured and unstructured data. It does not conform to a rigid schema like relational databases, but it contains tags, markers, or self-describing structures that impose some organizational hierarchy (e.g., JSON, XML). This allows for flexibility in data format while still enabling partial querying and parsing. Examples include log files, configuration files, and web APIs returning JSON or XML — formats that carry embedded structure but do not enforce fixed fields or relationships.
- **Unstructured Data** lacks a predefined data model or schema. It includes text documents, emails, social media posts, images, audio, video, and other media that do not fit neatly into relational tables. A key characteristic of unstructured data is the absence of self-description — meaning the data does not inherently contain metadata or semantic context that explains its content, structure, or meaning. While it may contain internal structure (e.g., headings in a document), it is not inherently organized for algorithmic processing. Unlike structured data, unstructured data requires external processing to extract meaning or impose structure.

The distinction between structured and unstructured data is not binary but exists on a spectrum. Structured data [1] has been around for a while. Since the development of System R in the 1970s, it has evolved into an ecosystem of

efficient tools (RDBM, CSV, etc.). Nevertheless, in a distributed data ecosystem, incompatibilities arising from other dimensions require novel ways to describe structured data. We need to be more dynamic and go beyond traditional standards that are too slow and often require complex consensus before data use can take place. Semi-structured data (e.g., JSON, XML) occupies the middle ground, offering some organizational hierarchy without rigid schema enforcement. 80 percent of the world’s information is unstructured. Unstructured information is growing at 15 times the rate of structured information [1]. Unstructured data is like a set of arbitrary functions. It has no fixed basis, no constant dimensionality. It must be mapped into a structure (e.g., vector embeddings) before computation.

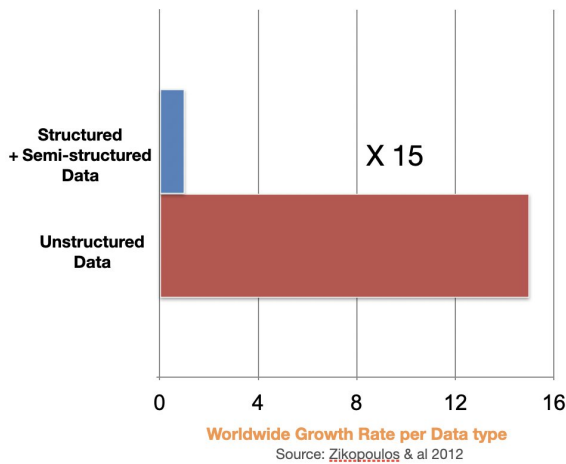


Figure 3.1: Unstructured data growth.

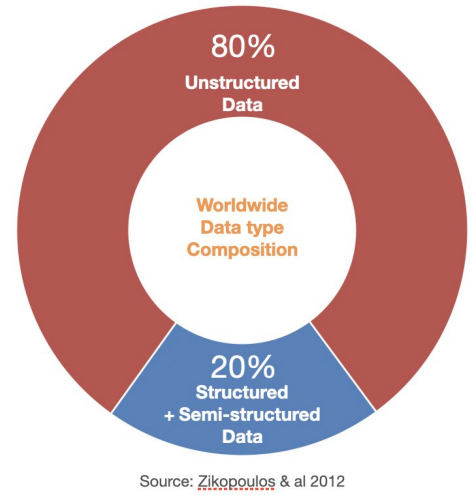


Figure 3.2: Proportion of unstructured data proportion

## 2.2 Introducing the Overlays Capture Architecture -OCA

The Overlays Capture Architecture, OCA is an explicit representation of task-specific objects called OCA Overlays that have deterministic relationships with other OCA objects. These “Overlays” define individual semantic tasks, which, when combined, provide additional context to the object. An OCA bundle consists of a “Capture Base” and “Overlays”. The sum of its parts represents a contextually-rich schema.

In data management the design of a system using the deterministic structure of OCA enables a cryptographic verification of the integrity of the data sets and other digital artefacts. OCA is based on the FAIR data principles [FAIR2016], a set of guiding principles to make data findable, accessible, interoperable, and reusable, to enable scientific data management and stewardship, and to be relevant to all digital economy stakeholders [HCF2023].

In NextGen, we deploy the Overlays Capture Architecture (OCA) version 2.0 and tooling to support participants for the integration of structured, semi-structured, and unstructured data types. OCA is a design choice to address the thorny question of “structuring unstructured data” — one of the current challenge of data integration. OCA introduces a mechanical protocol ensuring the integrity of an OCA bundle (i.e. data model). As a result, OCA removes uncertainties around schema definitions [HCF2023]. The main OCA characteristics relevant for its use in NextGen are:

- ⇒ Deterministic data structure (transparent)
- ⇒ Robust across sites (platform)
- ⇒ Context preserving (interoperability)

As a result OCA provides an architecture to deterministically preserve semantic and is free of automated “guess” effect. It provides the basis for developing advanced interoperability tools (e.g.OCA repository described below), develop common morphologic schemas linking data ecosystems, integration of standards for data harmonisation, ingestion, cataloguing and contractual use. Examples include:

- **Overlay Files** and **Overlay registry** introduce a mechanism to enforce system governance requirements including data quality, ethics, privacy, and security.
- **OCA Files** and the **OCA repository** enable auditable lineage via a domain-specific language for schema development and evolution.
- **MMIO** enables multi-modal data integration through an interoperable object that binds metadata lineage, schemas, records, policies, and agreements into a unified entity with a digital fingerprints (hashes).

### 2.2.1 OCA core concepts

The Overlays Capture Architecture (OCA) as per the OCA specification version 2.0 is an architecture designed to provide a standardized and interoperable way to define reusable metadata overlays in distributed systems. It is used to enrich base data with additional information while maintaining the separation between the base data and its contextual overlays. Encoding the purpose (previously called the "Data Model"). Detailed information on the OCA specification, tooling ecosystem can be found on the [OCA website](#)<sup>1</sup>[HCF2023].

The definition of overlays provide a deterministic and robust mechanism to encode purpose of use and governance element fundamentally bound to the data structure represented by the OCA bundle. The use of content based identifiers (SAID)[SAID2022] allow linkability mechanisms for data lineage.

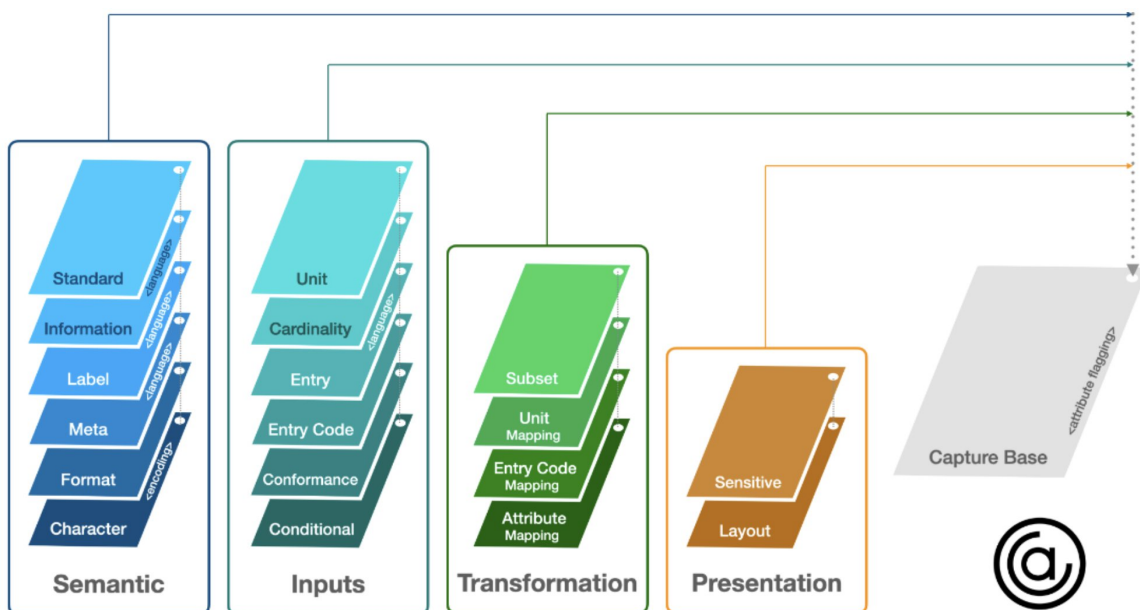


Figure 2.1 OCA as a Layered Architecture

### 2.2.2 OCA Overlays

In OCA layered architecture, each layer represent a “task” that represent a purpose (e.g. sensitivity, conformance) or property (e.g. units, encoding) of the data. Each layer therefore encode elements necessary for the use of and governance of data in a given context. The information is encoded into the **Overlay File** [HCF2025], the definition of an Overlay. In addition to the base Overlays defined in the specification, an ecosystem can define its own Overlays. Once defined, Overlays are stored in one or more registries, **Overlay Registries** that can be made accessible to the ecosystem participant.

Overlays are layered on top of the **Capture Base** that defines a single dataset in its purest form (see figure 2.1). The Capture Base plus one or more Overlays provide a structural base to harmonise data coined the OCA Bundle describe in section [OCA Bundles](#). The capture base

<sup>1</sup> <https://oca.colossi.network>

defines attribute names and types. Overlays as a task-specific object provides layers of definitional or contextual metadata.

The OCA specification defines a set of **Base Overlays** for the most recurrent tasks of data modeling. These are listed in the table 8.1 in appendix. They include thirteen (13) Overlays that have been identified through community consensus as fundamental for ensuring semantic interoperability and are among the most widely utilized.

In addition, starting with version 2.0, OCA allows the creation of new overlays for specific purpose in a given ecosystem. These Overlays that are created, maintained in an **Overlays Registry** made accessible by an ecosystem are called **Community Overlays**. The Human Colossus Foundation maintains an Overlays Registry openly accessible on GitHub [HCF2025a]. This registry contains additional information on Community Overlays. Other communities have created Overlays for specific purposes. For example Overlays for verifiable credential (VC) presentations have been created by [SWIYU2024] for the Swiss digital identity or by the Linux Foundation's Hyperledger Aries project [Curran2024]. These are based on OCA version 1.0 and work is underway to bring them to the specification of Community Overlays. We include in table 8.2 in appendix a list of selected Community Overlays.

### Example: Community Overlays & Governance

**Community:** NextGen is a consortium of 22 organisations operating under a Grant Agreement that provides the contractual relationships between the participants as well as the intended development to take place as part of the project. NextGen is therefore a community of interest interacting with the external world both in and out the EU.

**Overlays:** Specific Overlays will be defined as part of the NextGen project. Their creation could be motivated by technical (i.e. specific genomic format compatibility), research (i.e. specific items for search), or governance (e.g. authentication profile, diversity index). They could be created locally by a participant, by a group of participants of a Pathfinder pilot, by the whole consortium.

**Overlays Registry:** The use of these specific Overlays could be local, among a group of participants, by the whole consortium or even made available public for use outside the NextGen Consortium. As a result, the access to the Overlays Registry (i.e. The list of NextGen specific overlayfiles) needs to respond to NextGen governance. In practice, there will be multiple Overlays Registry made accessible either centrally or locally through the Pathfinder platform.

The OCA Specification deliberately minimizes predefined base overlays to avoid over-constraining the ecosystem. In NextGen we will develop the Community Overlays specific to the personalised cardiovascular medicine domain. This design choice delegates innovation to the community, enabling organic emergence of task-specific overlays through explicit use. From both spec and implementation perspectives, base and community overlays are semantically and functionally equivalent — no distinction is enforced, ensuring extensibility without fragmentation.

## Demonstration

### Adding an Overlay

#### Context

Within a dataset containing a patient cohort with attribute (defined by NextGen MVP variables (see D5.3, Appendix)) a researcher wants to ensure that a set of attributes is mandatory in the dataset while others optional. An OCA Bundle (i.e. a schema already exist)

#### Solution:

A CONFORMANCE Overlay is added. Its purpose is to ensure that mandatory attributes are flagged. Subsequently any data capture using the specific OCA Bundle will make the attributes either mandatory (M) or optional (O).

#### OCA Characteristic:

*Deterministic:* The conformance of the attribute is cryptographically bound to the data set during its complete life cycle from information collection (e.g. entry form, search), processing (e.g. input into an algorithm), presentation of the data (e.g. report, audit).

*Robustness:* The OCA bundle enforce the integrity of the dataset independently of the site. The conformance of each attribute is part of the definition of the dataset, independently of its location. Any change (i.e a new attribute becoming mandatory) can be detected through an integrity check at any point of the workflow.

*Context Preserving:* Adding an Overlay to an existing OCA bundle enables a mechanisms to encode a specific task relevant to the context. In the case of the CONFORMANCE overlay it allows to define context specific rules depending on the inteded use of the dataset.

```
ADD OVERLAY CONFORMANCE
attribute_conformances
age="M"
sex="M"
gender="O"
ethnicity="M"
previous_myocardial_infarction="O"
stroke="O"
chronic_obstructive_pulmonary_disease="O"
asthma="O"
atrial_fibrillation="O"
peripheral_artery_disease="O"
hypertension="O"
diabetes="O"
hypercholesterolemia="O"
chronic_kidney_disease="O"
height="M"
waist_hip_ratio="O"
waist_height_ratio="O"
sbp="M"
pulse_rate="M"
smoking_history="O"
```

### Example: Creating an Overlays

#### Context

A researcher in bioethics requires a finer definition of sex/gender (e.g. Following the SAGER guidelines [Ebbs2022]) traditionally found in cohorts. When collecting data for such purposes the available datasets might require enrichment or curation to meet the specific research needs prior to merging.

#### Solution:

A possible solution can be achieved with specific Overlays that are defined to ensure specific variables are either present or set to default values when the research data set is constructed. These Overlays can then be used to enrich existing schemas (OCA Bundles) of existing data set in the creation of research datasets specifically designed for the ethics research study.

#### OCA Characteristic:

*Deterministic:* The Overlay defines the expectation of the researcher. Any deviation can be identified at semantic level and trigger a specific response from the system

*Robustness:* The specific Overlay is uniquely identified, any modification is identifiable. If governance requires this Overlay, it must be present for the creation of a valid dataset.

*Context Preserving:* The Overlay is task specific. It can be added to existing semantics thus preserving the context of the original semantic.

We also include an example of Community Overlays outside NextGen to indicate the semantic interoperability with non-NextGen projects.

### Example: Community Overlays

#### Context:

We provide here an example of how overlays can be used for interoperability with other systems. Each ecosystem has different technology stack and different governance related to how a VC is displayed on a mobile device.

#### Solution:

Each ecosystem describes the VC visualisation independently according to their own rules. The Canadian pilot for digital wallet uses the hyperledger Aries open-source while the Swiss E-ID project develops a specific technology stack for their trust infrastructure. In both cases the end user needs is the same: securing the display of a certificate on a mobile device.

Both used the OCA and defined their own specific overlays of their use.

#### Additional Overlays

- Canada: [Hyperledger Aries Interoperability profile RFC 0877](#)<sup>2</sup>
- Switzerland: [Data Source Mapping Overlay](#)<sup>3</sup>

A solution provider that would need to develop an application capable of displaying both type of credential can simply use the two overlays with the assurance of semantic compatibility.

#### How this relates to NextGen:

A centralised Overlays Register hosted on the Pathfinder platform defines how certain digital artefacts are defined according to NextGen Governance rules. These are accessible to all participants. Each participant can use them and customise them according to their local need. Using the adaptor functionality of the MMIO, a transformation can be designed in a peer-to-peer manner to facilitate the interoperability between sites.

### 2.2.3 OCA Bundles

An **OCA Bundle** is a structured, self-contained and enhanced schema. It represents a layered semantic model of the data. The OCA Bundle is composed of:

- A mandatory **Capture Base** — defining the core data schema and semantic foundation.
- Zero or more **Overlays** — optional extensions that add context, constraints, or domain-specific semantics atop the base.

The OCA Bundle, as a cryptographically verifiable set of Overlays and Capture Base, is a semantic artifact, designed to encode meaning in a structured manner. The OCA Bundle is at the root of OCA key properties:

- **Deterministic Identifier:** Each bundle as well as its overlays are assigned a unique ID derived from a cryptographic digest (e.g., Blake3). This enables content-addressable storage and integrity verification. Once defined, the Bundle's integrity can be verified independently of its origin.
- **Robustness:** The OCA Bundle is format agnostic. It can be serialized in a JSON format for interoperability, but not restricted to it — binary or other structured formats are permissible.
- **Context Preserving:** The definition of the OCA Bundle can contain as many layers as required by the context.

OCA Bundles represent a specific instance of the data model. It is created using a specific domain specific language (DSL) **OCA File** designed to ensure traceability of the evolution of the data model definition. It also provides a powerful tool to create new Bundles from existing ones (see OCA Repository for more details). The OCA File is described in [HCF2025] the latest version of the OCA File specification openly available on the OCA website,

### 2.2.4 OCA Repository

The OCA Repository is part of the OCA Ecosystem, a set of tools to support developers of OCA based solutions. The current set of tools available is accessible on the official OCA website<sup>4</sup>. An OCA Repository is where OCA Bundles are stored and made accessible through their identifiers (SAID).

In data management they play a central role. They are a deterministic source of schema that can describe any dataset in a cryptographically verifiable manner. They are a key component of the verification of the integrity of data. OCA repositories include a specific knowledge graph structure (Directed Acyclic Graph –DAG) that ensures a deterministic ordering of the schema modifications. OCA Bundles are always modified and never deleted. This is necessary to ensure long term verification of the integrity of an OCA Bundle. The OCA version demonstrated at M24 is “stand alone” and is not integrated into an authentication mechanism. Once done (2026) users of the repository will also know where the object comes from.

In NextGen, the Pathfinder platform contains a “*central*” OCA repository and each node will operate independent local OCA Repositories under their own control. This design enables specific applications to be compatible across sites. For example NextGen federated catalogues rests on an information discovery system that includes semantic searches both done globally (central node) and locally. Separating the semantic search from the data record is a core new mechanism deployed in NextGen to ensure local privacy consideration can be implemented and managed dynamically as each node has full control over their repositories.

As part of this delivery, tooling to create and use OCA Repositories locally is deployed through Docker containers made available to NextGen participants on the private NextGen GitHub repository (access reserved to development). This will allow local experimentation and testing prior to integration in PathFinder platforms or other specific applications (e.g. Local Jupyter notebook)

<sup>4</sup> <https://oca.colossi.network/ecosystem/oca-repository.html>

### 2.2.5 OCA Ecosystem

As the OCA evolves more tools for developers of solution are added and form the OCA ecosystem.

The Human Colossus Foundation maintains access to these tools on the official OCA website. At the time of writing the OCA Ecosystem consist of:

- OCA bin <https://oca.colossi.network/ecosystem/oca-bin.html>
- Overlayfiles <https://oca.colossi.network/ecosystem/overlayfile.html>
- OCA File <https://oca.colossi.network/ecosystem/ocafile.html>
- OCA Repository <https://oca.colossi.network/ecosystem/oca-repository.html>
- Overlays Registry <https://oca.colossi.network/ecosystem/overlay-registry.html>

In NextGen each of these tools will be used either in the Pathfinder platform or locally to construct the advanced and interoperable tooling for data harmonisation.

### 2.3 Semantic Flow with OCA

By "Semantic Flow," we refer to the process of generating semantic information for any digital object (e.g., datasets, algorithms, agreements), binding that semantic information to the object, using it (e.g., in MMIO for indexing), and managing the semantic data. This section describes the elements required to create a Semantic Flow — which, in OCA terminology, translates to selecting or creating Overlays with OCA Files, generating OCA Bundles, storing them in the OCA Repository, and retrieving OCA Bundles from the OCA Repository. The data model is therefore represented by one or more OCA Bundle that can be fetched from OCA Repositories. The following section provides an overview of these steps.

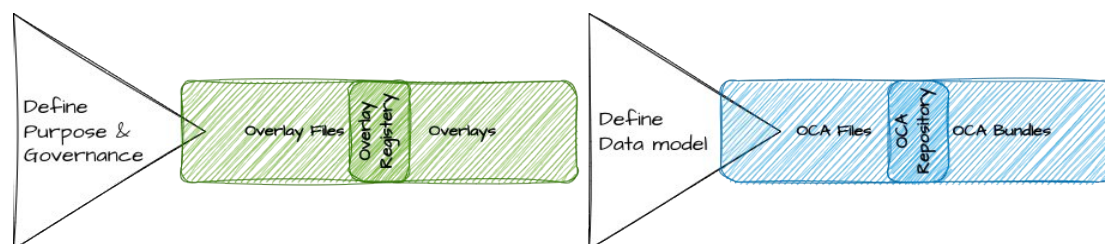


Figure 2.2 Semantic flow creation

#### 2.3.1 Why create a semantic

From a data management perspective, the implementation of the FAIR principles requires a strict management of what is called metadata - data describing the data and all relevant information pertaining to the meaning and context of use for the data. In practice, this is a challenge as data must be strongly bound to its metadata *across the workflows* in which data is used. By creating a machine-actionable semantic, we address this problem by forcing the data structure to be transparent and verifiable. This technology proves a game-changing tool for cross-platform compatibility as it allows different users in multiple site to verify independently the integrity of the data objects.

A file describes data. A semantic describes the meaning of that data - for example encoded in an OCA Bundle. Generally, a tabular file (e.g. CSV) or a genomic file (e.g. VCF), or database schema tells machines how data is stored, but not what each field means. In CSV files, a header row might be present, but it is just a label whose meaning is known by the creator, not by the file. The data is unstructured or semi-structured at best. In a VCF file, the complex header provides precise information but it requires adherence to a specific standard that evolves, is not comprehensive, and there is room for interpretation in certain fields. In data ecosystems involving many parties, compatibility is difficult to achieve and often requires quality assurance mechanisms (e.g. data curation). Similar problems face the structured data of specific database or biobank and lead to difficulties in maintaining robust interfaces.

As a result, in practical implementation, a file describes data but does not provide machine-accessible information on how it should be interpreted, validated, displayed, translated or presented. In short it is difficult to maintain the original meaning of data as it goes through a workflow across the system. Different systems may store or name things differently. For example, a tabular data file uses `date_of_birth`, another system `DOB`, or `birth_date`, or `dateOfBirth`. A semantic (e.g. OCA Bundle) defines a universal, canonical representation of the attribute. This allows mapping between different sources, lossless transformation, interoperability, and ultimately, long-term digital preservation

An OCA Bundle, as semantic object, adds a structured, machine-interpretable meaning layer. It makes the file self-describing. Think of it enriching values in a file with concepts, context and purpose of use.

**Semantic and FAIR principles:** Once you create a semantic for a concept for:

- “Person”
- “Address”
- “Genomic Variant”
- “Lab Test”
- ...

it becomes a reusable building block in an OCA Bundle. The OCA Bundle can be embedded as a reference, extended, or it can also be the basis to derive new schemas. The files become interoperable by design, not by accident. As a result, creating semantic within the Overlays Capture Architecture provides a deterministic mechanism to implement FAIR principles [FAIR2016].

### 2.3.2 Creation of Semantic using OCA

The high-level generic process of semantic creation is captured in figure 2.2 “*Semantic flow creation*”. With OCA, the definition of semantic starts with the encoding of the context, the purpose and governance of the digital artefact (e.g. files, data set, algorithm, agreement) with the OCA Overlays introduced in section 2.1.2.2. By default, NextGen will provide the base Overlays defined in OCA specification v2.0 as well as all the Overlays definitions common to all NextGen partners. These will include for example Overlays definitions for legal, compliance and ethics measures as defined by T6.4 in the WP6 REG work package. The definitions of Overlays are stored in one or more Overlays Registry. Then the encoding of the meaning of digital artefacts is done through the creation of OCA Bundles.

Both Overlays and OCA Bundles are build from text files. The process is facilitated by a domain specific language (DSL) that can be encoded in standard text files. The Overlay File defines the overlays and is available from an overlay registry. The OCA File is ingested by the OCA Repository that process and store them to produce the OCA Bundle upon user request (e.g. through API or command line interface).

#### 2.3.2.1 Creation of the Overlay Files

The NextGen project complies with the OCA Specification v2.0 [HCF2023] that includes the base overlays as described (see list in appendix). Therefore the definition of these Overlays is available by default in the NextGen Overlays Registry.

New Overlays definitions can be build by adding to existing Overlays definitions or creating an Overlay definition from scratch.

### Demonstration: Semantic Flow with OCA

The Pathfinder platform has done a first integration of the OCA component. The semantic flow has been independently demonstrated in a work session. The purpose is to first establish a local understanding of the approach with the participant themselves to facilitate the integration in the Platform.

The demonstration was based on the two executables (oca, m2io) that will be made available to NextGen participants via the private NextGen test GitHub repository (Jan.2026).

The **oca** executable will be used for tutorials, training and pilot development in 2026. The goal is for the participants to co-create the semantics relevant for their use cases and pilots.

Videos showcasing steps of the demonstration are available here: <https://vimeo.com/showcase/12027472>. Each video describe a specific aspect of the semantic flow with OCA. They will be further develop as the project progresses

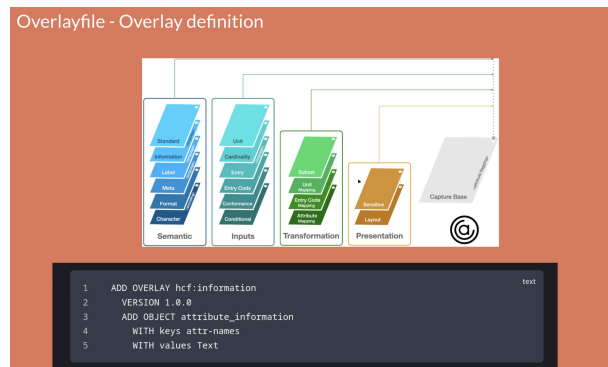
## NextGen

2. Semantic flow  
define accurately your data,  
understand others

3. Interoperability flow  
use your (and others) multimodal -data

### Summary:

- Local first
- Auditable & Findable
- Cryptographically verifiable (integrity & authenticity)
- Portable (including interoperable)
- Reusable
- Govern



### 2.3.2.2 Dissemination of Overlays: The Overlay Registry

OCA specifies the Base Overlays that provide a universal basis of core semantic tasks. The Community Overlays are defined by a specific ecosystem (i.e. community) and need to be made accessible according to the rules sets by the community. This is the purpose of an **Overlay Registry**, a list of Overlays Files that records specific Community Overlays and made accessible to the community.

The Human Colossus Foundation maintains an open-source Overlay Registry available on the Foundation GitHub repository<sup>5</sup>.

The NextGen project will first use the available Overlays. New NextGen specific Overlays will be stored in the dedicated Overlays Registry on the NextGen Test GitHub until a final repository is created as part of the sustainable NextGen platform.

### 2.3.2.3 Creation of the OCA File

The OCA File is a text file to be ingested into the OCA Repository. Text files is a simple format that ensures different tools can be developed to generate an OCA File depending on the use case. Text files are also easy to store and somehow future proof in the sense that they are simple and

<sup>5</sup> <https://github.com/the-human-colossus-foundation/overlays-repository/tree/main>

easy to parse. We illustrate this in the demonstration of some tools that can and could be used in NextGen.

### Example: OCA file for a .VCF

VCF, Variant Call Format [VCF2024] is a standard format used in different research workflows in genomic research (e.g. GWAS). VCF is very expressive, accommodates multiple samples, and is widely used in the community. The format is maintained by Large Scale Genomics work stream of the Global Alliance for Genomics & Health (GA4GH) and accessible on their GitHub repository<sup>6</sup>.

We display here a few extracts of an OCA File for encoding the semantic for a .VCF file. Examples are available to NextGen participants on NextGen Test GitHub repository.

#### Header and attributes

Define the set of Overlays used the name of the file and the set of ATTRIBUTE in the Capture Base

```
--name=vcf
--overlay-set=core

ADD ATTRIBUTE fileformat=Text fileDate=DateTime source=Text contig=Text filter=Text
info_id=Text info_number=Numeric info_type=Text info_description=Text
format_id=Text format_number=Numeric format_type=Text format_description=Text
chrom=Text pos=Numeric id=Text ref=Text alt=Text qual=Numeric record_filter=Text
record_info=Text record_format=Text
```

#### Label Overlays

The Label Overlay defines the label attached to each attributes for a specific language. This Overlay can be defined in any language.

```
ADD OVERLAY Label
language="en"
attribute_labels
fileformat="VCF Version"
fileDate="File Date"
source="Source Software"
contig="Contig (Chromosome)"
filter="Filter Definition"
info_id="INFO Field ID"
info_number="INFO Field Cardinality"
info_type="INFO Field Type"
info_description="INFO Field Description"
format_id="FORMAT Field ID"
format_number="FORMAT Field Cardinality"
format_type="FORMAT Field Type"
format_description="FORMAT Field Description"
chrom="Chromosome"
pos="Position"
id="Variant ID"
ref="Reference Allele"
alt="Alternative Allele(s)"
qual="Quality Score"
record_filter="Record Filter"
record_info="Record INFO"
record_format="Record FORMAT"
```

#### Sensitive & Entry Code Overlays

These Overlays define which attributes are sensitive and the possible entry code allowed for the attributes "contig", "filter", "info\_type", "chrom", and "record\_filter"

<sup>6</sup> <https://github.com/samtools/hts-specs/tree/master>

```

ADD OVERLAY Sensitive
  attributes=["id", "record_info", "record_format"]

ADD OVERLAY Entry_Code
  attribute_entry_codes
    contig=["chr1", "chr2", "chr3", "chr4", "chr5", "chr6", "chr7", "chr8", "chr9",
           "chr10", "chr11", "chr12", "chr13", "chr14", "chr15", "chr16", "chr17",
           "chr18", "chr19", "chr20", "chr21", "chr22", "chrX", "chrY"]
    filter=["PASS", "LowQual", "q10", "s50"]
    info_type=["Integer", "Float", "Flag", "Character", "String"]
    format_type=["Integer", "Float", "Character", "String"]
    chrom=["chr1", "chr2", "chr3", "chr4", "chr5", "chr6", "chr7", "chr8", "chr9",
           "chr10", "chr11", "chr12", "chr13", "chr14", "chr15", "chr16", "chr17",
           "chr18", "chr19", "chr20", "chr21", "chr22", "chrX", "chrY"]
    record_filter=["PASS", "LowQual", "q10", "s50"]
  
```

### Demonstration: Manual vs inference of an OCA file

OCA File is a text file can be created manually by using a predefined Domain Specific Language (see OCAFile specification<sup>7</sup>). In practice information on data is readily available and can be used to infer the structure. Therefore the OCA Files will often be generated through various inference mechanisms..

These mechanisms leverage different tools to infer semantics (e.g. extracting the header of a .CSV file) based on predefined functionality. The inference framework built into the OCA tooling is extensible, **allowing researchers to develop custom semantic inference logic and share it with others through community-driven efforts.**

This approach minimizes duplicated work and supports the creation of rich libraries of inference plugins that can generate semantics from well-known data structures, effectively bridging legacy datasets with the new data-plus-semantics ecosystem –FAIR principles in action.

As an example of such an inference mechanism, during the workshop we demonstrated how Large Language Models (LLMs) can assist in inferring semantics from unstructured datasets, significantly reducing the workload for researchers.

Example scan be found at: <https://vimeo.com/showcase/12027472><sup>8</sup>

#### 2.3.2.4 Dissemination of OCA File: OCA Repository & OCA Bundle

Once created, OCA Files are disseminated through the OCA Repository, which serves as the canonical storage and distribution layer for OCA artifacts. The repository enables OCA Files to be published, versioned, and discovered in a consistent and auditable manner. Rather than accessing raw or mutable OCA Files directly, downstream tools and applications consume OCA Bundles, which represent immutable snapshots of semantic definitions at a specific point in time.

An OCA Bundle encapsulates one or more OCA Files together with their associated overlays, metadata, and integrity information, forming a self-contained and verifiable unit. This snapshot-based packaging establishes the OCA Bundle as the general unit of semantic exchange and reuse across the ecosystem. By relying on OCA Bundles as stable inputs, downstream tools, pipelines, and applications can operate deterministically, ensuring reproducibility, traceability, and long-term interoperability, even as underlying semantic definitions continue to evolve in the repository.

<sup>7</sup> <https://oca.colossi.network/specification/ocafilename.html>

<sup>8</sup> Link as of time of writing.

## Demonstration: OCA repository

The OCA Repository is a REST-based service for

- managing,
- storing,
- sharing,

OCA artifacts such as OCA Bundles, Capture Bases, and Overlays. It supports OCA Files for bundle creation and provides a consistent **interface for publishing and retrieving semantic objects**.

OCA Repository can be run locally, in federated or centralised manner, due to its **immutable storage and deterministic mechanism** there is no risk of losing or modifying objects without noticing it. REST API allows for easy integration with any existing system.

Fully detailed instruction how to use OCA repository with examples can be found at: <https://oca.colossi.network/ecosystem/oca-repository.html>

### 2.4 SDK<sup>99</sup> components for semantic

The demonstration environment is delivered as a fully self-contained, Docker container that can be deployed and run locally by each researcher. The executables provided will lead to a SDK (Software Development Kit). The containerized setup is published via the NextGen GitHub Test Repository (restricted access) and is designed to operate without external service dependencies or complex configuration. This enables researchers to start working with their own data, semantic definitions, and MMIO structures in their own protected environment.

Beyond experimentation, the executables contributes to develop a collaborative feedback mechanism. Researchers can use the environment to explore semantic enrichment workflows and provide practical feedback on tooling, APIs, and usability. Insights gathered from this process are used to refine the underlying components and to extract reusable libraries that can be integrated directly into existing research workflows. These libraries are intended to support common environments such as Python (e.g., bindings suitable for Jupyter notebooks) as well as other platforms where researchers wish to embed semantic and MMIO-based enrichment processes into their data pipelines.

<sup>99</sup> SDK = Software Development Kit

### 3. Data-Level Compatibility



This section outlines NextGen’s contribution to advancing data-level compatibility within the consortium. Central to this effort is the Multi-Modal Integration Object (MMIO), a core architectural component, embracing the decentralised semantic architecture introduced in section 2, and designed to enable seamless integration, exchange, governance, and provenance tracking of heterogeneous multimodal datasets and models across diverse domains and platforms.

**Redefining Data-Level Compatibility in NextGen:** In NextGen, we have introduced a novel distinction: **Semantic Compatibility** that is treated as a separate, foundational dimension. As detailed in section 2, the OCA architecture support OCA Bundles —enhanced data schemas which preserves data meaning, context, structure, and integrity at the semantic layer.

This allows **Data-Level Compatibility** in NextGen to focus specifically on **syntactic alignment** — ensuring that datasets, regardless of origin, format, or storage system, can be exchanged and processed across platforms and standards without structural or format-related friction. In NextGen this refers to the development of compatibility tools to bridge common standards used in the consortium (e.g. FHIR, CDISC, OMOP)

To achieve this, we introduce the Multi-Modal Integration Object (MMIO) a complementary mechanisms to the decentralised semantic architecture. By explicitly decoupling semantic and syntactic compatibility, the MMIO enables a more precise, scalable, and robust approach to cross-governance portability. This architectural innovation goes beyond current standards, allowing for true end-to-end compatibility — from meaning (semantic) to machine execution (syntactic).

#### 3.2 Enhancing Data-Level Compatibility in NextGen: The Role of the Multi-Modal Integration Object (MMIO)

The MMIO is a syntactic harmonization container for any modality (dataset). Its verifiable integrity ensures that file formats, schemas, and structural conventions are cryptographically accurate — machine-verifiable even in multi-jurisdictional, multi-stakeholder environments. As such, users of MMIOs can precisely identify the content and determine the specific harmonization process required. Examples are provided in section 3.1.3

##### 3.2.1 MMIO in NextGen

*“The Multimodal Integration Object (MMIO) is a self-contained digital envelope designed to hold disparate data modalities — a standardised, secure, privacy-preserving semantic container whose integrity can be verified.”*

**What is MMIO ?** The Multimodal Integration Object is a digital container designed to encapsulate, describe, and govern any type of data — from genomic sequences and ECG waveforms to clinical variables, catalogue information and imaging files — while ensuring interoperability, privacy, and controlled access across systems and jurisdictions.

The introduction of MMIO in data management mirrors the introduction of shipping containers in global maritime trade. Like a shipping container with goods, the MMIO can hold any schema or data — but does not itself transport data or process it. Once delivered, the recipient knows exactly how to handle the MMIO for a given purpose.

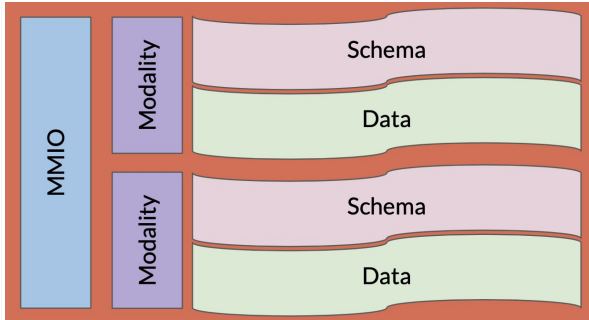


Figure 3.1 MMIO as an envelope for modalities.

It is not a database. It is a portable, self-describing object that travels with its own modalities representing data, metadata, rules, model, etc..

An MMIO can contain any number of modalities semantically described by an OCA Bundle (schema). Modalities as data records are commonly only referred to, not included allowing to leverage different privacy enhance technologies (PETs). An MMIO can be cryptographically attached to a controller to ensure authentic provenance.

**Why MMIO Matters for cardiovascular research.** Cardiovascular research increasingly relies on multimodal data — combining genomics (multiomics), imaging, EHRs, wearables, and comprehensive study phenotypes (phenomics) including patient sensitive information. But these data often live in silos, in incompatible formats, under different governance regimes (i.e. the integration barriers addressed in NextGen). The MMIO addresses these challenges through the three core design principles encoded in its name:

- **Multi-Modal:** Supports any modality type: –omics, -images (DICOM, PNG), –tabular / non-tabular data, catalogues (metadata), vocabularies, algorithms, models.
- **Integration:** MMIO exists to integrate modalities — not just to store or move data, but to make them analytically compatible across systems, formats, and institutions.
- **Object:** Treated as a first-class digital object — each MMIO is represented by cryptographic identifier which can be used to track it's identity, provenance, and behaviour . The MMIO is a self-contained object that “knows” what it is and the contextual information on how it may be used.

**MMIO Key functionalities:**

- **Adaptor Functionality:** Research multimodal datasets are composed of a set of modalities which may be stored in different formats or have structural variations. Interoperability requires that a conversion mechanism might be required if the modality original representation is not compatible with the representation of the task (e.g. algorithm, ML) to be undertaken or joint analysis. The MMIO facilitates this adaptor functionality by encapsulating the required information (e.g. metadata, types, purpose of use) define the structure and access each modalities. The required conversions are not part of the MMIO itself. For example, NextGen develops the Research Toolkit (RTK) concept where certain conversions (excluding OMOP/CDISC/FHIR) can be performed. At M24, the RTK is at the early concept and development stage. The RTK is also planned to include multimodal functionality including ways to address missing modalities.
- **Provenance Functionality:** A lack of traceability undermines reproducibility and trust, especially when combining data from multiple sources. The MMIO deterministic properties allows to create records of the origin, version, modifications, and actors involved in handling each modality. Changes can be cryptographically signed to ensure data integrity and authenticity. As a result, the MMIO with provenance chain provides comprehensive audit trails to support quality management and regulatory compliance (e.g., GDPR, HIPAA).
- **Governance Functionality:** Integrating data from different institutions often raises legal, ethical, and technical challenges related to consent, jurisdiction, and usage rights. An MMIO can embed usage policies (e.g., “for GWAS only,” “no commercial use”) using specific OCA Overlays cryptographically bound to the OCA Bundle defining the modality’s semantics. It can also incorporate modality-specific governance constraints such as jurisdictional boundaries (e.g., “data may not leave the EU”), consent records, or organizational permissions.

The following sections provide additional technical information on the MMIO itself, while its applications are described in other deliverables. In NextGen, the first application of the MMIO is within the discovery mechanism, where it is used to index datasets for subsequent searches (see Deliverables D1.3 and D1.4 “Data Discovery Functionality”). NextGen applications of the MMIO within NextGen tooling—such as the accelerated GATK pipeline, privacy-preserving federated learning, and transformer-based methods—will be developed based on input from WP5. Research partners will provide relevant use cases and pilots for cardiovascular personalised medicine. WP6 (Regulation, Ethics, and Governance) and WP7 (Broader Engagement & Exploitation) contribute to the governance functionality.

MMIOs are integrated into the Pathfinder platform (see D2.1 “Platform Design and Architecture Blueprint”), developed by WP2 (Infrastructure and Platform), demonstrating their value within WP5 (Pathfinder Project & Pilots).

### 3.3 Creation of MMIOs

In this deliverable, we describe the creation of the MMIO during the development phase of NextGen as of December 2025. The Part 2 of this deliverable, due in December 2026, will provide additional information on the deployment of the MMIO into specific NextGen pilots.

In NextGen, MMIO’s can be created either by users of the Pathfinder platform or locally through specific libraries. For the development of the project, a private GitHub repository has been created for the purpose of testing the development of the components for NextGen pilots. The aim of this repository is to create a collaborative space for the development of semantics and MMIOs. In addition to the *oca* binary already introduced in the Semantic Compatibility section, the test repository includes:

- *m2io* binary: Command Line Interface (CLI) helping with creation and serialization of the MMIO
- *search* binary: As an example of use of MMIO (see [D1.4-M24], “Data Discovery Functionality” for more details)

#### Binary delivered: *m2io*

```
Multimodal Integration Object CLI

Usage: m2io <COMMAND>

Commands:
  create  Create a new MMIO object
  parse
  said
  help    Print this message or the help of the given subcommand(s)

Options:
  -h, --help  Print help
```

MMIOs can be easily created using the *m2io* CLI which allows to process modalities of any type and bind them with provided semantic. The binary composes the MMIO object, calculate the identifier of every object and serialise an MMIO as a JSON object –a format ready for further processing steps, verification or sharing.

```

Create a new MMIO object

Usage: m2io create [OPTIONS] --output <OUTPUT>

Options:
  --modalities [<MODALITIES>...] Specify a modality using: file=path,bundle_said=<SAID>. Repeat for multiple modalities.
  -o, --output <OUTPUT>
  -h, --help Print help
    
```

### 3.3.2 Encapsulation of modalities

Each modality is represented as SAID (self-addressing identifier) of the data record and is cryptographically bound to a schema (OCA Bundle) via calculated digest, a content-based identifier of the MMIO.

```

{"version": "0.1",
 "digest": "EhK9G_UiPzUadrDhDgn3AFqLsg6n8evRxra-mR_iDXgw",
 "modalities": [
  {
    "digest": "EcXkbK8LQIJ0kKPEJzI17fRRVvBHxDYwFXw02hfo-dNF",
    "modality_said": "EK8TSbn1-aGmoEBN6jbcyUmbyyXJrcEST8yak8rKHZj1",
    "modality_type": "Text",
    "media_type": "application/json",
    "oca_bundle": {
      "type": "Reference",
      "value": "EEzne0qXG0WdLcwoSy6fQdwQ9TdzQqq6LKZm_4YwHir"
    }
  },
  {
    "digest": "EBIqX56eni1aKq34yg0gE8yuCr76gxm9J_TyUmQ0HD_a",
    "modality_said": "EBnZDZpPXXUqaPgbUvz0EmxbS6GlpwyHKVr7UADNt7oh",
    "modality_type": "Binary",
    "media_type": "application/octet-stream",
    "oca_bundle": {
      "type": "Reference",
      "value": "EEzne0qXG0WdLcwoSy6fQdwQ9TdzQqq6LKZm_4YwHir"
    }
  }
 ]
}
    
```

Figure 3.x: Example of a 2-modality MMIO in JSON serialisation

### 3.3.3 MMIO's library

The MMIO libraries expose interfaces to various development environments, enabling researchers to integrate its functionality into their preferred tools. The library is implemented in the Rust programming language, which allows its functionality to be exposed via Foreign Function Interfaces (FFIs) to other languages — such as Python or JavaScript — with minimal effort. Through seamless integration, researchers gain the flexibility to incorporate MMIO functionality into their existing workflows and data pipelines.

The MMIO properties are defined in the following subsection, MMIO properties, and their application to specific use cases is elaborated in the Purpose subsection.

## 3.4 MMIO properties

We list here a few of the properties of an MMIO made available to the Pathfinder platform. The properties listed can also be available locally but assume that the MMIO is used in an environment with supporting infrastructure integrating OCA and DKMS.

MMIO properties:

### 1. Content Addressability

The MMIO is **fully content-addressable**. Each MMIO object is assigned its own SAID (Self-Addressing Identifier). Within the object, each modality is individually addressable. It has a:

- SAID for the modality descriptor;
- SAID for the modality data blob.

Content **integrity is cryptographically enforced**: any modification — whether to content or structure — results in a change to the corresponding SAID.

**MMIO objects are immutable by design**: they are verifiable, not mutable. Any update or revision requires the creation of a new MMIO object.

## 2. Structural Properties

The MMIO structure is **deterministic**: **Field ordering is fixed** to ensure canonical JSON serialisation. **Canonical hashing** is applied to guarantee structural consistency.

It adheres to a **minimal core schema** and **enforces modality-level encapsulation** — ensuring each modality is self-contained within the envelope.

## 3. Semantic Integration (OCA)

Each MMIO object is associated with **one or more OCA Bundles**. **Linkage is flexible**: An OCA Bundle may be referenced by: SAID (REFERENCE). Alternatively, it may be embedded directly within the MMIO (OCA\_BUNDLE).

## 4. Conceptual Properties

The MMIO functions as a **multimodal envelope**, capable of encapsulating multiple heterogeneous data modalities — of any data type — within a single, unified object. Its **design is semantic-first**: every modality is explicitly bound to semantic metadata via OCA Bundles. It is **governance-aware**: supports attachment of purpose, consent, usage rules, and other governance-related semantics through OCA. Finally, it is **portable and interoperable**: designed for seamless exchange across systems without dependence on location-based references (e.g., URLs or file paths).

### Multi-Modal Integration Object (MMIO) Functionalities

MMIO Functionalities:

- Encapsulation
- Integrity Verification
- Semantic Resolution
- Governance & Policy Enforcement
- Interoperability
- Composability
- Extensibility
- Trust & Auditability

### 3.4.4 MMIO indexing

The MMIO provides a powerful indexing mechanism. It is at the centre of the 2-level privacy respecting search of NextGen’s discovery mechanism [D1.3, D1.4] where the first level search is limited to rich semantic search the level two search allows to involve data records themselves. Searching on the data structure provides information about the existence of data without revealing the underlying data records which can simplify sharing and discovery mechanism.



Figure 3.x: MMIO for search

Leveraging properties of the MMIO we can introduce the various types of search [D1.3-M12] for different purposes, including but not limited to:

- **structural search** - search only through semantic without revealing the data records
- **provenance search** - search through data lineage and it's sources
- **term search** - search through predefined terms encapsulating different meaning across different standards and ontologies
- **value search** - search specific data records and/or it's attributes

#### 3.4.5 Portability: MMIO as an accurate data container

The content-addressable identifier of the MMIO allows for the creation of a provenance and authentication mechanism. This mechanism provides secure and unique access to the MMIO as a whole or its sub-elements. It precisely controls access on semantic and/or data level no matter where the MMIO would be shared as it does not depends on specific platform or network.

Using decentralised key management infrastructure (DKMI) based on DKMS we are able to provide verification mechanism across platforms and/or ecosystems, increasing portability of such object without sacrificing it's security and privacy properties. To create authenticity for MMIO we are using Authentic Chain Data Containers (ACDC) it's flexibility and simple construction allows to digitally sign and verify any type of the data sets no matter how big or complex they are.

The compatible chaining mechanism of both MMIO and ACDC allows to map any existing governance structure into digital ecosystems allowing anyone to verify anywhere not only integrity of the provided MMIO but as well it's provenance (where comes from) and governance (what you are allow to do with them).

Both MMIO and ACDC support rich configuration of privacy enhanced technologies (PETs) like selective disclosure and selective authentication, giving researcher/owner of the data sets to precisely control what is visible and accessible to whom mitigating risk related with unauthorized exposure and access to the data sets and it's semantic.

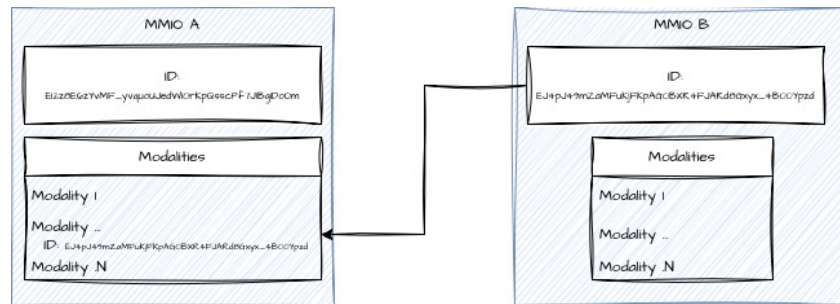
#### 3.4.6 Selective Disclosure

An MMIO provides out-of-the-box *selective disclosure*, a key property for privacy enforcement. As the MMIO can hold only identifiers to semantic and data records (modalities), the content behind this identifier does not have to be revealed. A robust selective disclosure mechanism can be implemented by the ecosystem governance. The selective disclosure property can be used in conjunction with chaining mechanisms (see Chained Linkability functionality) for powerful integrity verification mechanisms.

#### 3.4.7 Chained Linkability

MMIO embeds a linkability property, an MMIO can be immutably linked to another MMIO. This enables the design of robust, complex chaining mechanisms.

Example:



MMIO Chaining mechanisms through self-addressing-identifiers

The chained linkability enables an immutable, cryptographically verifiable association between two or more MMIO instances. This allows for the construction of complex data sets and its dependencies — essential for regulatory compliance, reproducibility, and federated analysis in precision medicine.

**MMIO chained linkability example** (Cardiovascular Personalized Medicine — NextGen Project):

An MMIO representing a patient’s genomic variant profile (MMIO-A) is cryptographically linked to an MMIO containing their clinical phenotype data (MMIO-B), which in turn is linked to an MMIO representing a computational model output predicting drug response (MMIO-C). This chain ensures:

- Traceability of decision-making from raw data to clinical inference
- Auditability for regulatory review (e.g., FDA/EMA submissions)
- Governance enforcement (e.g., restricting access to raw genomic data while allowing access to derived risk scores)
- Reproducibility across institutions in multi-center studies

This example also demonstrates how the MMIO is used to encode adherence to organisational measures to be implemented (NextGen WP6, T6.4) in a multi-jurisdictional environment. In the scenario above, the NextGen ecosystem or its participants could enforce the following specific compliance requirements:

- Data Provenance and Integrity in Health Data: NISTIR 8402
- Information Security Management: ISO/IEC 27001:2022— Clause 9.2: Monitoring and Review of Controls
- Resource Provenance and Audit Trail: HL7 FHIR R4
- Data Use Ontology: GA4GH DUO for governance-aware linking (discovery, access)

**3.4.8 Advanced authentication**

Not a property of the MMIO itself but a property of its designed around content based Self-Addressing Identifiers (SAID), the MMIO and its chaining property is a robust base compatible with ACDC (Authentic Chained Data Container) a data model used to represent verifiable, linked, tamper-evident digital information. While OCA focuses on semantics (structure + meaning of data through overlays), ACDC focuses on trust, ensuring that data:

- ✓ is authentic
- ✓ is cryptographically verifiable
- ✓ can be chained to other data objects
- ✓ supports fine-grained disclosure
- ✓ works with modern decentralized identity ecosystems

### 3.4.9 MMIO for Pathfinder Functionalities

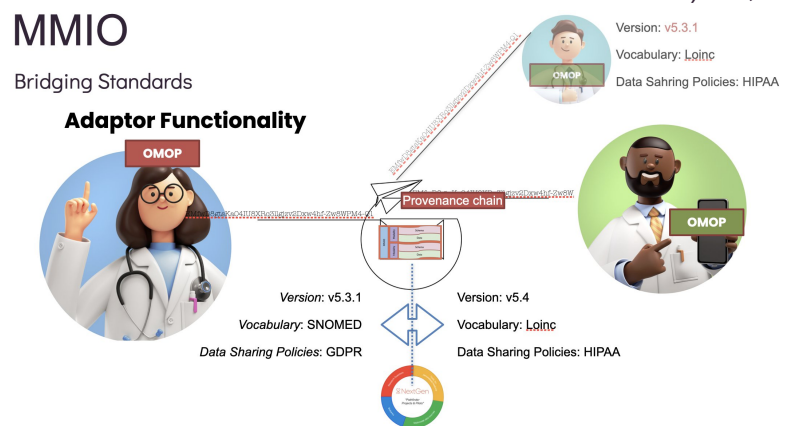
MMIO enables advanced mechanisms for data governance, discoverability, transformation, provenance tracking, cross-domain interoperability. In the NextGen architecture, MMIO properties are engineered for deployment within the Pathfinder platform using a DKMI® (Decentralized Key Management Infrastructure) for decentralised authentication. This infrastructure serves as the operational engine for MMIO functionality, underpinning three core functionalities (adaptor, provenance, governance).

We provide here a brief description of the MMIO functionalities from the cross-platform compatibility perspective. Additional information can be found in the deliverable D1.4 Discovery Mechanism Part 2 and D5.3 PathFinder functionality demonstration.

#### 3.4.9.1 Pathfinder Adaptor

The MMIO’s adaptor functionality addresses the challenges of data standards mismatch and fragile bridges between models. It provides a mechanism that let a researcher to expose complex datasets that can be reused by other researchers (or any other users with a role within the research workflow). Each party (i.e. the creator of the MMIO and the receiver of the MMIO) will, in general operate in different context. Even within the same context, minutes differences might impact the workflow.

For example if a group of researchers exchange data sets the OMOP<sup>10</sup> common data model differences in version, vocabulary used, data sharing policies, and others are bound to appear (see figures on the right). Instead of requiring everyone to adhere to one given precise data model, the approach is to let the creator of the exposed multimodal data set provide the contextual metadata with a layered semantic architecture that can describe precisely the dataset, its context and purpose of use. Then, on the receiver side, the system can detect the differences and apply an adaptor specific to the context. The adaptor itself is not part of the MMIO but the MMIO provides all the information to resolve (i.e. find) it if it exist or raise a flag if none are known.



Instead of requiring everyone to adhere to one given precise data model, the approach is to let the creator of the exposed multimodal data set provide the contextual metadata with a layered semantic architecture that can describe precisely the dataset, its context and purpose of use. Then, on the receiver side, the system can detect the differences and apply an adaptor specific to the context. The adaptor itself is not part of the MMIO but the MMIO provides all the information to resolve (i.e. find) it if it exist or raise a flag if none are known.

This approach has the advantage the the creator is responsible to generate one MMIO. Different receivers will have different adaptor needs. The example above shows that the MMIO enable complex workflows requiring multiple adaptors. In this case:

- *Version adaptor*: specific to OMOP development road map. Once done is valid for any users;
- *Vocabulary adaptor*: could be algorithmic if ontologies allow.
- *Data Sharing Policies*: Governance related. Require specific agreements. Can be automated if previous agreements have been established at consortium level.

The definition of the adaptor functionality in the Pathfinder platform integrating MMIO is described in deliverable [D5.3-M24] and will be developed in the second part of the project.

#### 3.4.9.2 Provenance & Information Discovery in NextGen

The MMIO’s provenance functionality addresses the challenges of data discovery, lawful access. An MMIO is uniquely identified by a content based identifier. This enables the NextGen system to bind the MMIO to the identifier of the creator thus making a strong link of its origin. But, due to the structure of the MMIO, the provenance of each modalities contained in the MMIO can also be

<sup>10</sup> Observational Medical Outcomes Partnerships, see [Rosenbloom2017]

provided. In other term, within the NextGen ecosystem the receiver of an MMIO knows not only the creator of the MMIO but also the provenance of its content, the modalities.

In the OMOP example given above the label “Provenance Chain” in the figure indicates that the MMIO can provide the provenance chain of its modalities as long as the governance allows it.

### 3.4.9.3 Discovery Mechanism: indexing of MMIO

MMIO can be created for information discovery (catalogue). It allows the creator to precisely describe the information that can be exposed to a reasoning system (e.g. Indexing mechanisms). Subsequently the search of information in the ecosystem can be done purely on the semantic (i.e. No data records are actually exposed) before a data record level search is actually carried out.

Additional information on the use of MMIO for advanced search mechanism can be found in the deliverable [D1.3-M12] and [D1.4-M24]

By governance we refer to the rules of usage that can be enforced with an MMIO. These rules can be encoded at different level. First at the semantic level, the MMIO modalities are described by an OCA Bundle that itself can contain specific Overlays enforcing a specific rule. For example an organisation can require that certain attributes are tagged SENSITIVE if the dataset is exposed externally. In the multi-jurisdictional setting of NextGen, this allows for example to define different semantic structure for the *same underlying data records*. This allows for example the research organisation to enforce at the schema level different rules for EU and Non-EU jurisdiction. An attribute visible (i.e. Searchable) within EHDS might not be accessible outside EHDS.

## 4. Other dimensions of cross-platform technical compatibility

Semantic and data-level compatibility introduce core elements that can be used across all other compatibility dimensions. At M24 these core elements are ready for testing across the NextGen landscape. As a result, they will contribute to the other cross-platform compatibility dimensions that are addressed in the project.

Part 2, due at M36 (December 2026) will therefore focus on the other three compatibility dimensions.

### 4.1 Hardware and Runtime Compatibility

Hardware and runtime incompatibility arises when identical analytical pipelines yield non-equivalent results due solely to differences in computational system and execution environments.

There are no general solution to closing a hardware compatibility gap. In-depth benchmarking to quantify this compatibility is one approach. The accelerated GATK secondary analysis pipeline developed in Nextgen will be accelerator independent and open source. This will provide significant improvement for research organisations that will not be required to rely on specific proprietary hardware. The pipeline will aim to produce results identical to the widely-used CPU-based GATK solution to retain compatibility with use case pipelines built on top of GATK.

In this context, we investigate the possibility to perform local benchmarking at the local level (like the result of an individual tool calling), at the pipeline level (like the variants that are produced by optimized tool), and at the global use case level (such as establishing if there are differences in actionable insights between the original tools and those developed by NextGen). This will allow an optimisation of the accuracy to close the compatibility gap.

### 4.2 Model-level Compatibility

Model compatibility refers to the coherence of the model assumptions used in a research workflows. Models assumptions must be clearly labeled, particularly in multi-stakeholder environments where these must be clearly communicated to avoid inconsistencies in data inputs and variability in data output [Chahal2025aa].

In NextGen we aim to offer the possibility to enrich model description with machine verifiable labeling with the OCA architecture. The blueprint of this approach in the framework of machine learning takes the following form:

*The problem statement:*

- The traditional software engineering notion of compatibility between different programs/classes/modules only considers the form of their inputs and outputs, such as dimensions/types/domain of validity.
- This notion of compatibility is too limited for ML software. ML software is a materialisation of probabilistic models, which come with their own assumptions about the distributions of their inputs and outputs (in addition to the dimensions/types/domain of validity).
- The software itself has no visibility into the probabilistic properties and assumptions of the ML models it implements. ML engineers are left with the task of ensuring the consistency of data handling across programs/classes/modules in complex data analysis pipelines, which is suboptimal and error-prone.

*What we aim to do is*

- to allow ML software itself to be able to check the probabilistic properties of the ML models it implements, their inputs and outputs at runtime.

*How we aim to do that is*

- to augment data objects with enough additional metadata to allow programmatic checks within the software at runtime.
- add such metadata via overlays from a technical standpoint.

### 4.3 Regulatory and Workflow compatibility

Regulatory & workflow compatibility is one of the most structurally decisive dimension of interoperability. Not handled properly, it simply vetoed the access to data. Unlike the other dimensions (data, semantic, model, hardware), it is not primarily technical. As defined here, it includes **governance, traceability, authorization, consent, auditability, and workflow alignment**. In multi-site, multi-jurisdictions, failing to meet compatibility requirement might significantly impact, possibly forbid, research workflows.

If not adequately addressed, regulatory barriers can significantly slow down research activities, increase costs, or even prevent access to data.

The NextGen approach starts with a set of relevant use cases in personalized cardiovascular medicine. The data integration tools developed in the project are applied to these use cases — where they have impact — and are the basis of a set of pilots part of the Pathfinder project

Requirements for compatibility across regulatory regimes and workflows includes:

- **Addressing different regulatory regimes governing identifiers, data flow, and data use.** This includes for example considering GDPR, HIPAA, EMA regulations and participants institutional policies.
- **Meeting operational workflows constraints.** This refers to how data is collected, labeled, consented, transformed, de-identified, shared, etc..
- **Consistent meaning of regulatory, ethics and governance terminology across institutions.** Audit trails, provenance, and accountability are preserved across chains of custody within and across jurisdictions.

Researchers face REG barriers that slow/prevent access and use data in their workflows. Within the NextGen project the regulatory and workflow compatibility dimension is addressed from a governance angle. Specifically in task T6.4 “*Technical and organisational measures in data governance*” we experiment how the use of the semantic layer and enriched metadata could facilitate the access to data.

### 4.4 Status as of M24

As of December 2025 (M24) the following parallel development that contribute also to improve regulatory and workflow compatibility have taken place (and will continue in 2026).

- A first mapping of the overall **regulatory landscape** has been done, including the local regulatory environments of each research organization participating in NextGen and is updated as necessary for the NextGen project.
- A first version of the distributed data oriented architecture in the form of the Pathfinder platform and including the first services has been demonstrated and continues to evolve.
- The semantic compatibility tooling (OCA) and a first versions of MMIO has been demonstrated to initiate an iterative feedback loop for further developments.

The Regulation, Ethics and Governance (REG) Board maintains oversight of activities relevant to NextGen as specified in its Terms of Reference.

#### Application: T6.4 Technical and organisational measures in data governance

The task T6.4 (HCF,HIRO,HL7, QMUL, MyData<sup>11</sup>) was designed to leverage in the context of governance the opportunities introduced by the different data integration tools developed in NextGen. Among these, the data management tools detailed in this document (OCA, MMIO) bring

<sup>1111</sup> MyData left the consortium at M18 but was deeply involved in the T6.4 work done in the M1 to M18 period.

new opportunities. The aim is to provide innovative approaches on two dimensions:

- *Internally to NextGen consortium:* use the OCA to provide granular contextual details on the contextual (e.g. jurisdiction), purpose of use, and governance of data in their research workflows. The MMIO being at the core of the interoperability mechanism for multimodal datasets.
- *Externally to NextGen consortium:* First, use OCA to demonstrate how compliance to EHDS requirements<sup>12</sup> can be facilitated with machine-actionable measures. The project will experiment this approach with work done on EHDS data quality requirements, diversity index, sex/gender inequality, and other specific topics brought by participants of the Regulatory, Ethics and Governance work package. Second, OCA, MMIOs and DKMS will be used to investigate the technical compatibility with other EU or genomic initiatives.

## 5. Conclusions

### Cross-Platform technical compatibility in Pathfinder

NextGen's approach to cross-platform compatibility will be piloted (TRL 3 to 4) within the Pathfinder project regrouping five research organisations around a platform piloting the tools developed in the project.

As of M24 (December 2025), the Pathfinder platform design and initial implementations have been documented (see [D2.1-M18, D2.2-M18]). The infrastructure will support:

- A FAIR-compliant data ecosystem (NextGen Data Management Plan [D1.1-M6, D1.3-M24]),
- A federated machine learning framework ([D3.1-M12, D3.2-M18]),
- A data discovery mechanism (catalogue) ([D1.3-M12, D1.4-M24]).

This document (Part 1) provided the foundational data management elements introduced by NextGen to ensure semantic (OCA) and structural data interoperability (MMIO) within Pathfinder. The tools demonstrated will be incrementally tested and integrated into Pathfinder pilots (see [D5.3-M24] for current functionality status) throughout 2026–2027 and reported in Part 2 (M36) and 3 (M48).

In this work we concentrated on the data management tooling developed in WP1 with two specific section detailing the outcomes of the first stage. With the objective to develop semantic interoperability and advanced integration tools facilitating data integration and interoperability in a multi-jurisdictional context, two data management innovations, the Overlays Capture Architecture and the Multimodal Integration Object have been documented. Together, they provide the core components, based on FAIR principles, to build re-usable cryptographically verifiable data structure (e.g. OCA Bundles).

These components will be tested for integration with NextGen participants from January 2026 onwards through a dedicated private GitHub repository.

In 2026, the cross-platform technology compatibility WP1 work will concentrate on the integration with other tooling developed in NextGen and how they impact one or more compatibility dimensions presented in this part 1.

<sup>12</sup> or other related EU initiative, genomic, and open-data initiative as project resources will allow.

## 6. References

Selected references used in this document.

- [Chahal2025aa] A.Chahal et al. “Data Interoperability and Harmonization in Cardiovascular Genomic and Precision Medicine”, *Circ.Genom Precis Med.*2025;DOI: 10.1161/CIRCGEN.124004624
- [Curran2024] S.Curran "Overlays Capture Architecture (OCA) for Aries", LINUX Foundation Hyperledger Aries, RFC0755. <https://identity.foundation/aries-rfcs/latest/features/0755-oca-for-aries/>
- [D1.1-M6] “Data Management Plan v1.0”, NextGen Consortium public deliverable D1.1 June 2024, available at <https://www.nextgentools.eu/wp-content/uploads/2025/02/NextGen-D1-1-Data-Management-Plan-1-0-1.pdf>
- [D1.2-M24] “Updated Data Management Plan”, NextGen Consortium public deliverable D1.2 December 2025,
- [D1.3-M12] “Data discovery functionality demonstrated-1”, NextGen Consortium deliverable D1.3 Dec. 2024, available at <https://www.nextgentools.eu/wp-content/uploads/2025/02/NextGen-D1-3-Data-Discovery-Functionality-Part-1-1.pdf>
- [D1.4-M24] “Data discovery functionality demonstrated-2”, NextGen Consortium deliverable **D1.4 Jan. 2026, to be publicly available after consortium review.**
- [D3.1-M12] “Technical methods for federated genomic analysis”, NextGen Consortium deliverable D3.1 Dec. 2024,
- [D3.2-M18] “Implementation of federated genomic analysis-1”, NextGen Consortium deliverable D3.2 June 2025,
- [D5.1-M6] “Pathfinder frameworks and pathways”, NextGen Consortium deliverable D5.1 June 2024,
- [D5.2-M12] “Pathfinder frameworks specification”, NextGen Consortium deliverable D5.2 Dec. 2024,
- [D5.3-M24] “Pathfinder functionality demonstration-1”, NextGen Consortium deliverable **D5.3 Jan. 2026,**
- [Epps2022] Van Epps H, Astudillo O, Del Pozo Martin Y, Marsh J (2022) The Sex and Gender Equity in Research (SAGER) Guidelines: Implementation and Checklist Development. *European Science Editing*, 48: E86910. <https://doi.org/10.3897/ese.2022.e86910>
- [FAIR2016] Wilkinson, M. et al. “The FAIR Guiding Principles for scientific data management and stewardship” (2016) <https://www.nature.com/articles/sdata201618>
- [GA4GH] Official website <https://www.ga4gh.org>

- [HCF2023] Knowles,P et al. "OCA Technical Specification", <https://oca.colossi.network/specification/>
- [HCF2025] R.Mitwicki, M.Pietrus "OCA File Technical specification v1.0rc" OCA official website, <https://oca.colossi.network/specification/overlayfile.html>
- [HCF2025a] Human Colossus Foundation "Overlays Repository" (2025) THCF GitHub repository, <https://github.com/the-human-colossus-foundation/overlays-repository/tree/main>
- [Jacobson2022] Jacobsen, J.O.B., Baudis, M., Baynam, G.S. et al. "The GA4GH Phenopacket schema defines a computable representation of clinical data." *Nat Biotechnol* 40, 817–820 (2022). <https://doi.org/10.1038/s41587-022-01357-4>
- [SAID2022] Smith, S. "Self-Addressing Identifier (SAID)" (2022) <https://datatracker.ietf.org/doc/html/draft-ssmith-said>
- [SANDAK2020] Sandak,M.P. et al. "Presenting machine learning model information to clinical end users with model facts labels" *npj Digital Medicine* (2020) 3:41 ; <https://doi.org/10.1038/s41746-020-0253-3>
- [THEDAS2024a] TEHDAS Joint Action Report "Options for Governance Models for EHDS" January 2023. <https://tehdas.eu/app/uploads/2023/01/tehdas-options-for-governance-models-for-the-european-health-data-space.pdf>
- [THEDAS2024b] TEHDAS Joint Action Report "Recommendations on a Data Quality Framework for EHDS for secondary use" September 2023 <https://tehdas.eu/app/uploads/2023/09/tehdas-recommendations-on-a-data-quality-framework.pdf>
- [VCF2024] Large Scale Genomic Workstream "The Variant Call Format Specification V4.5"(2024) GA4GH <https://samtools.github.io/hts-specs/VCFv4.5.pdf>

## Appendices

### 4 Examples

We provide here a few examples mentioned in the body of the document

#### Example: OCA Bundle

This example...

```
{
  "bundle": {
    "v": "OCAS11JSON000646_",
    "digest": "EKHBds6myKVIIsQuT7Zr23M8Xk_gwq-2SaDRUprvqOXxa",
    "capture_base": {
      "digest": "EBnF9U9XW1EqteIW0ucAR4CsTUqojvfIWkeifsLRuOUW",
      "type": "spec/capture_base/1.0",
      "attributes": {
        "digest": "Text",
        "i": "Text",
        "passed": "Boolean"
      }
    },
    "classification": "",
  },
  "overlays": [
    {
      "digest": "ECZc26INzjxVbNo7-hln6xN3HW3e1r6NGDmA5ogRo6ef",
      "capture_base": "EBnF9U9XW1EqteIW0ucAR4CsTUqojvfIWkeifsLRuOUW",
      "type": "spec/overlays/label/1.0",
      "language": "en-UK",
      "attribute_labels": {
        "digest": "Schema digest",
        "i": "Credential Issuee",
        "passed": "Passed"
      }
    },
    {
      "digest": "ED6Eio9KG2jHdFg3gXQpc0PX2xEI7aHnGDOpjU6VBfjs",
      "capture_base": "EBnF9U9XW1EqteIW0ucAR4CsTUqojvfIWkeifsLRuOUW",
      "type": "spec/overlays/character_encoding/1.0",
      "attribute_character_encoding": {
        "digest": "utf-8",
        "i": "utf-8",
        "passed": "utf-8"
      }
    },
    {
      "digest": "EIBXpVvka3_4lheeajtitiafIP78Ig8LDMVX9dXpCC2l",
      "capture_base": "EBnF9U9XW1EqteIW0ucAR4CsTUqojvfIWkeifsLRuOUW",
      "type": "spec/overlays/information/1.0",
      "language": "en-UK",
      "attribute_information": {
        "digest": "Schema digest",
        "i": "Credential Issuee",
        "passed": "Enables or disables passing"
      }
    },
    {
      "digest": "EJSRe8DnLonKf6GVT_bc1QHoY0IQOG6-lqqu7pqVCU8",
      "capture_base": "EBnF9U9XW1EqteIW0ucAR4CsTUqojvfIWkeifsLRuOUW",
      "type": "spec/overlays/conformance/1.0",
      "attribute_conformance": {
        "digest": "M",
        "i": "M",
        "passed": "M"
      }
    },
    {
      "digest": "EOxvie-zslkGmFzVqYAzTVtO7RyFXAG8aCqE0OougnGV",
      "capture_base": "EBnF9U9XW1EqteIW0ucAR4CsTUqojvfIWkeifsLRuOUW",
      "type": "spec/overlays/meta/1.0",
      "language": "en-UK",
    }
  ]
}
```

```

"description": "Entrance credential",
"name": "Entrance credential"
}

```

## 2.0 List of Overlays

Table 8.1 OCA v2.0 Base overlays (in alphabetical order)

Overlay Name	Short Description
<b>ATTRIBUTE_MAPPING</b>	<b>Purpose:</b> Defines attribute mapping between two distinct data models. <b>Used for:</b> Data integration (including transformation)
<b>CARDINALITY</b>	<b>Purpose:</b> Defines the minimum and maximum values an attribute can take. <b>Used for:</b> Specifying validity ranges
<b>CHARACTER_ENCODING</b>	<b>Purpose:</b> Specify the text encoding for each attribute. <b>Common:</b> "utf-8" for all text fields. <b>Used for:</b> Legacy or specialised encoding (e.g. ASCII - 7-bit, ISO-8859-1, Windows-1252, UTF-16, UTF-32, GBK / GB2312, Shift_JIS, EUC-KR, KOI8-R, Big5)
<b>CONFORMANCE</b>	<b>Purpose:</b> Flags if an attribute is mandatory or optional <b>Used for:</b> Specifying which parameters are necessary and optional.
<b>ENTRY</b>	<b>Purpose:</b> Adds extra metadata to entry codes or classification systems.
<b>ENTRY_CODE</b>	<b>Purpose:</b> Defines allowed values for enumerations. <b>Used for:</b> Gender, status, categories, yes/no codes.
<b>ENTRY_CODE_MAPPING</b>	An Entry Code Mapping Overlay defines the entry key mappings between two distinct code tables or datasets
<b>FORMAT</b>	<b>Purpose:</b> Defines an input and display format for data fields. Enables conversion of the input buffer to the program variable and displays program variable data to form fields. <b>Used for:</b> Date formats, number formatting, patterns.
<b>LABEL</b>	<b>Purpose:</b> Defines human-readable labels for each attribute in different languages. <b>Used for:</b> Field names (e.g., "First Name", "Date of Birth").

<b>META</b>	<p><b>Purpose:</b> Provides high-level metadata about the entire schema.</p> <p><b>Used for:</b> Name, description, language, title, version, publisher.</p>
<b>SENSITIVE</b>	<p><b>Purpose:</b> Identify and flag attributes that require protection against unauthorized or unwarranted disclosure.</p> <p><b>Used for:</b> Data subject to legal, ethical, or privacy-related considerations, including but not limited to Personally Identifiable Information (PII), Quasi-Identifiable Information (QII) or proprietary information.</p>
<b>STANDARD</b>	<p><b>Purpose:</b> Provides canonical, normative semantics and mappings for a schema.</p> <p><b>Used for:</b></p> <ul style="list-style-type: none"> <li>• declaring the OCA standard version and namespace for the file,</li> <li>• mapping local attributes to canonical terms/URIs (interop with ontologies or other standards),</li> <li>• specifying recommended defaults (language, encoding, compatibility rules),</li> <li>• indicating which core overlays are required/assumed.</li> </ul>
<b>UNIT</b>	<p><b>Purpose:</b> Associates units of measure with numeric attributes.</p> <p><b>Used for:</b> height (cm), weight (kg), temperature (°C).</p>

Table 8.2 : Examples of “Community Overlays”

Overlay Name	Short Description	NextGen usage
ATTRIBUTE_MODIFIERS	<p><b>Purpose:</b> Provides modifications or additional details for attributes.</p> <p><b>Used for:</b> Optional/mandatory flags, read-only flags, etc.</p>	
CONDITIONAL	<p>Defines how attributes map to other schemas or external systems.</p> <p><b>Purpose:</b> Defines conditional logic between attributes.</p> <p><b>Used for:</b> “If X is true, Y must exist.”</p>	
CONSTRAINT	<p><b>Purpose:</b> Adds long-form documentation for the schema or attributes.</p> <p><b>Used for:</b> Developer notes, extensive descriptions.</p>	
DOCUMENTATION	<p><b>Purpose:</b> Adds long-form documentation for the schema or attributes.</p>	

	<b>Used for:</b> Developer notes, extensive descriptions.	
EXAMPLE	Provides examples for attributes or the entire schema.	
IDENTIFIER	<b>Purpose:</b> Defines identifiers for the schema. <b>Used for:</b> SAID references, version tracking.	
MAPPING	Defines how attributes map to other schemas or external systems.	
MAP	<b>Purpose:</b> Adds semantic meaning to key/value structures. <b>Used for:</b> Dictionaries, attribute maps.	
MEASURE	<b>Purpose:</b> Defines how numeric values should be interpreted. <b>Used for:</b> Precision, decimal places, ranges.	
VECTOR	<b>Purpose:</b> Specifies semantics for array-type attributes. <b>Used for:</b> Lists of texts, lists of numbers, nested arrays.	