EUROPEAN COMMISSION

DIRECTORATE-GENERAL FOR COMMUNICATIONS NETWORKS, CONTENT ANDTECHNOLOGY

Digital Excellence & Science Infrastructure

Technologies for Smart Communities

D05.02 Public report on the LDT Toolbox detailed specifications requirements

CNECT/2022/OP/0098 – Procurement of the Technical Specifications for the Twins (LDTs) Toolbox

This study has been prepared as part of the Local Digital Twins Toolbox project. The project is an initiative of the European Commission. Directorate-General for Communications Networks, Content and Technology of the European Union is responsible for contract management of the Local Digital Twins Toolbox project.

European Commission
Directorate-General for Communications Networks, Content and Technology
Unit C.3 Technologies for Smart Communities

Last update: 11 August 2023

Task	TASK-05: Outline design of the LDT Toolbox and draft procurement specifications
Deliverable	D05.02 Public report on the LDT Toolbox detailed specifications requirements
Authors:	Maria Garcia Barron; Cristiana Ramos; Francisca Carvalho; Nuno Regêncio; Ana Correia; Eva Birecki; Kamil Radomski; David Niepceron; Philippe Michiels; Ingrid Croket; Stefan Lefever; David Vermeir; Nico Spijkers; Walter Lohman; Christian Pasquale Aprile; Razgar Ebrahimy; Shahrzad Pour; Khurshed Ali; Gert Hilgers; Michael Mulquin; Carlo Montino
Reviewers:	Lambert Verhagen; Eline Lincklaen Arriens

Brussels: Directorate-General for Communications Networks, Content and Technology of the European Union, 2023

ISBN: 978-92-68-06527-3 doi: 10.2759/384198 KK-09-23-368-EN-N

Table of Contents

Table of Tables

Table of Figures

Abbreviations and Acronyms

List of Abbreviations and Acronyms	
AI	Artificial Intelligence
AR	Augmented Reality
API	Application Programming Interface
BB(s)	Building Block(s)
DEP	Digital Europe Programme
EC	European Commission
EDIC	European Digital Infrastructure Consortium
EU	European Union
GDPR	General Data Protection Regulation
HDD	Hard Disk Drives
ICT	Information and Communications Technology
IoT	Internet-of-Things
IP	Intellectual Property
LDES	Linked Data Event Streams
LDT	Local Digital Twin
MIMs	Minimal Interoperability Mechanisms
MR	Mixed Reality
SaaS	Software as a Service
SCC(s)	Smart Cities and Communities
VR	Virtual Reality
W3C	World Wide Web
XR	Extended Reality

1. Introduction

1.1 Context

The European Union (EU) has placed a strategic focus on **advancing the digital transformation of smart cities and communities**. This has materialised in a wide ecosystem of initiatives, not only linked to the digital transition, which includes common European data spaces, local and interoperable data platforms and Digital Twins, the Living-in.eu movement and the Digital Europe Programme (DEP), but also connected to the green transition, through initiatives supported under the European Green Deal, such as the New European Bauhaus.

In particular, the DEP targets EU cities and communities at various stages of digital maturity. For cities and communities that are less advanced, two upcoming DEP-funded projects on "Advancing initial stages for the transformation of smart communities"[1] will provide the tools and support needed to increase their awareness and readiness. For more advanced EU communities, the current project makes a proposal for the **design of a European Local Digital Twin (LDT) Toolbox** with advanced Building Blocks (BBs). The project has benefitted from a consultation phase with stakeholders to make the Toolbox "fit-for-purpose," by identifying the main needs and challenges when implementing Digital Twin solution(s) in cities and communities. As the Toolbox is intended to serve cities and communities across Europe, it should fulfil multiple features such as being **technology-agnostic, sustainable, and based on European standards and Minimal Interoperability Mechanisms (MIMs)**.

The concept of a **'Building Block'** (BB) provides **a level of abstraction that translates the desired capabilities into specifications for possible 'tools' that fulfil the needs of the LDT**, in a technology-agnostic and vendor-agnostic way. A BB delivers certain functionalities that correspond to different capabilities and that can be practically implemented by one or more tools. Each BB presented in this document will refer to capabilities, related functional requirements, and MIMs and other standards. The tools that can be used to fulfil the needs of the BBs, should comply with the specifications described in this document. A reference architecture is also proposed to explain how BBs interact with each other.

This **report identifies the prioritised BBs of an LDT Toolbox and the respective technical specifications**. Moreover, it puts forward a proposal for the **delivery solution of the LDT Toolbox**, namely how it will be made available to the EU communities, as well as the assumptions and criteria that should be considered for its future deployment and management on the functional side. The BBs technical specifications presented in this document form the basis for the development of the LDT Toolbox in an upcoming procurement opportunity[2].

1.2 Purpose

The purpose of this report is to identify the concept for the delivery solution of the EU LDT Toolbox and detail the technical specifications for the prioritised BBs, which will be the starting point of the development of the LDT Toolbox. More specifically, this report is organised as follows:

- **Chapter 2** – identifies and explains the concept behind the EU LDT Toolbox delivery solution and which business and functional requirements are applicable, as well as the prioritised BBs; and
- **Chapter 3** – presents the detailed technical specifications of each prioritised BB of the LDT Toolbox and lists some of the tools that can accelerate the implementation of BBs.

[1] See call for tenders https://etendering.ted.europa.eu/cft/cft-display.html?cftId=11876
[2] See Prior Information Notice for the call for tenders Services - 438605-2023 - TED Tenders Electronic Daily (europa.eu)

The report is complemented by two annexes providing more detailed information about the following topics:

- **Annex 6.1 –** presents a high-level description of non-prioritised BBs (Annex 6.1)
- **Annex 6.2 –** provides a detailed description of the identified capabilities resulting from the analysis of stakeholders' needs and requirements

1.3 Approach & Methodology

This report includes two main phases of activities that were fundamental to the completion phase of this report.

Phase 1: Summary of the EU Local Digital Twin Toolbox Design

This phase introduces the proposal for the delivery solution of the LDT Toolbox, namely the assumptions and criteria that were considered in the design, as well as the BBs that should integrate the Toolbox.

Phase 2: Building Blocks Technical Specifications

This phase describes in detail the technical specifications for each BB, including the following information (that will be further explained in section 3. Detailed Building Blocks Technical Specifications"):

- The minimum ambition level;
- The kind of BB;
- The maturity level;
- An internal code, to better identification;
- The relevant MIMs;
- Detailed description of what is each BB about;
- The capabilities covered by each BB, the MIM to be considered and the standards; and
- The requirements covered by each capability, as well as the mechanisms and the respective interoperability guidance.

Phase 3: Draft final report

This phase is the result of consolidating the outputs of the previous phases that lead to the final report "D05.02 Procurement of the Technical Specifications for the Local Digital Twins (LDTs) Toolbox."

2. Conceptualisation of the EU Local Digital Twin Toolbox Design

The transformation journey of EU cities and communities on adopting an LDT is a gradual and cumulative process, which follows different maturity stages. In that sense, an EU LDT Toolbox should be designed to serve cities and communities across Europe and to support the development and expansion of their LDTs. This Toolbox includes a set of BBs, where each BB can be recognised as an independent accelerator for the implementation and operationalisation of LDTs.

Initially, the LDT Toolbox will focus on more advanced cities and communities. Over time, The LDT Toolbox is expected to quickly evolve through the identification of tools that have already effectively been used within those LDTs. Those tools can then be incorporated in the LDT Toolbox and will add value to the initial community members, not only by further developing the tools they have used themselves, but also by better understanding their needs and matching them with a selection of new tools to adopt.

2.1 Overview of the Delivery solution

Concept of the EU LDT Toolbox Delivery Solution (The Container)

The delivery solution of the EU LDT Toolbox, to be developed, represents the container of the different BBs of the Toolbox, following a **platform business model** that allows BBs to be offered to EU cities and communities. This container aims to include and deliver all LDT Toolbox BBs. The BBs contained in the LDT Toolbox act as open and reusable digital solutions, which can take the shape of a guideline (e.g., framework, standard) , a tool (e.g., software, Software as a Service (SaaS)) or any combination thereof.

Following a "one-stop-shop" concept, the deployment of a delivery platform would allow stakeholders to **access common BBs**. The platform should bring together the resources and capabilities required to, on the one hand, deliver BBs, and on the other, to become a true driver of innovation, transformation, and scalability of LDT development.

The proposal is for the delivery solution to be a **web-based and open-source platform hosting BB** solutions (for instance code repositories, accompanied by guidelines and usage instructions). This would enable EU cities and communities to access the EU LDT Toolbox, enabling them to work together on BBs, track changes over time, propose and review code modifications through pull requests, and contribute to a vast community of open-source BBs.

Further specifications on an EU LDT Toolbox Delivery Solution should be designed by the integration partner during the deployment phase, after BBs have been developed in the scope of the upcoming procurement targeting the "Development of the LDT toolbox for advancing the transformation of Smart Communities"[3]. Nevertheless, according to best practices applying to web-based and open-source code repository platforms, a set of **integration, security, and operations guidelines** should be considered for deployment. The aforementioned guidelines are presented in more detail below.

Integration Guidelines

The deployment of an EU LDT Toolbox Delivery Solution should consider the following integration guidelines:

- Loosely coupled & event-based architecture, standardising data formats and avoiding point-2-point integrations.

[3] See https://ted.europa.eu/udl?uri=TED:NOTICE:438605-2023:TEXT:EN:HTML

- Well-structured, well-documented and published Application Programming Interface (API) to improve visibility and re-use.
- Use of APIs to provide BBs and/or services that abstract the complexity of systems, business rules & data flows to improve re-use.
- Design and implement both for failure (by focusing on best-case and alternative paths), and for support (by building in monitoring, logging, and alerting capabilities from the start).

Security Guidelines

The deployment of an EU LDT Toolbox Delivery Solution should consider the following security guidelines:

- Establish context before designing a system. Determine all the elements which compose the Toolbox delivery solution, so defensive measures do not have any blind spots.
- Design a resilient system to denial-of-service attacks and usage spikes. Regarding high-value or critical BBs, it is essential to ensure that technology is always available.
- Design a system that can make penetration difficult and spot suspicious activity as it happens to take necessary action.
- Design to naturally minimise the severity of any compromise.
- Ensure data-centric protection with Zero Trust – never trust, always verify.

Operations Guidelines

The deployment of the EU LDT Toolbox Delivery Solution should consider the following operational guidelines:

- Maintain consistency of the delivery solution performance, functional, and physical attributes with its requirements, design, and operational information throughout its life.
- Guarantee the near real-time monitoring, control, reporting and auditing of the delivery solution.
- Ensure a rapid response to any reported issue through an effective delivery solution performance monitoring to ensure operational efficiency.
- Integrate BB management practices to respond to any notified occurrence, ensuring the quality and efficiency of the provided BBs against the defined targets.

2.2 Guidance for LDT Toolbox deployment

Choosing and specifying a platform on which an LDT Toolbox application will be made accessible and operational is a critical step in the development and deployment of the LDT Toolbox Container. In that sense, a well-defined process supported by clear guidelines and industry standards should be considered to ensure the container platform meets the EU stakeholders needs and enables an efficient and scalable tools deployment.

This guide outlines a seven-phased process to specify the LDT Toolbox Container platform.

Step 1 | Define the Vision for the LDT Toolbox Platform

The main aim of this first step is to determine the approach for the concept of the web application based LDT Toolbox platform, following the activities below. At this stage, at least two alternative options should be considered:

- Leveraging an existing EU container/ platform, such as Net Zero Cities Portal, DS4SSCC Inventory and eIDAS; and
- Creating a new and dedicated container/ platform for the LDT Toolbox BBs.

Purpose Identification: Clearly define the purpose and goals of the platform. Consider scalability, portability, and operational efficiency.

Stakeholder Engagement: Engage stakeholders through workshops, surveys, and interviews to gather diverse perspectives and expectations.

Vision Alignment: Create a concise vision statement that emphasises scalability, developer empowerment, and seamless application deployment.

Step 2 | Defining Key Actors

The goal of this step is to identify the key actors who will interact with and benefit from the LDT Toolbox platform, considering the activities presented below.

Stakeholders Identification: Identify the key actors that will interact directly or indirectly with the platform – considering both actors that will be developing and/or maintaining the platform, as well as those who will consume its BBs.

User Role Definition: Clearly define roles such as developer, operator, and administrator, outlining their responsibilities and permissions.

Responsibilities Specification: Conduct role-mapping sessions to define the responsibilities and tasks of each key actor, clearly outlining their interactions with the LDT Toolbox platform.

Role-based Training: Introduce a role-based training program to empower actors with the skills they need.

Step 3 | Use Cases Identification

The following step aims to define a set of use cases aligned with the LDT Toolbox goals. These use cases provide specific scenarios in which the platform will be utilised.

Use Cases Identification: Work with stakeholders to brainstorm and identify common use cases and potential edge scenarios where an LDT Toolbox platform will deliver value. These could include application deployment, scaling, orchestration, and management but also sharing knowledge items, guidelines and encouraging collaboration, as being a community-led solution.

Use Cases Prioritisation: Rank the identified use cases based on their importance and alignment with the container vision. Focus on use cases that provide the most value and impact. Form a cross-functional use case prioritisation team to ensure balanced decision-making.

Step 4 | Use Cases Documentation

The focus of this step is to detail how each prioritised use case will be addressed using the chosen LDT Toolbox Container platform. It will entail creating detailed description for each use case including context goals and expected outcomes.

Use Cases Description: Develop detailed use case narratives that encompass user goals, interactions, and expected outcomes.

Use Cases Steps and Actions: Define the exact steps actors should take, including commands, API calls, and interactions with the user interface.

Use Cases Decision Points: Identify pivotal decision points within each use case, guiding users toward optimal choices.

Step 5 | Use Cases Technical Specification

The focus of this step is to establish an architectural representation of the application, detailing its main specifications and design patterns (user interface, access management, modules, integration with other tools, etc.)

Feature Mapping: Translate use case requirements into technical features, considering aspects like container orchestration, networking, and security.

Technical Architectural Design: Create architectural diagrams that illustrate how the container platform components will be integrated with existing infrastructure, including networking, storage, load balancing, and security.

Technical Requirements Specifications: Detail the technical specifications for each feature, including data exchange formats, communication protocols (e.g., HTTP, gRPC), integration points, and performance benchmarks.

Scalability and Performance: Define how the platform will handle scaling, load balancing, and resource allocation. Specify performance expectations and metrics.

Step 6 | Validation and Documentation

The aim of this step is to validate and document the technical specifications for an LDT Toolbox Container.

Review and Feedback: Share the specified technical specifications with relevant stakeholders for validation and feedback. Incorporate any necessary revisions or improvements based on their inputs.

Iterate as Needed: Iterate on the specifications based on feedback and evolving requirements. Ensure that the technical specifications remain aligned with the overall LDT Toolbox Container vision and use cases.

Specifications Consolidation: Compile all the specifications, use case details, and architectural diagrams into comprehensive documentation.

Communication Plan: Develop a plan for communicating the finalised technical specifications to development teams, architects, and other relevant parties.

Step 7 | Planning and Preparation

The aim of this step is to ensure the proper preparation of all required foundations for the deployment of the several prioritised use cases.

Infrastructure Setup: Prepare the underlying infrastructure where the container platform will be deployed. This might involve provisioning virtual machines, setting up networking, and configuring storage.

Container Platform Installation: Follow the platform's official documentation to install and configure the container platform components. This includes setting up master nodes, worker nodes, and the container orchestration layer.

Networking and Security: Configure networking policies, load balancers, firewalls, and security settings to ensure proper communication and isolation between containers.

2.3 Functional and Business Requirements

A set of functional and business requirements must be considered for the development of an EU LDT Toolbox. To build, maintain and deploy an EU LDT Toolbox, the following criteria function as main drivers:

1. Design driven by Local Communities' requirements and by EU priorities and values.
2. Local Communities, Industry, and Academia co-create tools.
3. Generated Intellectual Property (IP) is managed in a way to ensure core tools can continue to be available on an Open-Source Basis.
4. Involvement of an active community in an EU LDT Toolbox steering, maintaining, and developing it.
5. Effective governance processes are put in place.
6. Long-term Self-sustainability of the EU LDT Toolbox.
7. MIMs Plus compliant and Standards- based EU LDT Toolbox.

A deep dive on each driver is provided below.

1 | Design driven by the Requirements of Communities and by European Principles and Values

LDTs can be essential to achieving the twin (green and digital) transition in the EU, in line with the goals of the European Green Deal, and the European Commission's (EC) digital priorities. This will ensure that European principles and values are met. Moreover, it is important to take into consideration the six principles of the Living-in.EU declaration to accelerate the sustainable digital transformation in the EU, namely:

- a citizen-centric approach;
- a city-led approach at EU level;
- the city as a citizen-driven and open innovation ecosystem;
- ethical and socially responsible access, use, sharing and management of data;
- technologies as key enablers;
- interoperable digital platforms based on open standards and technical specifications; and
- APIs and shared data models.

These principles underline the need for the tools to be developed in response to the requirements of cities and communities. The LDT tools provided in the Toolbox need to be those that will enable the data collected to be used in ethical and socially responsible ways and that will support the needs of their citizens.

2 | Co-creation with Communities, Industry, Academia, and Other Relevant Stakeholders

EU communities are consistently looking for innovative technologies that can enable them to make decisions on accurate, comprehensive, and up-to-date information, and that is what LDTs can provide. However, to always be up-to-date and deliver the most value to citizens, a big effort is needed. This includes the involvement of industry and academia to identify appropriate technology options.

To ensure its future fitness, the LDT Toolbox needs to be co-created with the key stakeholders.

3 | Management of Intellectual Property

Following the development and deployment of the LDT Toolbox that will generate valuable IP, such as software, technologies and authoritative documentation, the need to manage IP arises. Considering that these will support the legal, operational, and other aspects during the implementation, it is crucial to safeguard these assets.

To ensure the correct management of IP, well-constructed legal agreements, and effective management processes need to be in place.

4 | Facilitating an Active Community

The LDT Toolbox it is designed to use Digital Twin technologies to help with the better management of EU cities and communities for the benefit of its citizens. Based on that, it should be tailored to seamlessly integrate with a broad range of legacy systems, infrastructures, resources, and expertise found across the EU communities.

To effectively develop and deploy the LDT Toolbox, it is important to establish a sizable, well-organised and engaged community that will collaborate in defining, developing, and evaluating the tools, namely by making use of the wide variety of expertise and backgrounds found throughout Europe (and beyond).

It is envisaged that this community will support the small- and medium-sized enterprises (SMEs) and/or volunteer experts from individual local communities, on developing and testing tools that are specifically relevant to a local community, to meet their needs.

5 | Governance

To ensure a proper management of the LDT Toolbox, a multi-layer governance process must be implemented to effectively manage the business, model, and data governance.

Business Governance: identifies the key entities engaged in the LDT Toolbox; defines input-output and inter-related LDTs collaboration rules; agrees on the level of decision-making and autonomy of the LDT; and enables data sharing mechanism rules for decision-making[4].

Model Governance: specifies the standards and agreements governing an LDT Toolbox to ensure accountability and compliance to the overall business rules (defined/to be defined by the EC). The model governance must be compliant to the guidelines regarding transparency, explainability, user engagement, ethics, sustainability, and societal goals[5].

Data Governance: defines a complete set of guidelines and standards, which should cover the overall management and operations of the data, such as rules, policies, procedures, and processes. Data must be treated according to the Data Act, Artificial Intelligence (AI) Act, General Data Protection Regulation (GDPR)-compliance, and FAIR principles. The data governance plan may follow the GAIA-X European initiative for data sharing and should address the following[6] aspects:

- Define data inputs, processing, and outputs;
- Agreements regarding data sharing, privacy, and security;
- Data ownership regarding raw and metadata;
- FAIR treatment of the data following privacy, security, and applicability guidelines; and
- Continuous risk monitoring of the data.

6 | Self-Sustainability

Although it is foreseen funding for the 30-month startup phase, it is vital that there is a clear path to ensure the LDT Toolbox is self-sustainable in the future. For that purpose, sustainability can be seen through the lens of the components presented in Table 1.

[4] Kalaboukas, Kostas, Dimitris Kiritsis, and George Arampatzis. "Governance framework for autonomous and cognitive Digital Twins in agile supply chains." *Computers in Industry* 146 (2023): 103857.
[5] Ibid.
[6] Ibid.

Table 1: Self-Sustainability Components

Components	Considerations
Sustainable governance	The governance structure needs to remain stable, so it is important to be aware of the LDT Toolbox and its assets could not be taken over by a commercial entity. The possible governance breaks down by one entity taking it over, poorly drafted procedures or hampered decision making.
Financial sustainability	The sustainability business models that may fit the toolbox are: 1) **Community model –** A community of users and industry partners that actively collaborate on the ongoing development and maintenance of the software, being the burden distributed among the members of this collaborative community. 2) **Subscription model –** The users subscribe to gain continuous access to centralised maintenance and support services, ensuring a steady stream of support and software updates for the users. 3) **Commercial model –** The software is offered in a commercialised version, typically as part of other software licenses, allowing users to receive dedicated support and maintenance services in exchange for a fee. 4) **Central support model –** A centralised entity that takes on the responsibility of managing software releases and plays a crucial role in providing support and maintenance services, ensuring the software's stability and availability.
Technical sustainability	An LDT Toolbox should provide support on how to develop, create and enhance the technical tools, as well as its maintenance (functionality, stability, and reliability) over time.
Sustainable community	The basis for this component should be a community-based, collaborative environment that will ensure that there are good processes in place to keep all stakeholders engaged.
Sustainable scalability	To the meet the EU communities needs, an LDT Toolbox should allow for the scale of the set of tools to fully meet the aspirations for how LDTs can help enhance the lives of citizens.

7 | Standards-based and MIMs Plus Compliant

Standards and MIMs Plus enable the interoperability of data, systems, and services between cities around the world, as well as faster and easier development of tools, as they include tested components.

2.4 EU Local Digital Twin Toolbox Building Blocks

The EU LDT Toolbox is expected to comprise a set of BBs. **Each BB represents an open and re-usable accelerator** and can take the shape of a framework, a standard, a software, or a SaaS, or any combination thereof. The main aim of a BB is to support the advanced EU communities in building and implementing an LDT and/or taking the existing LDT to more mature levels.

In that sense, a six-phased approach, was followed to identify, analyse, and prioritise relevant BBs to be included in the EU LDT Toolbox, considering the results of the inputs provided by key stakeholders during a consultation process:

- **Phase 1 | Execution of consultation process**, including interviews, surveys, and workshops, focused on the identification of key stakeholders' needs and requirements for the future LDT Toolbox.
- **Phase 2 | Determination of LDT ambition levels** (i.e., awareness, experimental, predictive, intelligent).
- **Phase 3 | Identification of strategic, technical, and legal capabilities** (see Annex 6.2), which were mapped against different ambition levels to understand fundamental features for a LDT to work).
- **Phase 4 | Determination of BBs** to underpin the infrastructure that enables the LDT capabilities to be realised and utilised effectively.
- **Phase 5 | Execution of prioritisation exercise** on the total list of BBs to identify the final list of BBs for which detailed technical specifications are to be provided.
- **Phase 6 | Definition of technical specifications for BBs**, which is the basis of Section 3 of this report.

As a result of phases 1 to 5, the project team has identified **18 priority BBs** for the Toolbox, as shown in the table below. This table indicates for each BB the respective BB Category, the capabilities that the BB supports, and the associated ambition level.

The LDT Toolbox focuses on advanced maturity levels, to support the digital transformation journey towards AI enabled LDTs. The selection of the prioritised BBs considers the existing ecosystem wherein the LDTs operate and focuses on the BBs that deliver the LDT core services, related to the higher ambition levels (Level 3 and 4). In the LDT reference architecture, also non-prioritised BBs are referred to but not described in detail as they are either only specific for lower ambition levels or they are already described as re-usable BB in view of other EU initiatives (e.g., Identity and Access and User management are defined in view of the DS4SSCC Inventory). For the prioritised BBs, it is important to note that certain BBs are already required as from lower ambition levels but in that case, it is important that they are already designed to fit the future requirements for higher ambition levels. For instance, when defining the needs for data storage, it is required that the implementation of this capability from the early stage of the awareness stage, is already supporting the concepts for federated data storage and the integration with data spaces. Because of the importance of the concept of data storage needs for a LDT setup, this BB was added to the list of prioritised BBs.

Table 2: Final List of Prioritised Building Blocks

#	Building Block (BB)	BB Category	Capability	Ambition Level
BB.01	**Reference Architecture**	Strategy	Public procurement LDT Roadmap Management	1 – Awareness of Twins
BB.02	**Case & Scenario Manager**	Integration components	Usage Context Information Provisioning	4 – Intelligent Twins
			Case, Scenario & Experiments Management	2 – Experimental Twins
BB.03	**Collaboration & Community Management System**	Visualisation and UX	Citizen Engagement and Case & Scenario Feedback Gathering Collaboration and Community Management	2 – Experimental Twins
BB.04	**Data Query Service**	Data Services	Data Publishing	2 – Experimental Twins
BB.05	**Data Replication Client**	Data Services	Data Replication	2 – Experimental Twins

BB.06	**Data Workflow and Component Orchestration**	Integration components	Data Processing Data Flows and Component Orchestration	2 – Experimental Twins
			Data Transformation	3 – Predictive Twins
			Technical Transparency & Explainability	4 – Intelligent Twins
BB.07	**Extended Reality (AR/VR Services, etc.)**	Visualisation and UX	Augmented Reality (AR) Virtual Reality (VR)	4 – Intelligent Twins
			Advanced Visualisation & Geo dashboarding	2 – Experimental Twins
BB.08	**Federated Learning Service**	Algorithms & Models	Federated Learning & Training	3 – Predictive Twins
BB.09	**Integrated Environment**	Visualisation and UX	Integrated User Experience	2 – Experimental Twins
BB.10	**Interaction Service**	Integration components	Interaction Support	2 – Experimental Twins
BB.11	**Message Broker**	Integration components	(Near) Real-time Data Processing Data, Publishing and Subscribing Ingest Data Streaming Data Urban Measuring, Sensing & Control Alerts and Notification	3 – Predictive Twins
			Data Flows and Component Orchestration Interaction Support	2 – Experimental Twins
			Supervised or Unsupervised Actuation, Command & Control	4 – Intelligent Twins
BB.12	**Model Abstraction SDK/Service**	Algorithms & Models	Model Abstraction Model Hosting	2 – Experimental Twins
BB.13	**Model Catalogue**	Algorithms & Models	Algorithm & Model Management	2 – Experimental Twins
			Accountability Information Provisioning	4 – Intelligent Twins
BB.14	**Model Usage Guidelines**	Governance & Conformance	Model Governance & Compliance	2 – Experimental Twins
BB.15	**Algorithms & Models**	Algorithms & Models	Prediction Machine Learning & AI	3 – Predictive Twins
			Simulation	2 – Experimental Twins
BB.16	**Asset Registry**	Data Services	(Meta) Data Schema Management Data Source Management Ontology Management	1 – Awareness of Twins
			Semantic Governance & Compliance	2 – Experimental Twins
			Procedural Transparency Technical Transparency & Explainability	4 – Intelligent Twins
BB.17	**Synthetic Data Generation Tools**	Data components	Semantic Governance & Compliance	2 – Experimental Twins
			Procedural Transparency Technical Transparency & Explainability	4 – Intelligent Twins

BB.18	**Data Storage: Data Lake, Data Warehouse, Data Lakehouse**	Data Components	Data Processing, Data Replication, Data Storage, Data Time Travel	2 – Experimental Twins
			Data Transformation	3 – Predictive Twin
			Technical Transparency & Explainability	4 – Intelligent Twins

The technical specifications of each BB are described in Section 3 "Detailed Building Blocks Technical Specifications" of this report.

3. Detailed Building Blocks Technical Specifications

To support cities and communities in the development of LDTs, the Toolbox will suggest tools that can be used to achieve the necessary capabilities for LDTs, according to the envisioned ambition level (the four ambition levels are explained in Table 3).

The notion of 'Building Blocks' is introduced to create a level of abstraction that translates the desired capabilities into specifications for possible 'tools' that fulfil the needs of the LDT, in a technology-agnostic and vendor-agnostic way. A BB delivers certain functionalities that correspond to different capabilities and that can be implemented by one or more tools. The BBs are described in Section 3.2 with reference to capabilities, related functional requirements, and reference to MIMs and other standards. The tools that can be used to fulfil the needs of the BBs and should comply with the mentioned specifications that are documented in the detailed specifications. How the BBs interact with each other is explained in the reference architecture, which is also defined as BB.01. The reference architecture includes BBs that were not prioritised for the LDT Toolbox project and are therefore not documented in Section 3.2. Instead, the high-level descriptions for the BBs are documented in Annex 6.1.

Regarding detailed specifications, such as the examples and possible tooling, the document presents a snapshot based upon the current experiences and does not include a full analysis of all possible tools. The information in this document should serve as a starting point. Thorough research and evaluation should be conducted to ensure that the proposed tools align with the specific objectives and requirements of each project.

3.1 Description Template for Technical Specifications

The template used for the technical specifications is inspired by the MIMs Plus template[7] delivered in the context of the work of the Living-in.EU Technical Subgroup. The technical specifications aim to document the link between capabilities and functional requirements in a structured manner. The template structure is identical for all BBs, except for the reference architecture that elaborated on the description and a visual scheme. Commonly, the template includes four main items:

- Summary Table
- Description
- Capabilities, Functional Requirements, Mechanisms, and Interoperability Guidance
- Examples

The type of information and level of detail can vary between BBs based on the maturity level of the existing tools and practices.

Summary table

This summary table addresses metadata for the BB – minimum ambition level, kind, maturity level, internal code and relevant the MIM.

Table 3: Summary Table of the Template

Minimum Ambition Level	**1 – Awareness Twins** At this level, explorative visualisation using layers, sensors, and sensor data plays a crucial role. The focus is on gaining situational awareness in various

[7] Approved MIMs Plus_LI.EU_v6.0.docx (living-in.eu)

	domains such as traffic, logistics, environment, demographics, and health. Although limited data management capabilities are required, this level serves as the initial step in the development of a Digital Twin roadmap. **2 – Experimental Twins** This level involves experimental visualisation and the focus shifts towards what-if simulations, which enable the exploration of different scenarios and their potential outcomes. These simulations require a more structured approach to data and a scenario management system to effectively manage the complexity of the analyses. **3 – Predictive Twins** Predictive analysis is a technique that uses various data sources, including near-real-time data flows and validated models such as AI, to anticipate and forecast future events. By analysing historical data and patterns, predictive analysis enables us to make informed predictions about potential events and their impact on specific areas or domains. **4 – Intelligent Twins** The highest level, the intelligent twin, refers to a concept which involves the autonomous operation of twins to automate city infrastructure, guide traffic and people, and anticipate critical events. This concept represents a significant advancement in the field of smart cities and requires a high degree of maturity and expertise.
Kind	**Software |** Requires the implementation of tools to ensure the necessary capabilities of the BB. **Guidelines |** Requires the definition of guidelines, processes, input from best practices to implement the BB. **Software/Guidelines |** Requires both the implementation of software tools and elaboration of guidelines to implement the BB.
Maturity Level	In line with the terminology of the MIM template, the maturity level types are used to indicate the level of maturity of existing technical solutions that comply with defined standards for the defined requirements that support interoperability in an LDT environment and integration with other BBs. **Plug & Play |** standards exist and facilitate interoperability and integration. **Good Enough |** existing standards are applicable; however, there will be a need for extra effort to support integration with other BBs. **No Existing Standard |** needed standards and policies for this BB and associated tools are not defined yet.
Internal Code	**BB.XX** = Represents a prefix to identify the BBs in the scope of the project.
Relevant MIM	Summary of relevant MIMs to indicate which topics of interoperability are of main interest for this BB.

Description

The description adds descriptive text to explain the purpose and way of working of the BB. The description points to the objective(s), and the dependencies with other BBs are also described.

Capabilities, functional requirements, mechanisms, and interoperability guidance

This section focuses on the identification of the capabilities and related ambition level that is covered by the BB and details the associated requirements. A first table (represented in Table 4) lists the capabilities, the capability categories with associated ambition levels, MIMs to consider for these capabilities, and an overview of existing standards.

Table 4: Capabilities Template Table

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards

In a second table (represented in Table 5), the functional requirements for the BB are listed, with a reference to the related capability and links to mechanisms[8] and interoperability guidance[9]. Mechanisms and interoperability guidance can be seen as a suggestion of current Best Practices; it is therefore not a fully complete list. The Mechanism section will mention what software specs/ standards/ components are used by the tool to address those requirements whereas the interoperability guidance refers to more specific examples of de facto standards, published specifications from standardisation bodies, among others.

Table 5: Mechanism Template Table

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1.				
2.				
3.				

Examples

The examples of tools referenced in this document have been selected based on the experience accrued from prior projects, with a preference for open-source tools. If relevant, commercial tools are also listed as they can be used as inspirational examples for further analyses of the necessary tools for the LDT Toolbox. Please note that this listing does not constitute a comprehensive, formal market analysis encompassing all potential tooling options. As the technology landscape is dynamic and constantly evolving, continuous monitoring is necessary to keep abreast of industry trends and emerging tools. The tools must also be individually assessed for evaluating the overall requirements in line with the system's needs. The level of detailed information related to the tools therefore varies,

[8] As defined in section 3.5.4 of Approved MIMs Plus_LI.EU_v6.0.docx (living-in.eu)
[9] As defined in section 3.5.5 of Approved MIMs Plus_LI.EU_v6.0.docx (living-in.eu)

i.e., the provided information can differ from one BB to another. In general, tools have been listed in a table with minimum information on:

- Name of the tool
- Open source: Yes / No
- Popularity & community size
- Standard adherence for the tool

With regards to the features of a tool, a detailed analysis per feature is not executed, so this information cannot be supplied for all tools.

When no tools are known or defined for BB, examples are also given that deliver descriptions and assumptions for the services related to the functionalities for possible tooling.

3.2 Technical Specifications for prioritised Building Blocks

This sub-section presents the technical specifications for each of the 18 BBs identified in this project, following the template described previously.

3.2.1 Building Block 01 | Local Digital Twin Reference architecture

Table 6: Summary Table of Building Block 01

Minimum Ambition Level	1 – Awareness Twins
Kind	Guidelines
Maturity Level	N/A
Internal code	BB.01
Relevant MIM	All

The description of an LDT reference architecture addresses different topics to explain how to interpret and use the reference architecture:

- Scope of the reference architecture: the concepts and purpose of a reference architecture are explained.
- An LDT as a System of Systems approach: in this section the specific System of Systems approach for an LDT reference architecture is described.
- Visual scheme and description: a visualisation of the reference architecture and the different BBs is presented as well as the interactions between the BBs.
- Use case narrative: based upon a fictive use case a narrative describes how different BBs from the reference architecture can work together.
- Evolving from a reference architecture towards a detailed system design.
- Examples reference architecture and system architecture: in this section, table 7 refers to inspirational examples of reference architectures for Digital Twins and related concepts such as data spaces and other EU initiatives. In addition, examples are given of Digital Twin platforms of which the system architecture is in line with the presented LDT reference architecture.

Scope of the reference architecture

A reference architecture aims to create a joint vision around a certain topic using a joint language and structure to initiate system architecture design. It contributes to a joint understanding of the concepts and components that compose the system. Moreover, it allows to better identify and describe the needed fundamentals to reach the necessary levels of interoperability between the different architectural components, between architectures, and increase the software portability between platforms that contribute to the realisation of the architecture.

The reference architecture offers a framework to make agreements for standardisation of the different technical elements throughout the lifecycles of the systems. It enables and fosters the dialogue with and between the different stakeholders contributing to the realisation of LDTs. For that reason, a reference architecture introduces the following aspects:

- **Terms and concepts** for capabilities. The capabilities are inspired by the input from the surveys and were elaborated further based upon the Digital Twin Capabilities Periodic Table from the Digital Twin Consortium[1]. The capabilities cover different domains from a functional perspective, as well as a strategic, governance and compliance perspective. The functional capabilities are detailed into different areas, covering requirements related to Data, Analytics, Visualisation, Integration, Internet of Things (IoT) and Security.
- Alignment with **best practices and existing agreements,** such as the OASC MIMs or the selection of differentiating components that are not addressed by other activities.
- Pointers to standards that contribute to the **interoperability**, such as mentioned in the detailed descriptions of the component requirements.
- List of important **components** that are typically used within the reference architecture, such as the BBs mentioned further in this document.
- **Conceptual description** on how the system as an integrated implementation of the components should behave, such as the system of systems description in this BB.
- **Visual representation** of the technical BBs for an LDT and their function within the reference system, such as the visuals mentioned in this BB.

Therefore, the information that supports the reference architecture is not only present in the description of this BB, but spread throughout this document (e.g., listing the capabilities, the BBs, and the references to standards and interoperability guidelines within the BB descriptions, etc.). In this description, the focus is put on the conceptual description and a visual representation of the reference architecture.

A Local Digital Twin as a System of Systems approach

LDTs are a virtual representation of the physical assets, processes, and people within a geographically located community, which reflect and derive from cross-sectorial, historical and (near) real-time data. Their purpose is to **enhance evidence-based decision-making** at the operational, strategic, and tactical levels to better meet the needs of communities. **LDTs combine multiple technologies**, such as data analytics and AI, enabling predictive and simulation models that can be updated and changed as their physical equivalents change.

The above paragraph highlights keywords that indicate that an LDT is a **System of Systems**. The master system is a software materialisation itself, as it is a software representation that can be interacted with. That software materialisation can be called the "**LDT System**" and refer to the systems it interacts with as subsystems because these can have their own purpose without needing a Digital Twin. This LDT System needs to use data and services, that it derives from existing sources. These data sources and services can exist outside of the Digital Twin materialisation but need to have a defined interaction. These sources can be Open Urban Data Platforms, IoT, cloud platforms, Open Data APIs, static data sets, and data spaces, among others. Although a LDT needs to interact with these

subsystems, their technical internal requirements are being defined out of the scope of the LDT toolbox. The same holds for combining data analytics and AI tools.

Moreover, an LDT **addresses cross-domain problems** and provides suggestions that can create impactful changes in the community, under continuous change of data and environment. **An LDT System, needs to be able to clearly manage the change and efficient use of diverse subsystems to be resilient and reliable**. The system will need to be able to orchestrate the subsystem interactions and processes and maintain (or require) interoperability with its subsystems. It is here that different important European standardisation activities come together, such as data spaces and LDTs and common interoperability mechanisms like MIMs are the glue to assure that the interactions are compliant with these standards as well. In order to facilitate the realisation of LDTs, important considerations include architecture design of the system, and utilisation of underlying technologies and services. The **purpose of the reference architecture is to document technology-independent reference architecture with its underlying BBs** and define generic components that can be used to classify related technologies and products/services with respect to the reference architecture. The reference architecture relates to the needed functionalities that are required in view of realising system functionalities.

There are (under construction) lots of specifications and standards out there on the subsystems, such as Visualisation Platforms, Open Urban Platforms (e.g., DIN) and data spaces (e.g., IDSA, Gaia-X, SIMPL). The LDT System must take these into account to **leverage its representing, enhancing, combining, enabling and change management** features. For example, common European data spaces will ensure that more data becomes available for use in the economy and society, while keeping the companies and individuals who generate the data in control.[10] An LDT Toolbox for cities and communities will contain solutions and guidelines for the technical BBs of this reference architecture, as well as guidelines to manage and operate the LDT implementation within the System of Systems context. As this is an evolving ecosystem, this also means that new components may be needed in future. An example is the SIMPL[11] middleware project that will be implemented. This **System of Systems view** is highlighted in Figure 1.

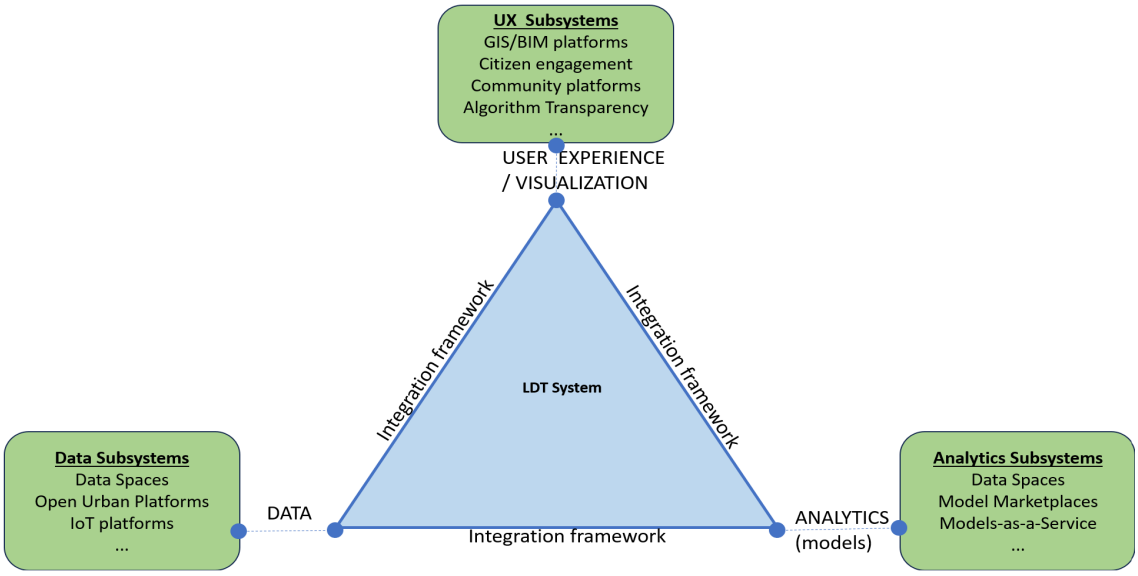


Figure 1: System of Systems view of an LDT

Source: Consortium

[10] A European Strategy for data | Shaping Europe's digital future (europa.eu)
[11] Simpl: cloud-to-edge federations and data spaces made simple | Shaping Europe's digital future (europa.eu)

An **LDT System will mainly interact with 3 types of subsystems: data, analytics and user experience**. Of course, these systems can also have mutual dependencies. The successful implementation of an LDT system will depend on the evolution of the standardisation of these subsystems, and the capability of the LDT core system itself to adapt seamlessly to these evolutions. That is why **interoperability is a key design concern for any system implementation**.

The objective of an LDT Toolbox is to achieve horizontal interoperability so that LDTs can interoperate on a horizontal scaling and vertical interoperability within the broader scope of digital solutions that are developed in the overall EU ecosystem as for instance the DS4SSCC and data spaces initiatives as depicted in Figure 2.

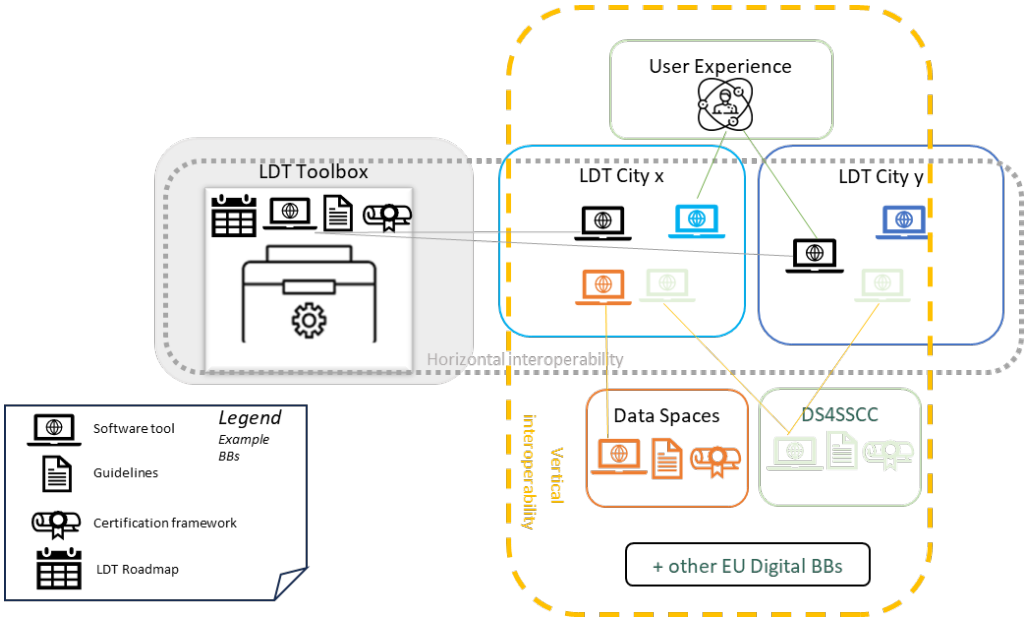


Figure 2: Horizontal and vertical interoperability for LDTs

Horizontal scalability means that a BB from an LDT Toolbox like a case & scenario manager and the associated assets like case and scenario descriptions can be used in a uniform way by City x and City y to support their specific use cases. The vertical interoperability means that this LDT BB can interoperate with a BB like Identity Management from the DS4SSCC inventory: the city can use this Identity Manager BB also for other applications like eGovernment, but it can also be used for the LDT and interoperate with the case & scenario manager.

Visual scheme and description

The summary of BBs used in the reference architecture, their importance within the categories of capabilities and their state of description within this document is shown in Figure 3.

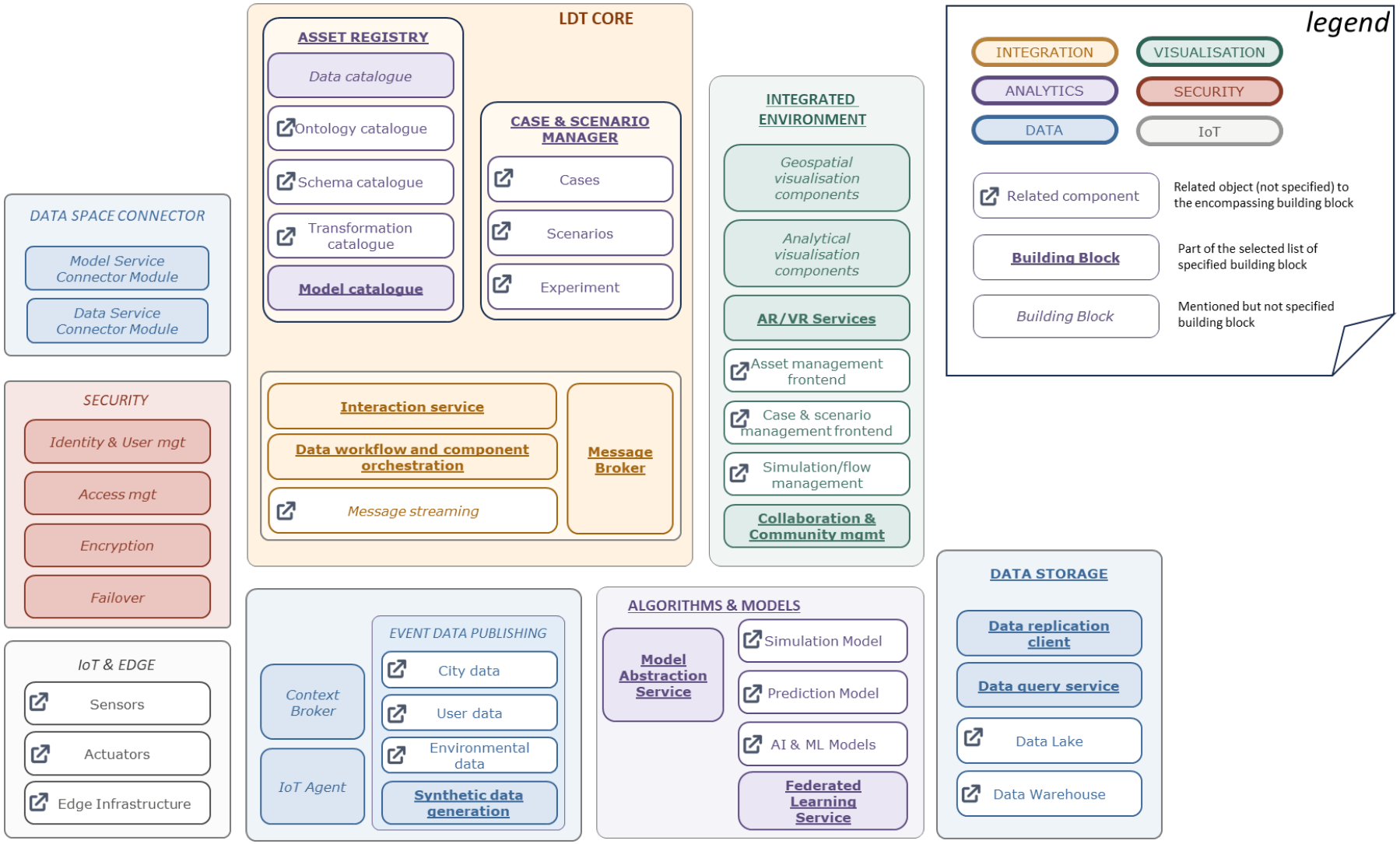


Figure 3: LDT Reference Architecture (Building Blocks)

Source: Consortium

As specific BBs from this reference architecture are also needed from or related to other EU digital projects or initiatives, the development of an LDT Toolbox with 18 BBs that are described in more detail with their functional requirements in Section 3.2 of this document have been prioritised. The descriptions of the non-prioritised BBs are provided in Annex 6.1. This reference architecture lists the selected BBs and the non-prioritised BBs and mentions related components that are not specifically defined as a separate BB. Note that these components are commonly used for LDT implementations and are either outside an LDT system (e.g., sensors, actuators) or can be specific developed applications that are used on top of a BB (e.g., asset management frontend is an end-user application developed on top of Asset Registry to provide user a readable overview of all assets that are managed within the LDT).

Figure 4 gives a more detailed vision on the position of the BBs and their dependencies in the total picture. In the detailed description of each BB, the interdependencies with other BBs are also described. The aforementioned System of Systems triangle is used to illustrate the position of these BBs as part of the System, but in relation to the subsystems.

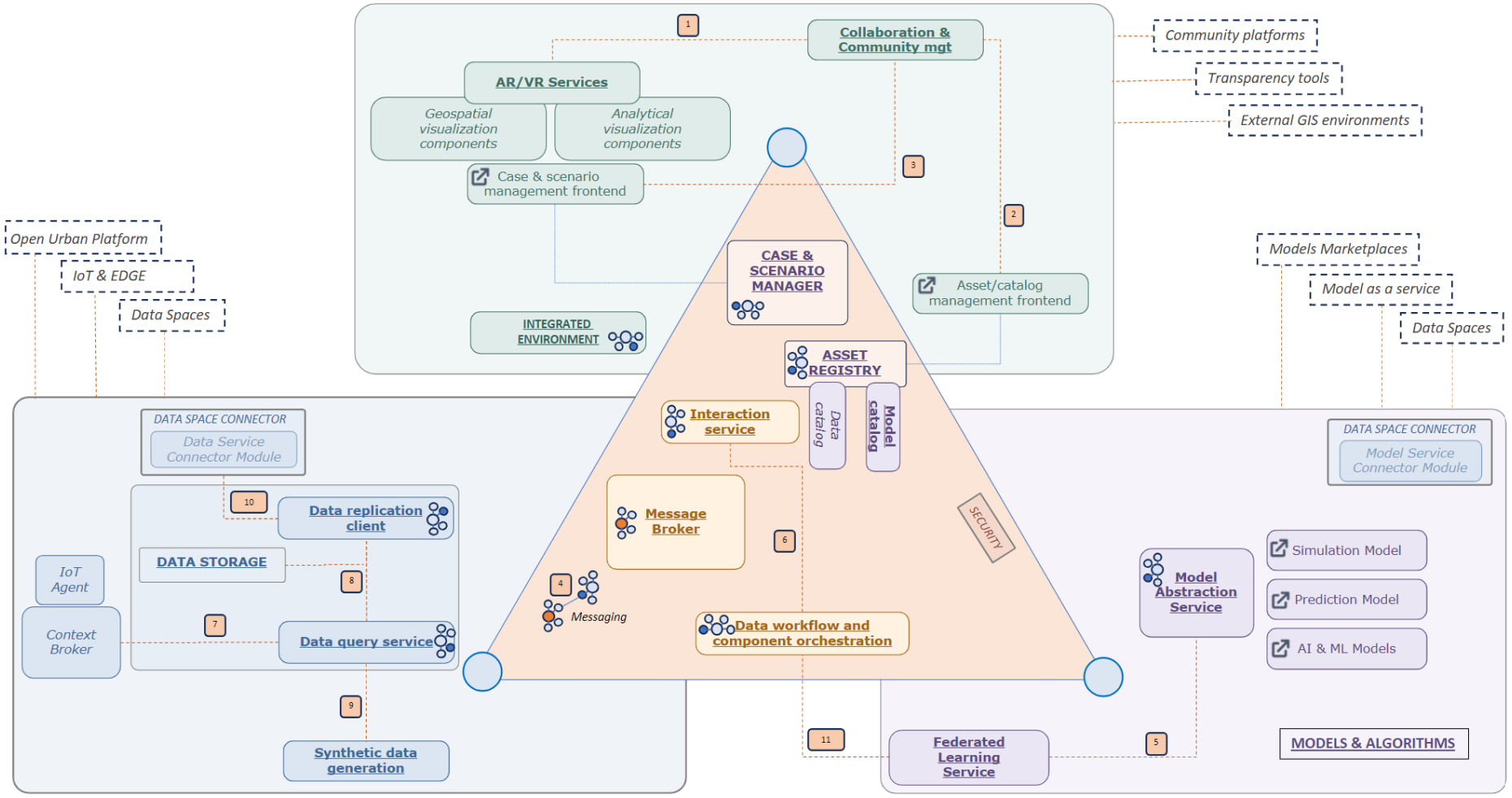


Figure 4: System of Systems vision on the BB interactions

Source: Consortium

The goal of the above reference interaction diagram within the scope of the reference architecture is to **illustrate the role and responsibilities of the different components** and to **highlight the importance of abstraction for interoperability within a scalable system of systems** approach. Note that this is not a system architecture and does not aim to be complete. Instead, it should give a good reference and mind map to understand the description of the BBs specified further in this text. Thus, it is advised to refer to the detailed descriptions of each BB for more information.

However, it is important to note that several important interactions need to be considered when focusing on the reference architecture concepts and principles for an LDT. The colours in the figure are the same as in the BB view in Figure 3 (green for the user experience and visualisation, blue for the data-centric blocks, purple for the algorithm-centric blocks, and brownish in the middle for the core abstraction components of an LDT system itself). An LDT system needs these collections of components to interact with each other and to form the system. Moreover, they also need to be able to interact with other subsystems, indicated by the dashed white boxes, such as data spaces, citizen transparency tools, and community engagement platforms.

Selected interactions have been identified and illustrated to explain the principles of the reference architecture, indicated with numbers (from 1 to 11) in Figure 3. An important remark is that these interactions are chosen to illustrate the reference architecture dynamics, and as such are not an exhaustive summary of all interactions as this is a reference architecture, not a detailed system architecture. Note that these interactions are not indicating any sequence in the flow of component communication but are chosen to illustrate mutual BB functional interactions.

Interaction 1: The Collaboration & community management BB can use AR/VR services to interact with its stakeholders and illustrate the functions of the Digital Twin.

***Example**: In the Urbanage project, citizen engagement is required to realise inclusive decision-making processes regarding the urban planning for age friendly. The objective of a first use case developed in Flanders is to calculate a 'Green Comfort index' based upon location specifications and gather feedback from the elderly people on this Green Comfort Index. A feedback app is developed where elderly people can provide input for improvements on the Green Comfort Index on a specific location for instance by suggesting installing benches or add trees. This project is still under development, but it is foreseen that more interactive UIs will be used in combination with 3D visualisations to interact with the users.*

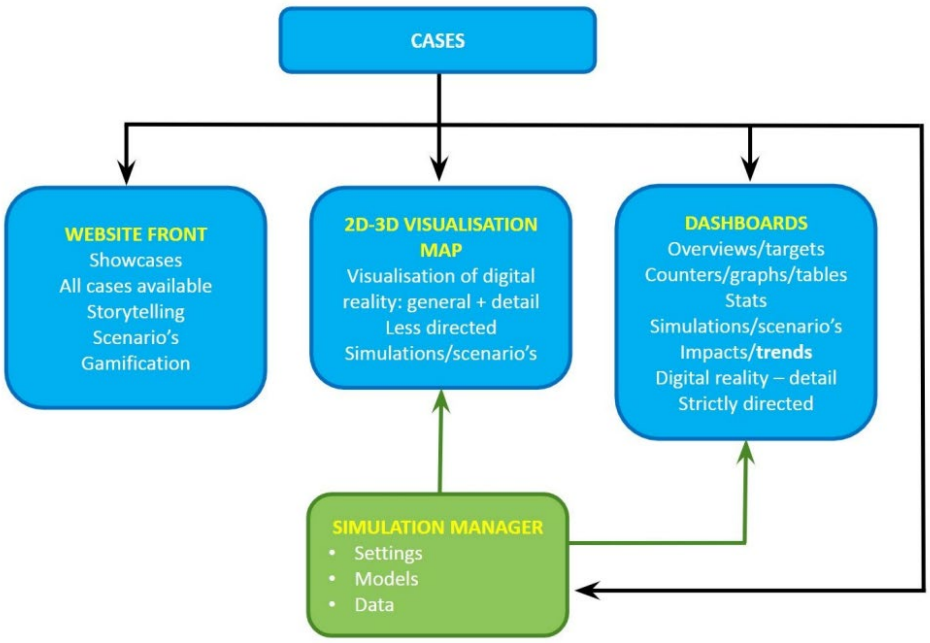
***Example:** The Urban Digital Twin of Herrenberg implemented a VR application for their city and an app for residents to provide feedback and used the interactive visualisations to gather input on city projects[12].*

***Example:** In DUET, some gamification techniques for citizens are mentioned in D4.5 that have a strong potential to enhance interactions with citizens, such as voting, linking, sharing scenarios, ratings, feedback forms, questionnaires, etc. See section 3.3.4 on Gamification.*

Interaction 2: The Collaboration & community management BB can bring transparency of algorithms towards the citizens, such as using an algorithm register that explains various aspects of their functionality, ethics, datasets used and related privacy, among others. For that purpose, it can consult the asset management frontend (linking to data and model catalogues) to get the information with the goal to automate this flow.

[12] How a small German town is using an advanced Digital Twin - Cities Today (cities-today.com)

Example: *Deliverable D2.3 from DUET specifies user requirements for the DUET solution, e.g. specifying a website front that lists cases, scenarios, etc. (e.g. from the Asset Registry) and dashboards, as shown below.*



Source: Figure from DUET, illustrating a part of the Digital Twin frontend bringing transparency

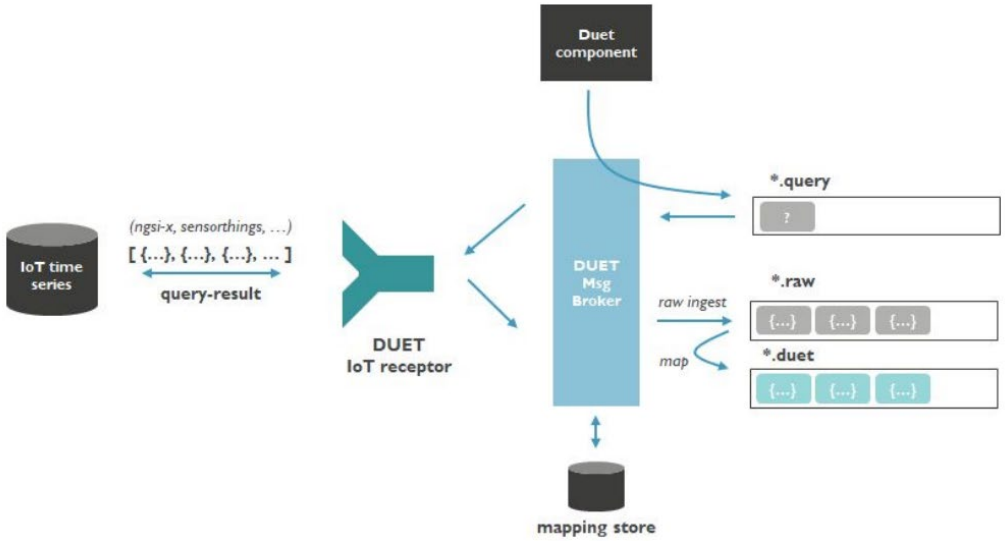
Interaction 3: The Collaboration & community management BB will need to feed the context and output of cases and scenarios to its stakeholders. For that it will have a link with the case & scenario frontend. Also, if the LDT owners want the citizens to execute defined cases and scenarios, this is the interaction to use.

Example: *Deliverable D2.2 of DUET specifies examples of city scenarios that can apply to case & scenario manager, and which describe possible contexts that need to be handled by the case/scenario manager, and of which the results can be fed back to the collaboration BB.*

Also Deliverable D4.4 of DUET gives more context on how cases can be represented towards the stakeholders.

Interaction 4: The message broker is at the heart of the Digital Twin. It provides a clearly defined abstraction of the different messages that can be sent between the different components. For that reason, it enables communication between, for example, the case & scenario manager and interaction service, the data workflow engine, and the data query service, among others. Some of these interactions are described in the respective BB descriptions.

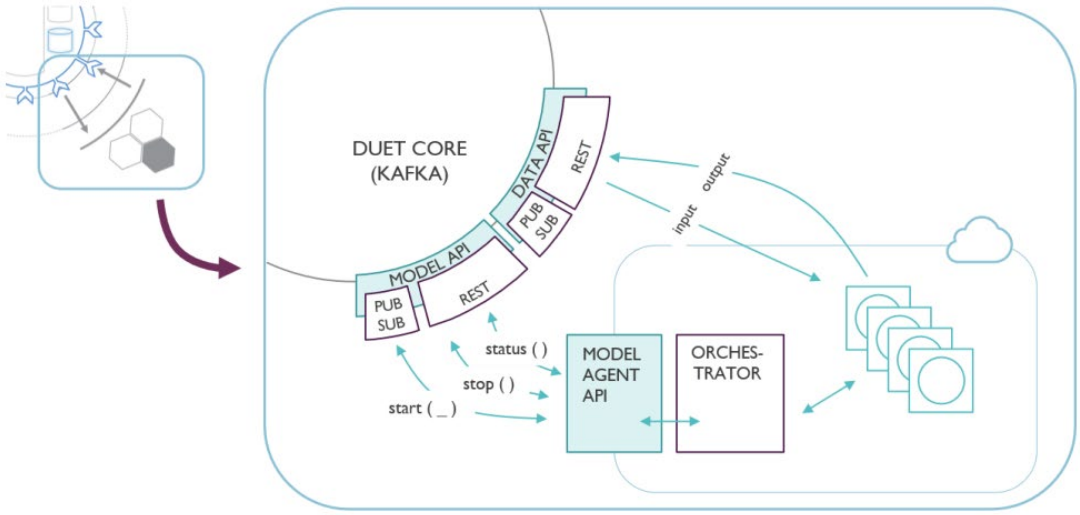
Example: *Deliverable D3.1 from DUET describes a pattern to onboard IoT data into the Digital Twin, connecting to the DUET message broker, and transforming the raw IoT messages into the messages that abstract from the vendor-specifics of the IoT sensor supplier and align communication with the other building blocks.*



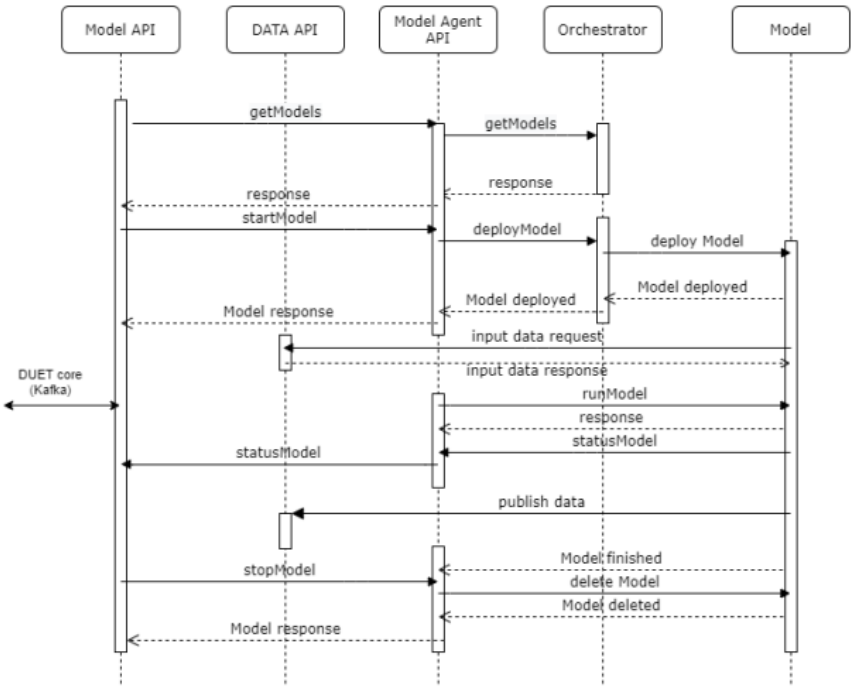
Source: Figure from DUET illustrating the position of the message broker and mapping of data towards the unified Digital Twin messages

Interaction 5: The model abstraction service plays a crucial role in abstracting several model aspects (such as model deployment, execution, signatures, etc.). In this example, it can abstract the fact that models are federated towards its connected BB.

Example: *DUET deliverable D3.5 describes some patterns for interacting with models through a model abstraction service (model agent API in below figure) and orchestration agent that integrates with the workflow and component orchestration engine. More information can be found in the document.*



Source: Example from DUET illustrating the model abstraction service (represented by the model agent API) and the orchestration component (working together with the component orchestration BB)

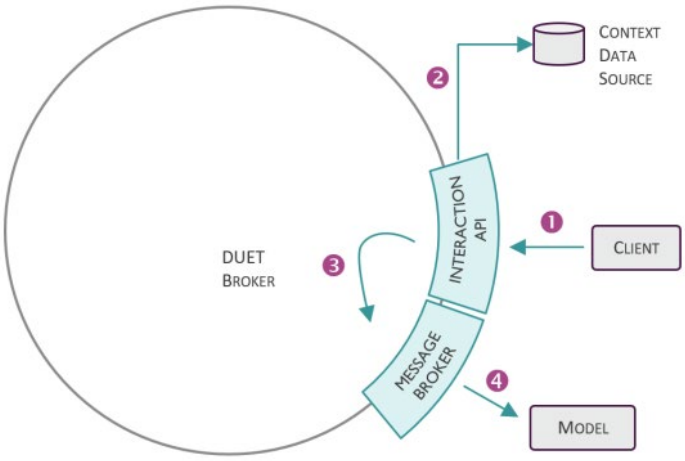


Source: A sequence diagram from DUET illustrating the above figure for interactions with the model abstraction component

Interaction 6: The interaction service is key in keeping and storing consistent state of scenario executions. It provides an abstraction towards the case & scenario manager to manage scenario execution, and has interactions with a lot of other BBs, such as the workflow and component orchestration BB that can deliver information on the consistent sequence of execution of different services that are needed for the scenario.

Example: *An example of how the interaction service in DUET interacts with other components can be found in D3.2 and is illustrated in figures below.*

More information can also be found in section 2.2 of the document.



DUET illustration of the interaction service and its API

In above figure, the different steps are as follows:

1. *client (or a model) sends an update to the interaction API via an HTTP REST web API.*
2. *The Interaction API processes this request by storing the change into a database that keeps the context.*
3. *The Interaction API pushes an update notification message onto a topic associated with the context data source.*
4. *The model subscribes to updates from the context data sources and responds accordingly.*

Interaction 7: The Data Query Service can get data from IoT (or other data platforms) through, for example, standardised context broker APIs.

Interaction 8: The Data Query Service can also get data from the internal LDT storage (e.g., a data lake house) and interact with the data replication client if the current state needs to be updated on top of a recent state of changes, or if changes are needed to be sent to another client BB. The data replication client can also use the data storage system to store its events and associated entities.

Example: *In the SIMPL tender architecture vision document, requirements ID-40 and ID-41 mention that the middleware shall enable infrastructure providers to provide storage service and add an abstraction layer on that. Also, IDSA and Gaia-X enable federated data storage services through data spaces (which can be seen as an abstraction layer). In the reference architecture, the data query service and data replication service offer ways to access federated storage, including the use of local storage capabilities as part of that federated access.*

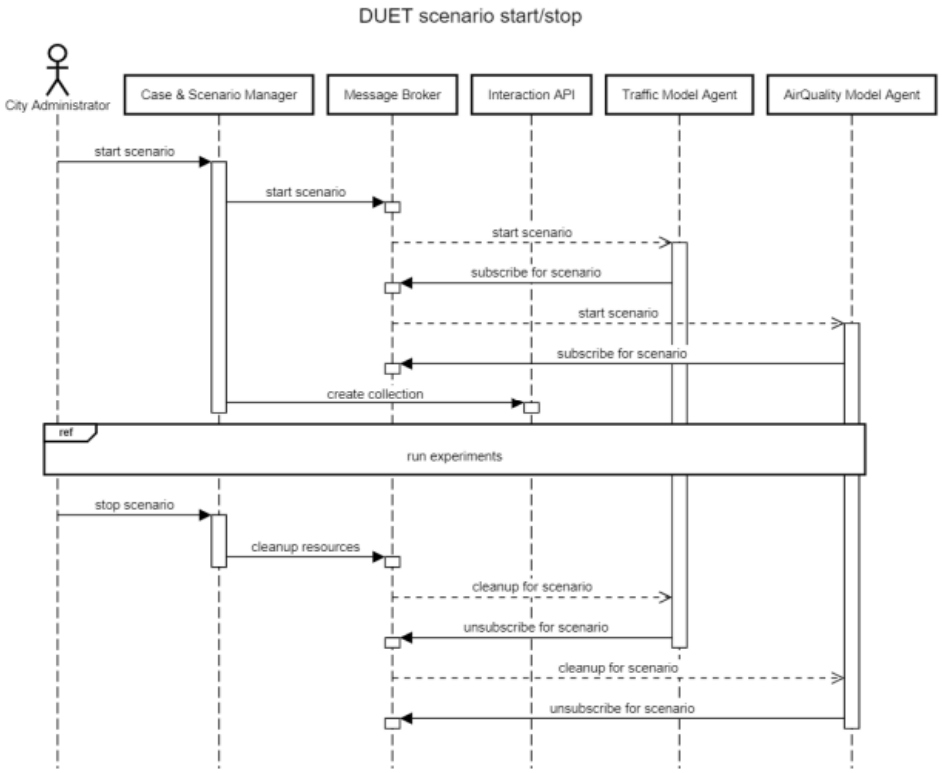
Interaction 9: A Data Query service could also need to directly service synthetic data from that BB.

Interaction 10: The data replication client could subscribe directly to a data space connector and its connected fragmentation or event services.

Example: *D3.2 of DUET describes the Linked Data Event Streams (LDES) protocol as a way to implement event sourcing for replication.*

Interaction 11: The model abstraction service can abstract from federated learning models, but the workflow and component orchestration engine need to be aware of that if the algorithms are deployed from within the LDT itself.

Example: *DUET deliverable 3.9 illustrates some interactions between the model abstraction service (with as examples traffic and air quality model agents), the interaction API, the message broker and the case & scenario manager, e.g. in the use case of starting and stopping a scenario.*



Illustrative example of the use case of starting and stopping a scenario

This is just a selected set of important interactions highlighting the **importance of abstraction** to manage interoperability in a system of systems approach and to **enable horizontal scaling** of LDTs where the BBs and associated tools can be re-used and connect in the end specific instances of Digital Twin instances together. In that sense, LDT knowledge and experience of one LDT community can be shared with other stakeholder communities on the level of assets such as cases, scenarios, datasets, among others.

Use case narrative

In addition to the reference architecture, the description of the BBs and their interdependencies, an example is given on how the BBs can work together based upon a narrative for a use case. Please note that this is a 'fictional and simplified' example on how the BBs can work together, so other implementations and combinations are possible, depending on a city's architecture and data platforms and of course depending on the use case.

Below a 'Persona' is described that represents the user of the LDT and a narrative or storyline is composed that describes how, for a certain use case, the different BBs of the reference architecture are used.

Use Case: Performing what-if scenario analyses to study environmental effects on proposed road infrastructure changes.

A recent trend in urban planning is to study integral effects on e.g., mobility, environmental impact and well-being to support the decision-making process. The complexity of understanding such a scenario and conducting a scenario-based analysis lies in connecting inter-domain dependencies. For example, air pollution and noise are influenced by (car) traffic and therefore it is required for both traffic and air pollution and noise simulations to take place. A Digital Twin should enable quick assessment of the effects of e.g., possible interventions and should facilitate the strategic questions of stakeholders.

Ambition Level: Predictive Twins

Persona 1: City planner – creation of what-if scenarios

As a city planner, I want to create two different scenarios on the road situation to run simulations for a particular two-way direction street: changing the speed limit in a street and closing the street for all traffic. When executing these different scenarios, I want to see the calculated effect for each option on the traffic in the surrounded areas and its impact on air pollution and on noise. I would like to see the results of these scenarios on a street map where I can also see the actual measured results of traffic and air pollution sensors.

In order to be able to create the requested scenarios, the data and models should be integrated in an LDT. A first data source, is the data from sensors that is managed (and stored) by an external IoT platform (IoT and Edge components.) The sensor data from the IoT platform is sent to an LDT through the IoT Agent. The sensor data is sent to the visualisation layer where it is mapped on street data, as the sensor data is linked to geospatial data. These data sources are documented in the data catalogue (identifying specific data values) and Asset Registry (referring to the overall data types). The message broker is responsible for the communication of the data between the different components.

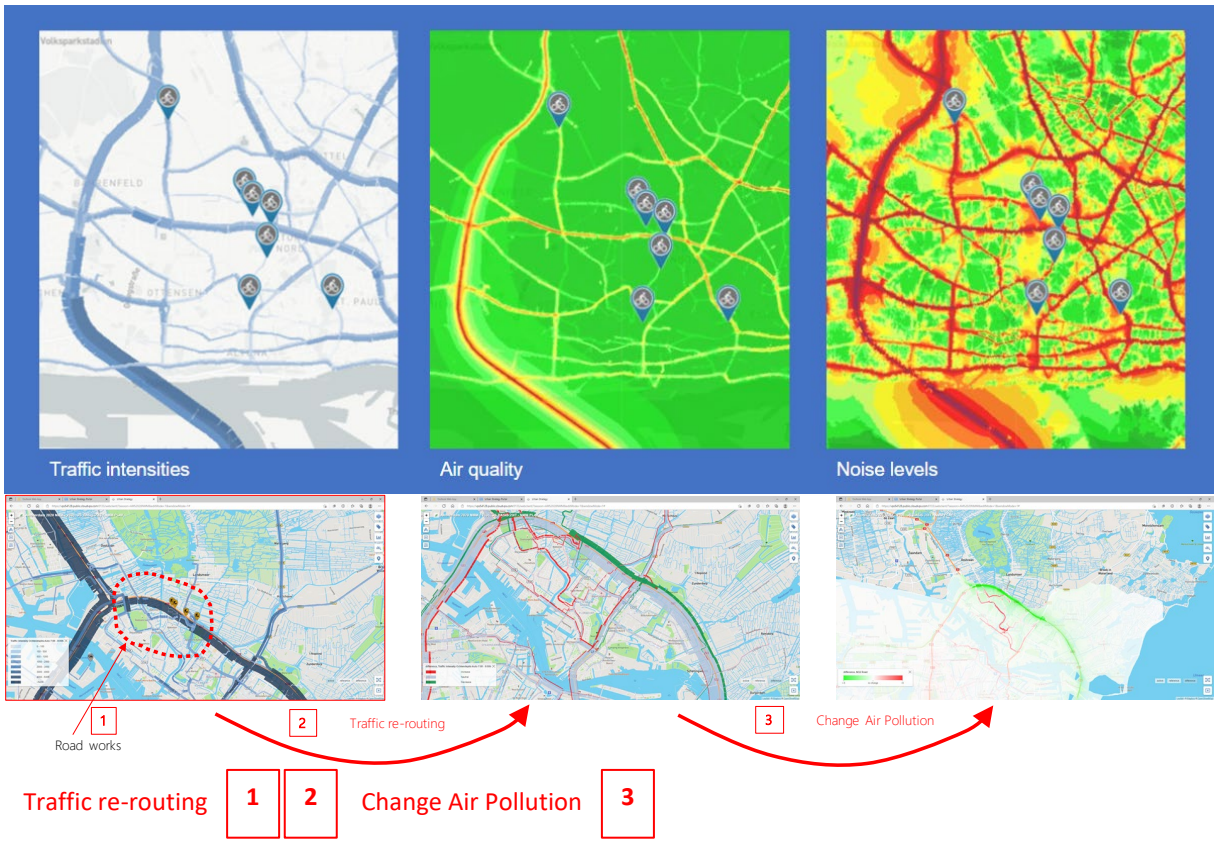


Figure 5: What-if scenario to study environmental effects on road infrastructure changes

As a city planner, I can freely access the 'integrated environment' where I see the actual sensor data on the street maps, but for creating new scenarios, I am required to sign in as only the city planner can create new scenarios (registration via Identity manager, User manager, Access manager). The city planner accesses the case & scenario manager front end and creates his 'case' for a particular street and identifies the two different scenarios. The case & scenario manager is aware of the 'data model concepts' as the definitions of a street and speed limit are defined in the data catalogue. For the required outcome of this scenario, three models are used: a traffic model, an air pollution model and noise model. The specification of these three models is registered in the model catalogue where the system can find the information to be able to connect to the model (the model can be hosted on a

different server). The specifications of these three models are registered in the model catalogue where the system can find the information to be able to connect to the model (the model can be hosted on a different server). The user wants to simulate the effects of changes sent to the traffic model and send this output to the air pollution and the noise model. These steps will be defined as a 'workflow' in the data workflow and orchestration component. The workflow consists of all actions that are needed to perform the scenario: based upon the input parameters for a scenario, the traffic model runs a simulation and the results of this 'simulated traffic model' are sent as input parameters to the air pollution model and air noise model.

The system can access the model to start a model run and receive the output of a model run in an optimisation way by using the model abstraction service to avoid that every model needs specific configuration. The city planner has defined the different scenarios and can now 'execute' the scenarios, which means that the workflow is initiated, and the models will start running: the actual interaction with the LDT core is executed by the interaction service component. The results of every model run are registered as a 'new asset' in the Asset Registry and the data is stored in the LDT data storage. This process is crucial to identify all actions that are made in an LDT environment to support decision-making policies as required by the principle of 'transparency and explainability so the city planner can refer to the exact version, time of when the model run was executed.

In the figure below, the described BBs from the use case narrative are indicated in grey.

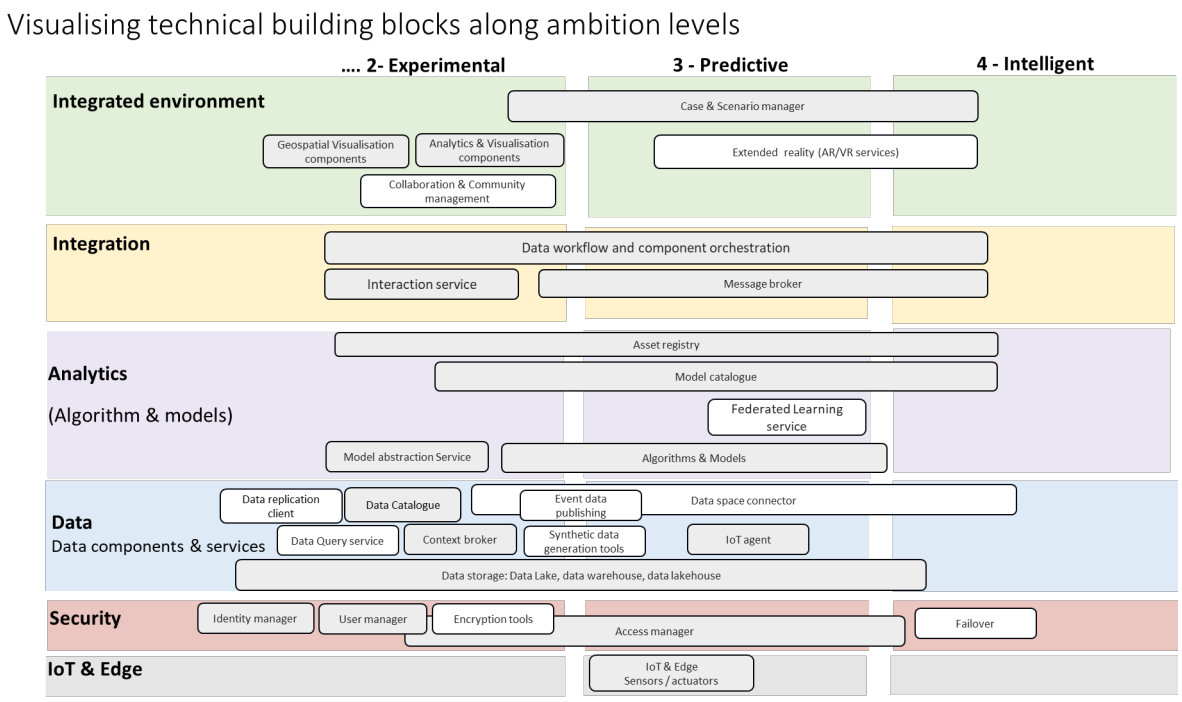


Figure 6: Visualising BBs described in Use case narrative

For the same use case, another perspective can be taken, such as Persona 2: The citizens. As a citizen, I would like to understand the impact of the two possible scenarios of closing the street and reducing the speed limit. I would like to be able to 'visualise' the new scenarios in a virtual environment to better understand the effects of the scenarios and provide feedback.

For this use case, the BB on XR is used. Here, the data from the different scenarios is presented in a virtual reality environment. Citizen's feedback can be requested using an app as part of the collaboration & community management system where the citizen can select the case that was discussed and provide feedback, for instance by given a preference related to the different scenarios that are presented. Again, this data needs to be captured and registered as a new data source in the

Asset Registry and linked to the case & scenario management to give additional information to the city planners to support the decision-making process for urban planning

Evolving from a reference architecture towards a detailed system design, and alignment with standardisation activities.

The goals of the reference architecture were scoped in the beginning of this chapter. It aims to be a **technology-agnostic pattern for guiding the design and implementation of LDT instances at scale**. It allows the necessary flexibility and interoperability choices in the next phase of implementation, which are needed as there is a lack of specific LDT standards (or some of them are incumbent and still pre-conception, like the algorithm register standard of MIM5), and an LDT toolbox can become the de facto standard using open and interoperable technologies in the design phase, of which some examples are illustrated in the BB descriptions further in this document. Other advantages are the fact that customisation towards existing (advanced) Digital Twins is still possible and implementation choices can still be aligned based on adoption of existing implementations.

As an example, ITU has been starting a focus group on smart cities and Digital Twins, illustrating the need to for smart city platforms and associated connecting standardisation like data spaces to be able to handle the evolving requirements of city Digital Twins[13]. A reference architecture and associated technology-agnostic BBs allows the procurement phase to embed a market dialogue in its design phase. Also in above document, it is mentioned that it is "*vital that the Digital Twins developed for different purposes can, over time, be added or linked together to enable more and more problems to be solved*", which is exactly the purpose of this reference architecture and its system of systems approach focusing on vertical and horizontal interoperability by abstraction. ITU (and ISO) have started the standardisation activities on Digital Twins on several levels[14], but there is still work to be done to come to a standardised reference architecture that integrates seamlessly with other reference architecture for smart cities. The reference architecture proposed in the Digital Twin toolbox can be an important input for these study groups.

As an analogy and example to illustrate the work described in this document for LDTs, this report refers to **ITU-SG13 published Recommendation ITU-T Y.3090**[15] that specifies a Digital Twin (communication) network (DTN) specifying a reference architecture, functional and service requirements, and security considerations. This recommendation contains four key characteristics that are also reflected in our system of systems approach: data as the cornerstone (represented by our data subsystem), models as the ability (represented by our analytics subsystem), real-time and interactive mapping (represented by our user experience subsystem) and a standardised interface (represented by the core of our LDT system). As our work for LDT is very in line with the approach in this Recommendation, its common functional and service requirements can be adopted for the LDT, which in essence also need to map physical networks of different nature.

This reference architecture is a good starting point to allow the next phase of building LDTs to have the flexibility to align with ongoing standardisation activities (like ITU-T described above, or the SIMPL data space middleware tender) and go into dialogue with the market to align with de facto industrial best practices, both from the specific technology best practices illustrated in the example descriptions from the BB specs, as from the existing Open Urban Platform and Data Space implementations. The deployment roadmap phases for the toolbox (prepare, design, implement,

[13] Section 2.6 in https://www.itu.int/hub/publication/t-tut-smartcity-2022-05/
[14] e.g., https://www.itu.int/en/ITU-T/focusgroups/ai4ee/Documents/T-FG-AI4EE-2021-D.WG1.11-PDF-E.pdf)
[15] https://www.itu.int/itu-t/recommendations/rec.aspx?rec=14852&lang=en

deploy, operate) are described in deliverable "D05.03: Roadmap proposal for the deployment of the LDT Toolbox".

The description of steps listed below illustrate a plan (to be executed in the next phase of the procurement) in the ***prepare-design*** **steps to use the reference architecture as a guide in these steps and translate it into a system design and implementation plan**, maximising alignment with the market state-of-the-art could be:

1. **Assessment and customisation**: understand and use the reference architecture (including the BB specs!) to align it with the technology and standardisation landscape, at the moment of procurement start, and establishing direct links with the stakeholders (like ITU-T, SIMPL contractor,) so that customisation is possible from different sides. (As stipulated by ITU-T).
2. **Identify and engage specific stakeholders and their expertise:** support the next phase of the procurement process (e.g., from the specific stakeholders like SIMPL contractor) to assist with the translation of the reference architecture into a system IT architecture and define the more specific technical and functional requirements.
3. **Conduct a gap analysis:** analyse the current IT landscape using the reference architecture and the identified stakeholders and decide on possible next steps where alignment is not trivial. This could e.g., lead to step 4 for certain parts of the reference architecture.
4. **Market dialogue (e.g., RFP):** it is possible for certain components to issue a Request for Proposal to get more insights on the procurement details and the market state-of-the-art, focusing on the interoperability mechanisms. If this is needed, the reference architecture can be a key part of the RFP.
5. **Elaborate specific requirements:** based on the above steps, elaborate the requirements already listed with additional details if needed and focus on the BBs of the reference architecture, and the relations between them.
6. **Select technologies and supporting tools:** at this point, the reference architecture has been mapped to the stakeholder landscape, and emerging or established technologies and tools (as already hinted in the examples) can be selected to feed a BIY (build it yourself) or buy strategy.
7. **Design the solution:** after a careful choice of technologies that can align with a standardisation and market reality, the detailed design of the (implementable) system can be done, clearly reflecting the guidance of the reference architecture.
8. **Compliance and governance:** as this is not a one-time effort, and Digital Twins (and their connected subsystems) will be in continuous evolution, governance mechanisms must be set out to guide the evolution of the design and the associated implementation guided by the reference architecture (evolutions).
9. **Next phases (out of scope in this description):** prototyping, development and implementation, documentation, change management, collaboration with vendors and partners, monitoring and metrics, licensing and IP considerations, continuous evolution, among others.

Examples reference architecture and system architecture

Inspiration for an LDT reference architecture and example of other useful reference frameworks of related digital concepts can be found in the table below.

Table 7: Examples of reference architectures from relevant projects and initiatives

Project or initiative	Link
DUET (Digital Urban European Twins) project)	https://www.digitalurbantwins.com/technical-approach
Urban Strategy Platform Architecture	Building Digital Twins of cities using the Inter Model Broker framework – ScienceDirect
Data Spaces for Smart and Sustainable Cities and Communities (DS4SSCC) project	https://www.ds4sscc.eu/ https://www.ds4sscc.eu/catalogueofspecifications
IDS Reference architecture model	https://docs.internationaldataspaces.org/ids-knowledgebase/v/ids-ram-4/
GAIA-X	https://docs.gaia-x.eu/technical-committee/architecture-document/22.10/
Digital Twin Consortium	https://www.digitaltwinconsortium.org/wp-content/uploads/sites/3/2023/07/Platform-Stack-Architectural-Framework.pdf https://www.digitaltwinconsortium.org/wp-content/uploads/sites/3/2022/06/Digital-Twin-System-Interoperability-Framework-12072021.pdf
ITU's U4SSCC reference framework for a Smart Sustainable City management	https://u4ssc.itu.int/publications/
UK National Digital Twin programme	https://www.cdbb.cam.ac.uk/what-we-did/national-digital-twin-programme https://www.cdbb.cam.ac.uk/files/architecture_principles_final.pdf

Examples of Digital Twin Platforms (built within EU projects) of which the system architecture is highly in line with the reference architecture.

DUET: see the references mentioned above in the examples to illustrate this.

LEAD: https://www.leadproject.eu/wp-content/uploads/2021/12/LEAD-D2.5.pdf

The high-level architecture of this platform is shown in the next figure and illustrates the system of systems approach (connections to the physical world and data and model federated spaces), where the LEAD platform contains very similar building blocks as scenario management (cfr. Our case/scenario manager), model and ontology store (cfr. Asset Registry and model catalogue), security, context entity management (cfr. Context broker and Asset Registry), simulation orchestration engine (cfr. Data workflow and component orchestration and model abstraction service), model management (cfr. Model abstraction service and model/algorithms), configuration and decision system (cfr. Integrated environment and visualisation), IDM and authentication, among others.

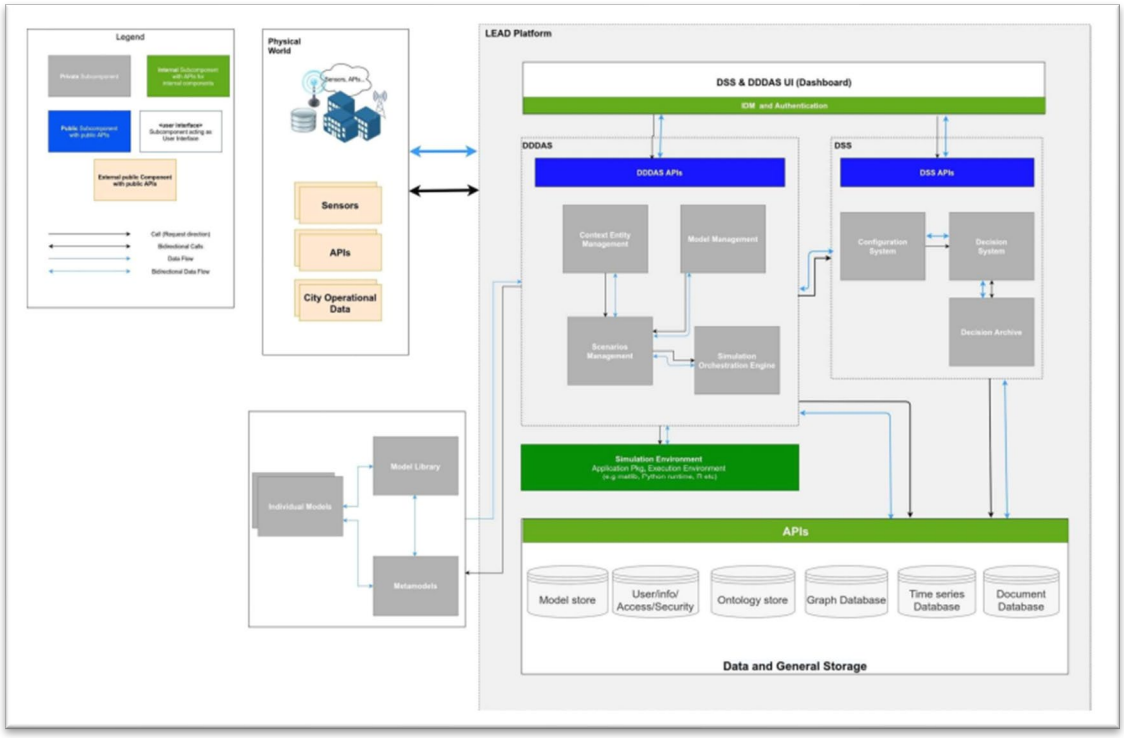


Figure 7: High-Level platform architecture from the LEAD project

This architecture also uses a message broker and storage system to connect the dots, as shown below.

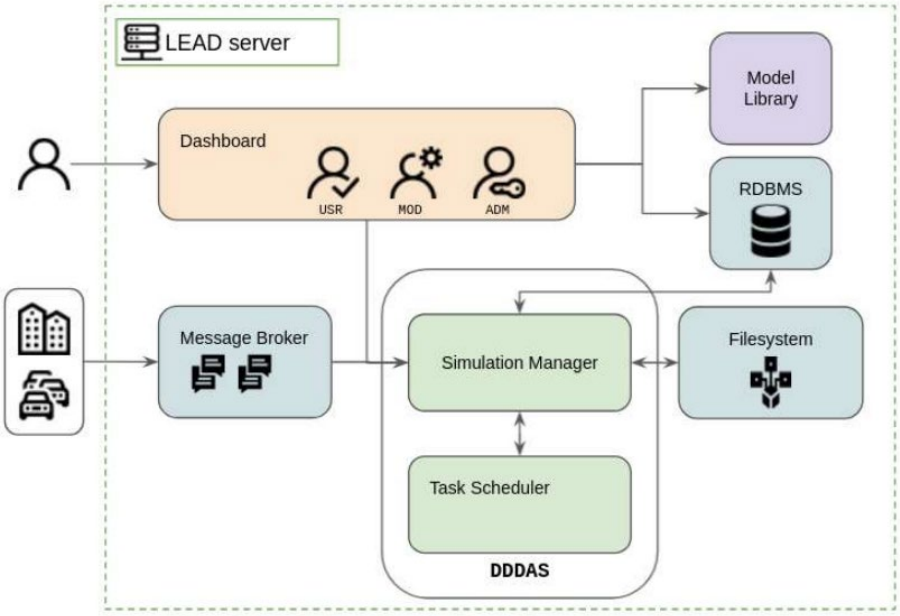


Figure 8: Relations between components in the LEAD project

And eventually leads to some technology choices within the deployment of the platform as shown below. These technology choices are also mentioned as examples within the BB descriptions further in this document.

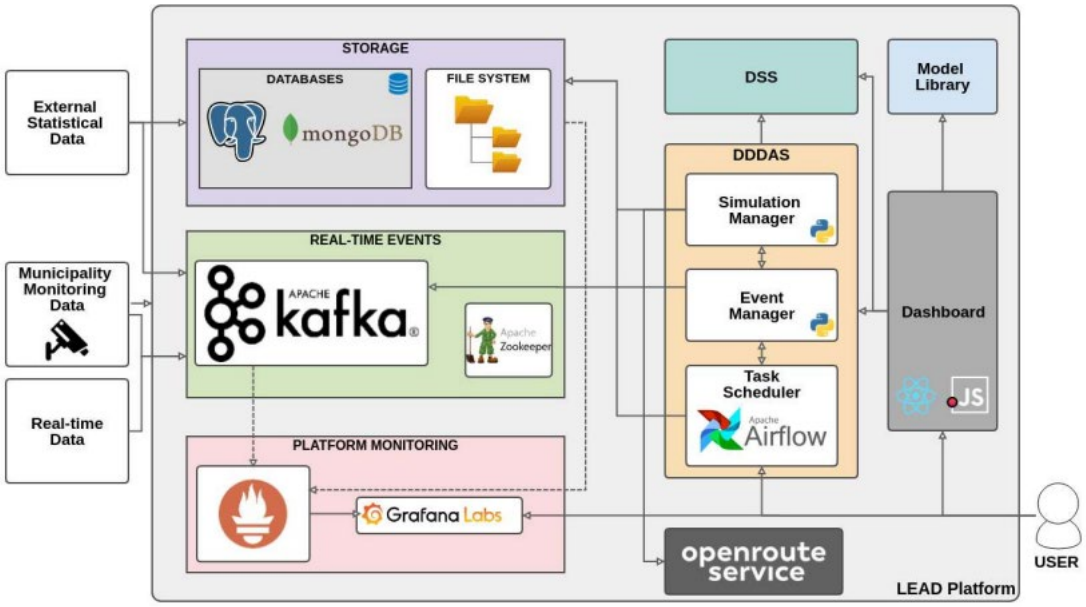


Figure 9:Technology choices for the deployment of the LEAD project

3.2.2 Building Block 02 | Case & Scenario Manager

Table 8: Summary Table of Building Block 02

Minimum Ambition Level	2 – Experimental Twin
Kind	Software/ Guidelines
Maturity Level	No Existing Standard
Internal code	BB.02
Relevant MIM	MIM5

Description

The Case Management Service that can be built upon this BB, plays an essential role in an LDT system to manage what-if scenarios in an Experimental Twin. It is the hub where all the Digital Twin BBs come together and can be managed via its exposed API. This service bundles the following concepts:

Cases are detailed instances of specific problems being addressed, such as planning a traffic circulation scheme or evaluating the potential impact of a new housing development. They include all factors involved in the problem and are designed to document and communicate the situation to all stakeholders, including the general public. After a resolution has been reached, cases may also include a report explaining the rationale behind the decision, detailing the data and insights used to arrive at this conclusion.

Scenarios, on the other hand, operate within cases. They are configurations that can be used to evaluate different potential solutions to the problem defined in the case, applying data and/or simulation models to assess the impact of each solution. Scenarios specify which models are applied to which data sources, with this static configuration stored in a scenario record.

Scenarios can also be used to compare and evaluate the performance of different models (referencing the **model catalogue**) or different configurations of the same model. In this setup, the scenario references the **data catalogue** (see Annex 6.1) for data sources of interest, which can include input data sources for models or visualisations, as well as output data sources. By keeping output data sources separate, this setup facilitates later comparison of results, which can aid in decision-making and ongoing optimisation efforts.

Experiments are specific model/data source configurations within a scenario. They represent distinct sets of inputs, parameters, or conditions applied to the models. For instance, an experiment might involve a specific traffic model run with a particular set of sensor data or different population density parameters. In that sense, experiments configure specific scenario parameters.

Each experiment could generate a set of results and data when the model is run. These results and data can be stored and associated with the specific experiment configuration. The stored results can then be used for further analysis, comparison with other experiments, or to validate or calibrate the models. The latter is especially relevant when collaborating with multiple models (e.g., traffic, noise and air models). For example, when a traffic simulation model computes new intensities based on its input data, an associated noise simulation model should operate instantly on these new intensities produced by a traffic simulation model.

This creates a multi-level structure where a **Case** defines the overall problem, a **Scenario** outlines a solution or approach, and an **Experiment** represents a specific configuration of models and data used to evaluate the scenario. The results from these experiments then feed back into the system, informing decision-making and potentially leading to adjustments in the models or approaches used.

As mentioned, the Case and Scenario manager is the primary interface for the Digital Twin user and has important links to other core BBs mentioned in this document, of which the most important links are described below with examples of their key interactions.

Asset Registry

- Cases, scenarios and experiments must be managed themselves as assets within the LDT and thus need to be governed to allow defining and changing metadata schemes, track lineage and provenance of changes, point to vocabularies used within their metadata, among others. So, they need to be configured within the Asset Registry.
- Cases, scenarios, and experiments lead to the use of data and model assets that are governed and managed within the Asset Registry. Linking to and using the (versioned) information from the Asset Registry allows to contextualise and enrich the interpretation in the most qualitative way, even for archived versions.

Collaboration & community management system

- Cases, scenarios and the output of experiments are key elements in explaining the possibilities of Digital Twins but can also point out the tricky points in using them. As such, they can play a vital role in feeding the collaboration with the community and induce feedback.
- Thus, it is especially important that different cities and communities can communicate their cases, scenarios, and experiments in a unified way, so that discussion between cities can be leveraged. It is for that reason that this BB plays a significant role in inducing the o of these elements forward, at least by providing them as assets with defined schemes, vocabularies in such a way that a collaboration and community management system can easily absorb them.

Message Broker

- As mentioned, BB.02 is at the heart of the Digital Twin and needs to interact with other BBs, for example to operate (start/stop) scenarios using data and model assets. Thus, it will be heavily interacting with the message broker as sender and receiver of messages with the

different components of the Digital Twin. The message broker communication abstraction (topics) will therefore need to be enriched with the communication and interaction.

Model and Data catalogue

- As described above, scenarios specify which models are applied to which datasets at the end. It is evident that they will point to models and data sources that are listed in the catalogues, such that they can be referenced and provenanced.

Data Query Service

- An experiment can call on different data sources and models. Thus, it is important that the necessary data can be queried or requested, which can be managed and modified to be able to conduct the right scenarios and experiments.

Data workflow and component orchestration

- A scenario can call for several models and use multiple data sources to list them. In that sense, it will need to interact with and trigger the workflow engine to orchestrate the execution of the scenario.

Interaction service

- Scenarios are executed in steps and trigger changes which lead to interactions with other BBs. For example, adding sound barriers, closing down roads, or updating the virtual state of a city after a computation. All of these changes need to be tracked so that the Digital Twin becomes interactive, and changes can be replayed. The interaction service provides the way to do this and BB.02 can use it to leverage and track interaction.

Capabilities, functional requirements, mechanisms and interoperability guidance

This component addresses the main capabilities listed below. The case and scenario management capability is needed starting from experimental twins as a core capability but is essential in the end to enable applying intelligence to Digital Twins, enable continuous and incremental learning and track the context of usage to make it explainable in the end.

Table 9: Capabilities Table of Building Block 02

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards
2 – Experimental Twins	32	Case, scenario & experiments management	Integration	MIM5	None
4 – Intelligent Twins	48	Usage context information provisioning	Governance & Compliance	MIM5	None

The case and scenario management BB triggers the execution of algorithms on data and is therefore the key element that can link certain output of algorithms to the functional context in which this was used. Applying model and data assets can only be made accountable if it is linked to the functional usage. Because of this, when applying MIM5 within LDTs scope, the identified components and its assets play a vital role. MIM5 requires following capabilities: Procedural & technical transparency, technical explainability, fairness, context and accountability, so these are important requirements for this BB.

The next table summarises the main requirements for BB.02 with an indication of which technical mechanism can fulfil this, and pointers to possible best practices or standards that can enable interoperability.

Table 9: Mechanism Table of Building Block 02

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1	32,48	A uniform interface for case, scenario, and experiment management shall be used.	RESTful API or GraphQL for CRUD operations on cases, scenarios, and experiments	REST or GraphQL standards
2	32	Case, scenario, and experiment data must be securely accessible and modifiable by optimisation users only (Case Owners, Case Users).	Role-based access control (RBAC)	Oauth2 or OpenID Connect for access management
3	32	Case Guests must be able to consult cases, scenarios, and experiment results.	Access management providing read-only permissions for Case Guests	Oauth2 or OpenID Connect for access management
4	32	The system must support the creation and management of multiple independent scenarios and experiments within a case.	Hierarchical data model with a one-to-many relationship between cases, scenarios, and experiments	N/A
5	32,48	Changes to case, scenario, or experiment data must be logged and traceable.	Event Sourcing or Audit Logging	N/A
6	32	The system must support the retrieval of historical case, scenario, and experiment data.	Time-travel queries or versioning support	N/A
7	32,48	Case, scenario, and experiment data must be exportable in a standard, machine-readable format.	Export functionality supporting JSON, JSON-LD, or YAML formats	N/A
8	32,48	Case Users must be able to run experiments with specific parameters.	API endpoints to manage experiment parameters and execution	REST or GraphQL standards
9	32,48	The system must support linking cases, scenarios, and experiments to relevant assets.	Asset reference fields in the case, scenario, and experiment data models	N/A

10	32,48	Changes in linked assets must be traceable within the context of cases, scenarios, and experiments.	Event Sourcing or Audit Logging for assets	N/A
11	32,48	The system must support the retrieval of asset data associated with a specific case, scenario, or experiment.	API endpoints to retrieve linked asset data	REST or GraphQL standards
12	32,48	The system must allow updating asset associations within a case, scenario, or experiment.	API endpoints for updating asset references	REST or GraphQL standards
13	32,48	The system should offer an overview of all assets related to a case, scenario, or experiment, including their metadata and provenance information.	API endpoints to retrieve asset metadata and provenance information	REST or GraphQL standards
14	32,48	Case Owners and Case Users should be able to add or remove assets linked to a case, scenario, or experiment.	Role-based access control for managing asset links	Oauth2 or OpenID Connect for access management
15	32	Scenario descriptions must be available in a structured, machine-readable format for use in data workflow and orchestration.	JSON, JSON-LD, or YAML formats for scenario descriptions	N/A
16	32	The system must provide API endpoints for retrieving scenario descriptions.	REST or GraphQL API for scenario descriptions retrieval	OpenAPI specification, GraphQL schema
17	32,48	Scenario descriptions should include details about data sources, data transformations, models to be applied, and expected outputs.	Detailed fields in the scenario description data model	N/A

Examples

Currently, no standards or solution for case & scenario management have been found. However, it should be possible to envision an open API specification along with some business rules for the creation of such a service. Ideally, the solution is accompanied by an ontology and supports **json-ld**. This will help maintain interoperability and extensibility overall. An example outline of the open API specification is highlighted below.

It is important to consider that the data of the case and scenario manager could be used by the workflow and orchestration BB to understand what models are linked to what data sources when running an experiment. The notion of a case and scenario manager was introduced in the DUET project, where it was implemented as a set of micro-services.

Below is a list of examples regarding high-level open API specification, containing the need for schemas for cases, scenarios, and their experiments.

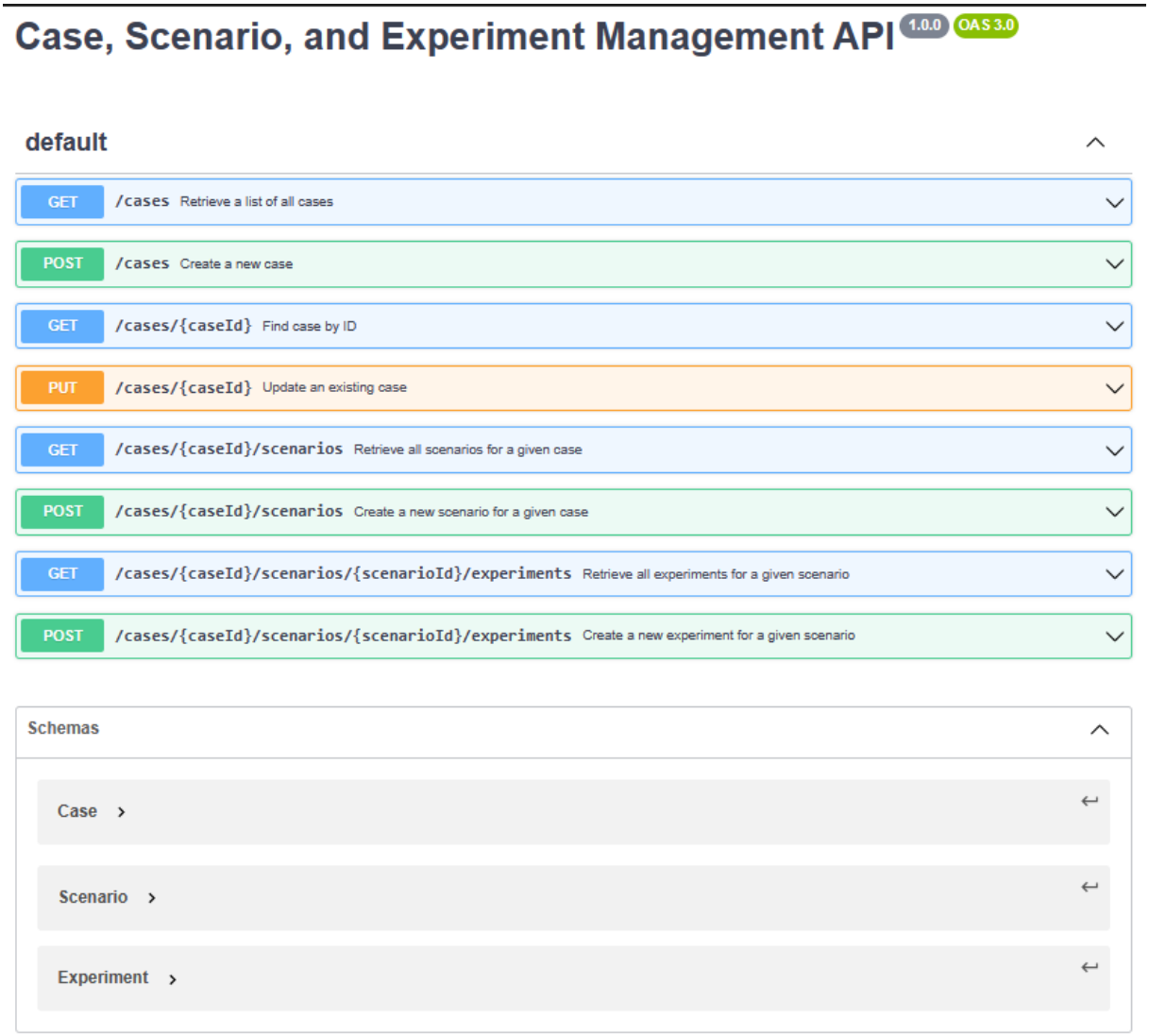


Figure 10: Example of API Specs Case & Scenario Manager

A more detailed suggestion for the data model scheme could be the following:

Table 10: Illustrative Data Model Scheme

Entity / attribute	Description	Type
Cases		
CaseID	Unique identifier for each case.	UUID
CaseOwner	The user or entity responsible for creating the case.	String
CaseDescription	A brief explanation of the case.	String
CaseCreatedDate	The date and time when the case was created.	DateTime
CaseStatus	The current status of the case.	String

CaseMetadata	Any additional data related to the case.	JSON-LD
Scenarios		
ScenarioID	Unique identifier for each scenario.	UUID
CaseID	The ID of the case that the scenario is part of.	UUID
ScenarioOwner	The user or entity responsible for the scenario (Reference).	UUID
ScenarioDescription	A brief explanation of the scenario.	String
ScenarioCreatedDate	The date and time when the scenario was created.	DateTime
ScenarioStatus	The current status of the scenario.	String
ScenarioParameters	The specific parameters of the scenario.	JSON-LD
ScenarioMetadata	Any additional data related to the scenario.	JSON-LD
Experiments		
ExperimentID	Unique identifier for each experiment.	UUID
ScenarioID	The ID of the scenario that the experiment is part of.	UUID
ExperimentOwner	The user or entity responsible for the experiment (reference).	UUID
ExperimentDescription	A brief explanation of the experiment.	String
ExperimentCreatedDate	The date and time when the experiment was created.	DateTime
ExperimentStatus	The current status of the experiment.	String
ExperimentParameters	The specific parameters of the experiment.	JSON-LD
ExperimentResults	Link to the output data source (if available).	UUID
ExperimentMetadata	Any additional data related to the experiment.	JSON-LD

3.2.3 Building Block 03 | Collaboration & community management system

Table 11: Summary Table of Building Block 03

Minimum Ambition Level	2 – Experimental Twins
Kind	Software/ Guidelines
Maturity Level	Good Enough
Internal code	BB.03
Relevant MIM	MIM4, MIM 5, MIM6

Description

This BB is part of an LDT reference architecture to support use cases for cities and communities in their LDT environment to facilitate collaboration and create community practices. The Collaboration & community management system BB is a combination of tools and processes aimed at involving the public in decision-making, policy development, and service improvement. These practices are particularly important for governments, organisations, and institutions seeking to gather insights, opinions, and feedback from citizens or users to enhance their operations and address their needs effectively.

Citizen engagement refers to the active involvement of individuals in public or optimisation affairs. It is a process that fosters communication, collaboration, and participation between citizens and government entities or other organisations. The main goal of citizen engagement is to ensure that decisions made, and policies implemented are more inclusive, transparent, and representative of the public's preferences and concerns.

With regards to the interaction with citizens, building a trustful, sustainable relationship is key for success. In this respect, the MIMs related to Trust, Transparency and Security are crucial requirements for the foundations of all processes and tool implementations for citizen engagement. The usage of a chatbot functionality implies for instance that AI modelling is used, and this should be made transparent.

Key aspects of this BB are:

- **Communication**: Open and effective communication channels are established to encourage interactions between citizens and decision-makers. These channels can include town hall meetings, public forums, surveys, social media platforms, and dedicated websites.
- **Participation**: Various methods are employed to enable citizens to actively participate in the decision-making process. This can include seeking public input on policy proposals, soliciting feedback on community projects, or involving citizens in task forces and committees.
- **Accessibility**: Ensuring that the engagement process is accessible to all citizens, regardless of their background, location, or ability, is essential. Efforts should be made to accommodate different demographics, including marginalised groups and people with disabilities.
- **Transparency**: Providing clear information about the decision-making process and how citizen input will be used fosters trust and transparency.
- **Feedback Incorporation**: The feedback gathered from citizens should be carefully considered and incorporated into policymaking and project development to demonstrate the impact of citizen engagement and show that their contributions are valued.

- **Case & Scenario Feedback Gathering**: A more specific process that involves obtaining feedback on hypothetical situations or specific instances. This approach is often used in user research, product development, and risk assessment. This has a clear dependency on the Case & Scenario management BB, which provides the capabilities to design, simulate and share the cases and scenarios.

Tools with which to accomplish citizen engagement are:

- **User Interviews and Surveys**: Feedback can be collected through interviews or surveys where users are presented with the scenarios and asked to provide their thoughts, preferences, and reactions.
- **Observations and Usability Testing**: In certain cases, direct observation and usability testing might be employed to understand how users interact with products or services when faced with specific scenarios.
- **Iterative Process/Co-design/Co-creation**: Feedback gathering is often an iterative process, where initial feedback informs the refinement of scenarios and the collection of further feedback. To enhance the organisational setup and processes for this process, Living Labs can be organised to stimulate user-centric co-creation. Co-creation can be based either on feedback for operational purposes or for future design decisions. The tools with which cities simulate cases and scenarios can also be made accessible to citizens to facilitate this process.
- **Feedback analysis**: Applications can be developed to interact with an LDT environment to gather input from a specific user group. The feedback collected from users is analysed to identify patterns, trends, and pain points. This analysis helps in making informed decisions and improvements based on user preferences and needs.
  Social Media: Additionally, software platforms that provide the tools and digital spaces to communicate (through diverse media), collaborate, discover target audiences, segment and construct specific user panels, etc., is also needed to form a feeding ground to apply the citizen engagement processes on.

The specific requirements on functionalities and associated tools are very much dependent of the domain and uses cases that are covered by an LDT and how the city organises the decision-making process to incorporate citizen participation. An overall approach for determining the tools and the process for citizen engagement is described in following process steps, illustrated by examples.

Identify domain of the LDT for which citizen engagement is required and identify the purpose.

- As part of open government processes, it is important for a city or community to define the objectives for the citizen participation approach. OECD developed a Deliberative Democracy Toolbox to support cities and communities in this process[16] and Eurocities also provide guidelines related to the approach and principles to be considered regarding citizen engagement[17].
- Urban planning is a very valuable domain to stimulate co-creation with all stakeholders and in particular to engage the citizens as they can pro-actively give input but are also valuable sources to provide feedback on existing situations. As defined in the 'Citizen's City' philosophy[18], it is important to engage citizens in different processes related to city governance, investing in the city and planning the city?

[16] Innovative Citizen Participation - OECD
[17] Citizen engagement - Eurocities
[18] The Future of Cities | European Commission (europa.eu)

- The City of Rotterdam[19] is exploring the use of Digital Twin for involving citizens in urban planning e.g., visualisation of new building plans with AR, or feedback from citizens on new city plans for city squares like where to place benches, etcetera.

Define process(es) where citizen engagement is required and takes into consideration all stakeholders. Based upon an overall 'integrated' process, it should be clear how the citizen engagement is part of the overall decision-making process.

- An example on how to validate that the process design includes all stakeholders and avoids conflicts between diverse points of view is the framework provided by the geodesign approach[20].

Define the high-level use case requirements and target groups. The target group is important to specify as it defines the UX and UI requirements for tools to be used to engage the citizens in the most effective way:

- High level use case description involves stakeholders analyses to define the required interactions. As an example, an overview is presented of stakeholder requirements for Citizen Participation chatbot.

As a..	I want to/need to..
Citizen	Talk to a Chatbot to participate in the ideation process using natural language.
	Submit new ideas for government representatives to consider in their policy making.
	Browse existing ideas submitted by other citizens and vote on them.
	Be able to interchange between different tasks without disrupting the flow
	Talk to the Chatbot without necessarily following the same conversation every time
Chatbot Admin	Label conversations for usage in training.
	Retrain chatbot AI models.
	Reconfigure various Chatbot parameters.
	Easily add new functionality to the Chatbot and integrate with external platforms.
Government Representative	Browse existing ideas submitted by citizens
	Analyze ideas by applying filters and visualizing statistics.

Figure 11: Example stakeholder requirement for Citizen Participation[21]

- When defining the target group, it is useful to be very specific. For instance for elderly people, a user interface should be very intuitive and simple and might require a custom-developed app.

[19] PowerPoint-presentatie (oascities.org)
[20] Geodesign for collaborative spatial planning: three case studies at different scales - ScienceDirect
[21] (PDF) An Architecture for Dynamic Conversational Agents for Citizen Participation and Ideation (researchgate.net)

- When feedback is required from 'as many people' as possible, for instance when an event is ongoing, it can be a requirement to provide a very easy tool that is accessible via QR code. Commonly used apps can increase the user acceptance willingness to participate and acts as a threshold-breaker. Existing market tools can be analysed to evaluate whether an existing tool is fulfilling the requirements and complies with overall security and integration requirements, for example by doing a specific market search for a type of tool; 7 Free Survey Tools to Gather Attendee Feedback | Capterra

Describe the use cases (= input for the defining the functional requirements for the tools to support the use cases.)

- The use case description identifies the preconditions, postconditions, dependencies and actions that need to be supported. The use case description can refer to different tool requirements. For instance as described in the prototype of the Urban Digital Twin of Herrenberg[22], to gather feedback on the design of an LDT, the collaborative feedback process implied a mobile unit for VR projection that could be used by different users in different locations and a survey was provided to gather feedback.
- Use cases can also be 'scored' to prioritise the use cases. In the DUET project (Deliverable 2.2[23]), scenario specifications (epics and user stories) were also created regarding urban planning and public engagement for open data. In Deliverable D2.3[24] the user requirements are further described based on scoring criteria for user stories.

Select the appropriate tooling, aligned with the overall city and LDT architectural setup

- In determining the appropriate tooling for citizen engagement, existing applications can be investigated to evaluate if new functionality to interact with the LDT can be added. An example from the COVID period was the raise of new contact tracing applications. These applications are defined initially for a specific purpose of delivering one-way information from the citizen to the government, but they could also be used as an additional dataset to be integrated in an LDT and integrate two-way information sharing by pushing alerts or notification to people for instance on population crowds. This way, an integrated solution delivers additional value for different purposes in an LDT[25].

Define technical requirements and integration requirements

- For each tool to be implemented to engage users' feedback, overall software requirements such as Identity, user and access management and security (e.g., GDPR) and other requirements as defined in Table 13 are to be described.

Implement tools for citizen engagement and interactions

- To ensure a successful implementation of tools and processes of citizen engagement, a thorough testing phase needs to be included that can be organised by a friendly user testing setup within a testing environment where also all integrations with other LDT tools are setup and tested. This also requires an extensive user manual and testing scenarios to be validated by the testers. It can also be a 'tool' requirement that training material and user documentation are made digitally available for the users.

[22] (PDF) Urban Digital Twins for Smart Cities and Citizens: The Case Study of Herrenberg, Germany (researchgate.net)
[23] Microsoft Word - DUET_D2.2_Scenario specifications of the DUET solution (1).docx (digitalurbantwins.com)
[24] 725ca8_c173251232774329a7430bd3f9785db0.pdf (digitalurbantwins.com)
[25] Complex Urban Systems: Challenges and Integrated Solutions for the Sustainability and Resilience of Cities (hindawi.com)

The tools that are used for citizen engagement must interact with the specific environment of an LDT and hence have dependencies with other BBs from the reference architecture.

Case & Scenario Management

Both citizen engagement and case & scenario feedback gathering are crucial for creating a more inclusive, responsive, and user-centric approach in governance, business, and service delivery. They allow organisations to make well-informed decisions that reflect the desires and concerns of the public or user base they serve.

Integrated Environment

The BB has dependencies on the Integrated Environment BB for its UX and UI elements and to comply with accessibility best practices which are especially important for citizen engagement processes.

Extended Reality (XR) (AR/VR services)

The holistic experience that is achieved by implementing XR services contributes to citizen engagement by simulating action-based interactions and exchange of experiences and information. The usage of XR can thus be an integrated part of a full inclusive strategy of collaboration & community management processes and systems.

Capabilities, functional requirements, mechanisms and interoperability guidance

Table 12: Capabilities Table of Building Block 03

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards
2 – Experimental Twins	41	Citizen engagement and Case & scenario feedback gathering	Strategy	MIM4, 5, 6	N/A
	42	Collaboration and community management		MIM4, 5, 6	N/A

Table 13: Mechanism Table of Building Block 03

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1.	42	The Collaboration and Community Management system must have user registration and authentication.	Implement user self- registration and login functionality	Oauth 2.0, OpenID connect
2.	42	The Collaboration & Community management system must allow users to create and manage their communities.	User tagging and bucket creation	N/A
3.	42	The Collaboration & Community management system must provide a platform for users to create and join discussion forums and threads.	Forums, bulletin boards and group chats/threads	Integration with forum tools
4.	42	The Collaboration & Community management system must enable real-time messaging and push notifications.	Chatbots, push notifications, optimisation notification	iBeacons, Firebase

5.	42	The Collaboration & Community management system must support event creation, management, and RSVP functionality.	Allow user to create events and manage calendars	iCalendar
6.	42	The Collaboration & Community management system must offer features for conducting polls and surveys.	Integrate with survey tools	Qualtrics
7.	42	The Collaboration & Community management system must provide multilingual support for content.	Provide localised content	I18n and L10n best practices
8.	42	The Collaboration & Community management system must allow users to manage privacy settings and data sharing.	Comply with local and international privacy regulations	GDPR, SOLID
9.	42	The Collaboration & Community management system must generate reports and analytics on community engagement.	Export data for analysis in specialised tools	SPSS
10.	42	The Collaboration & Community management system must integrate with social media platforms for content sharing.	Integrate with popular social media platforms	Facebook, Twitter, Instagram
11.	42	The Collaboration & Community management system must be mobile-responsive and accessible on various devices.	Implement responsive design	N/A
12.	42	The Collaboration & Community management system must provide a help centre and responsive user support.	Provide a help centre and responsive user support.	Chatbots, support line
13.	42	The Collaboration & Community management system must comply with accessibility guidelines for people with disabilities.	Comply with accessibility guidelines	WCAG 2.0
14.	41	The Collaboration & Community management system must allow users to share different scenarios and cases.	Integrate with the Case & Scenario management BB tools	REST API
15.	41	The Collaboration & Community management system must enable users to provide feedback on specific scenarios or cases.	Incorporate a feedback form or commenting system for each scenario or case.	APIs for integration with feedback systems
16.	41	The Collaboration & Community management system must allow exporting and importing scenarios and feedback data.	Export of scenarios and associated feedback in standardised formats	JSON

17.	42	The Collaboration & Community management system must provide the tools for mass and directed e-mail campaigns.	Integrate with mailing platforms	MailChimp (note: US tool, to be implemented according GDPR standards)

Examples

An overview of citizen engagement platforms is also described as a knowledge item for Net Zero Cities project. As per the knowledge factsheet on Citizen Participation Platforms, three types of platforms are described:

- **E-Participation**: Empowers citizens to leverage digital tools and platforms – including the combination of tools like geographic information systems, Web 2.0, and mobile technologies (encompassing video, mobile messaging, and Internet access) – to effectively communicate, engage, and deliberate on policy and planning challenges.
- **E-Government**: Applies Information and Communication Technologies (ICT) to government operations to boost efficiency, improve public service delivery, and enable cost-effective and swift public interaction with authorities.
- **Open Governance**: Encompasses government data transparency and access, facilitating collaborative initiatives for innovative policy solutions, heightened awareness, public engagement, behavioural change, e-democracy promotion, and transformative service delivery. Linked with open government data, it enables fresh insights into matters and services, inviting participation, commentary, and policy influence to advance citizen engagement.

Apart from this overview, there are more tool examples listed in Table 14, categorised by their main purpose for citizen engagement.

Table 14: Example of tools for Building Block 03

Purpose	Tool	Open source
Participatory platforms and apps / community engagement	Mindmixer	No
	Shareabouts	Yes
	MySociety	Yes
	Bang The Table	No
	Decidim	Yes
	Consul	Yes
	Hoplr	No
	YourPriorities	Yes
	CitizenLab	Yes
Collaborative visualisation platforms	Covise	Yes
	ClimateView (ClimateOS)	No
	PanelKit	Yes
	MetroQuest	No

Policy simulation tools / process engagement tools	ZenCity	No
	Sociocracy 3.0	Yes
	OpenGov Platform	Yes
	Balancing Act	No
Collaborative Decision-making Platforms	Loomio	No
	Pol.is	Yes
	ZenCity	No
Social Media Integration Tools	Hootsuite	No
	Buffer	No
Open Data Platforms	CKAN	Yes
E-Petition and Voting Platforms	Consul	Yes
	Helios Voting	Yes
Chatbot and AI Tools	Dialogflow	Yes
	OpenDialog	No
	PayIt	No
	IBM Watson Assistent	No
	Botpress	Yes
	Microsoft botframework	Yes
	Rasa	Yes

3.2.4 Building Block 04 | Data Query Service

Table 15: Summary Table of Building Block 04

Minimum Ambition Level	2 – Experimental Twins
Ki	Software
Maturity Level	Good Enough
Internal code	BB.04
Relevant MIM	MIM6

Description

Data query services provide a secure interface to access, explore, and understand the data within the Digital Twin. They allow users to ask questions and perform analyses on the data using a standard query language, typically SQL, GraphQL or SPARQL although other interfaces can be used as well,

especially when scalability is important and running arbitrary queries is not desirable (e.g., using Linked Data and Linked Data Event Streams).

By providing an abstraction over underlying data storage systems, query services ensure efficient, optimisation access to data. They can manage large volumes of data and provide responses in real-time or near-real-time, which is often a requirement for Digital Twin systems. The main elements of the Data Query Service BB can be summed up as:

- Query Languages: Depending on the complexity and capabilities of the Data Query Service, it may support a specific query language, multiple or a standard query language like SQL. This allows users to specify the criteria for data retrieval and depending on the language can provide advanced filtering, sorting and grouping of results.
- Data Sources: The Data Query Service needs to be able to connect to multiple heterogeneous data sources, which might include databases, data warehouses, data lakes or even API's or message brokers. It needs to abstract the complexity of dealing with several types of data storage systems away from the user.
- User Interface: There is typically a user-friendly interface that allows users to input their data query in a structured way. This could be in the form of an application or API.
- Result formatting: The service might allow users to specify the desired output format. Examples are JSON, CSV, XML or other structured formats.
- The Data Query Service also needs to concern itself with some non-functional requirements:
    - Scalability: The service should be scalable depending on the expected number of users and data volumes.
    - Performance: The query performance might be optimised by implementing caching mechanisms, parallel processing and query optimisation techniques.
    - Security: Often there are security measures in place to ensure that users can only access the data they are authorised for. Authentication of users/applications and implementation of mechanisms such as RBAC might be in place.

The tools that are used for data must be able to interact with the different data storage tools of the LDT and hence have dependencies with other BBs from the reference architecture.

Asset Registry

Since the Data Query Service needs to be able to find and access various kinds of data sources across the LDT system, an Asset Registry can provide the dual capabilities of data discovery to find the relevant data sources for a query, and schema management which aides in the result formatting aspect of the BB.

Data Replication Service

This BB's capabilities can ensure that the Data Query Service operates in an environment where there is a scalable, performant, and cost-effective way to query disparate data sources.

Data workflow and component orchestration

Executing a query might require a complex set of data flow actions and orchestration of tools, even the execution of full workflows, which is the capability of this BB.

Integrated Environment

Since the Data Query Service needs to interact with multiple different BBs within the LDT, it needs the Integrated Environment's capabilities to ensure correct error handling, UI and UX elements to ensure user-friendliness and accessibility best practices.

Synthetic Data Generation Tools

Synthetic Data might be necessary on-demand, which means the Data Query Service should be able to communicate with the Synthetic Data Generation Tools and have them generate relevant data based on a query's parameters.

Capabilities, functional requirements, mechanisms and interoperability guidance

Table 16: Capabilities Table of Building Block 04

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards
2 – Experimental Twins	10	Data publishing	Data	MIM6	SQL, GraphQL, SPARQL, REST, LDES, NoSQL

Table 17: Mechanism Table of Building Block 04

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1.	10	The Data Query Service must support SQL querying.	Implement an SQL query engine	SQL-92, SQL:2011 SQL:2016
2.	10	The Data Query Service should handle large datasets efficiently.	Implement query optimisation techniques and parallel processing	MapReduce (Hadoop), Apache Spark
3.	10	The Data Query Service should support various data sources.	Implement connectors for databases, data warehouses, APIs	Plugins/extensions compliant with JDBC, ODBC
4.	10	The Data Query Service should have caching mechanisms to improve query performance.	Implement result caching and data caching strategies	RFC 7234
5.	10	The Data Query Service should allow users to specify desired output formats.	Support various output formats	JSON, CSV, XML
6.	10	The Data Query Service must be scalable to handle many concurrent users.	Deploy a distributed infrastructure	Apache Spark, MapReduce
7.	10	The Data Query Service must ensure data security and confidentiality.	Encrypt communication and data storage and enforce access controls	Comply with industry-standard security practices and regulations such as ISO/IEC 27001 for information security management.

Examples

LDES has been identified as a possible new paradigm for scalable and cost-effective data publishing. As often the costs of publishing data are keeping candidate publishers from making their data

accessible, LDES – a SEMIC standard – provides as solution that addresses these issues. Based on the principle of replicating/ synchronising the sets partially or entirely at the side of the consumer, it can also be used as a basic data transfer protocol in Digital Twins.

Table 18: Examples of tools for Building Block 04

Tool	Open Source	Popularity & Community Size	Standards
Linked Data Event Streams (LDES Server)	Yes	Medium	Linked Data Formats, LDES, SPARQL
Apache Druid	Yes	High	SQL
Google BigQuery	No	High	SQL
Amazon Redshift	No	High	SQL
Microsoft Azure SQL Data Warehouse	No	High	SQL
Presto	Yes	High	SQL
Snowflake	No	High	SQL
Apache Hive	Yes	High	SQL
Apache Hbase	Yes	Medium	SQL and NoSQL
Apache Drill	Yes	Medium	SQL
TNO Request Layer	No	Low	
DUET Federated Query API	No	Low	OpenAPI
Apache Impala	Yes	High	SQL
Elasticsearch	Yes	High	N/A

Alignment with SIMPL specifications (https://ec.europa.eu/newsroom/dae/redirection/document/86727)

In developing this component, an alignment with the following building blocks from the referenced SIMPL implementation roadmap plan is needed:

- "Data Store Connector": defined as "*The data store connector foresees the integration with the internal data federation of a data provider. This connector foresees integrations with multiple popular data management solutions, such as NTFS file systems, MySQL or PostgreSQL relational databases, MongoDB key-value databases…*" and the associated simple and bulk data transfer BBs.
- "*Paas Services*": SQL databases, graph databases, etc.

3.2.5 Building Block 05 | Data Replication client

Table 19: Summary Table of Building Block 05

Minimum Ambition Level	2 – Experimental Twins
Kind	Software
Maturity Level	Good Enough
Internal code	BB.05
Relevant MIM	MIM5

Description

Maintaining an accurate and up-to-date representation of a city in a local Digital Twin is a complex task due to the dynamic nature of urban environments. Given this context, it is paramount for Digital Twins to adopt mechanisms that can adequately capture these evolving cityscapes.

Capture the evolving changes from different asynchronous sources can be a complex problem, especially as a Digital Twin will continuously produce results based on these changing data resources. These can be output of models, IoT platforms, physical environment changes, and in the context of MIM5 to deliver transparency, rollback in time will sometimes be needed, making links between data sets also persistent.

To be able to address this problem, a data replication strategy is instrumental for the success of a local Digital Twin, hence this BB to interface with such a system from the context of an LDT. There can be different strategies to deploy data replication in streaming and batch mode. A crucial aspect is to maintain versions of key data objects, as it is crucial for data lineage and data archiving.

There are also several ways to implement data replication in the system of systems. It can be done federated, where the data source systems keep track of versions and offer access to previous versions, or it can be replicated within the LDT itself. Another way is to use the concept of event sourcing, where the data sources are publishing their data changes as events which are stored in a strictly ordered, immutable manner. So instead of storing the current state a series of events that describe the changes over time were stored. Keeping track on the history of changes can then be a direct negotiation with the data source.

Capturing metadata about the data replication process itself is key, and having a standard way to do this is needed across the data suppliers and data users, with the LDT acting as a replication service. Earlier, the concept of event sourcing was discussed, which can deliver key features for an LDT such as immutable history and audit trailing (think about the capabilities needed by MIM5), temporal querying and resilience. Using events as the basic BBs for data replication can be a good strategy for an LDT, but also brings its own challenges. Depending on the nature and complexity of the data (volume, velocity, variety, etc.) a hybrid approach (federated, central, even sourcing) could be needed.

Thus, this BB has relations to the direct data production and consumption interface BBs, and the data storage and archiving BBs, which will provide the basic mechanisms to implement data replication. It can function as an interface to these BBs and to functions that need to track history (auditing, rollbacks, etc.). Moreover, it will also have a close dependency with the event-based publishing service.

Capabilities, functional requirements, mechanisms and interoperability guidance

The data replication capability is needed from ambition level 2, but its requirements can scale further when entering the higher ambition levels, as the introduction of predictive analysis will increase the number of data sources used adding near-real-time data flows and validated models. Predictive analysis uses historic data and patterns and will heavily depend on the features and maturity of the data replication strategy. So, this is a clear indication that the data replication strategy and client need to scale with the ambition level.

Table 20: Capabilities Table of Building Block 05

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards
2 – Experimental Twins	5	Data Replication	Data	MIM5	LDES, SPARQL, JSON, AVRO, MQTT, etc.

The following table lists the main requirements for a data replication system that can respond capture data changes and replicate state. Remark that the functionalities for data storage and archiving are the responsibility of the data storage BB. The following list is an indication of the basic functional requirements to be able to track data entities (mutable or immutable), their associated change event streams and some indications on which mechanisms can be used.

Table 21: Mechanism Table of Building Block 05

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1.	5	The data replication client shall be able to subscribe to and receive event streams.	Event-driven architecture with pub/sub	MQTT, AMQP, Kafka, LDES
2.	5	The client shall be able to manage different event formats.	Support for commonly used serialisation formats	JSON, LDES, Protobuf, Avro
3.	5	The client should maintain an entity registry to keep track of all entities and their last known state.	Entity registry with up-to-date snapshots	In-memory or persistent databases
4.	5	The client should support the concept of mutable and immutable entities.	Mutable/immutable entity types	Design patters for Immutable and mutable objects
5.	5	The client should be able to reconstruct the state of an entity from its event history.	Event sourcing and snapshotting	Event sourcing, CQRS
6.	5	The client should be able to manage relations between entities and keep track of them in event reconstruction.	Event processing considers and respects defined entity relations	Event-driven architectures
7.	5	The client should be fault-tolerant and able to recover from failures.	Backup/checkpoint system and use of the storage BB	Use of SOTA technologies

8.	5	The client should support security (access and transmission).	Secure communication and access protocols	OAUTH2.0, HTTPS, SSL/TLS
9.	5	The client should be able to transform event data on the fly as needed.	Event stream transformation capabilities	Apache technology, Kafka streams
10.	5	The client should be able to filter event streams based on certain criteria.	Event stream filtering capabilities	Apache technology
11.	5	The client should support monitoring and alerting for data streams and entities.	Integration with monitoring and alerting tools	Prometheus, Grafana, ELK

Examples

There are lots of tools that can be used to compose this BB. This report indicates that a hybrid approach (depending on the ambition level) will be needed addressing the complexity of a Digital Twin.

When observing key enablers to keep track of history and relations, attention needs to be paid to immutability, the relations between event streams, and their entities and searchability. One way to do this is by employing the concept of LDES (SEMIC standard). In the LDES model, every change in the urban environment can be represented as an event. These events are stored in a strictly ordered, immutable manner, which allows for a complete and reliable recording of the city's changes over time. Moreover, LDES is based on Linked Data and can thus have built-in relations to other Linked Data objects, vocabularies and schemes, which is a powerful asset to conquer the complexity of data replication. LDES can be seen as an open standard for scalable and cost/maintenance effective data publication.

The following table lists some tools that can be used **to compose** a data replication client in the context of LDTs, covering some of the requirements.

Table 22: Examples of tools for Building Block 05

Name	Description	Open Source	Popularity & Community	Features	Standards Adherence
LDES client	LDES is a model for representing data changes as a series of ordered, immutable events. LDES is built upon the principles of Linked Data and provides an efficient method for tracking and publishing data that changes over time.	Yes	New and growing community with several adopters in the academic and public sector.	Data Changes as Events, Immutable Events, Linked Data, Accessible over HTTP, Page-based Access, Scalability	Complies with Linked Data principles, LDES, Open Standards
Apache Kafka Client	Kafka is a distributed streaming platform that allows publishing, subscribing, storing, and processing of streams of records in	Yes	Extremely popular, with a large active community.	High-throughput, Real-time, Fault-tolerant, Durable, Scalable	Supports different message formats like Avro, JSON, XML etc.

	real-time. It is commonly used for building real-time data pipelines and streaming apps.				
RabbitMQ Client	RabbitMQ is a robust messaging service for applications. It supports multiple messaging protocols and can be deployed in distributed and federated configurations.	Yes	Popular with a large active community.	Distributed Deployment, Federation, Multi-protocol, High Availability, Load Balancing	Supports AMQP, MQTT, STOMP, etc.
Google Pub/Sub Client	Google Pub/Sub is a scalable, dependable, and real-time messaging service that allows sending and receiving messages between independent applications.	No (Google Service)	Popular in cloud-native applications with a large community.	Real-time, Durable, Scalable, Integrated with Google Cloud	Supports different message formats
AWS Kinesis Client	AWS Kinesis is a platform for streaming data on AWS, offering powerful services to make it easy to load and analyse streaming data, and also providing the ability for you to build custom streaming data applications for specialised needs.	No (Amazon Service)	Popular among AWS users with a large community.	Real-time, Durable, Scalable, Integrated with AWS	Supports different message formats
Microsoft Azure Event Hub	highly scalable data streaming platform and event ingestion service, capable of receiving, processing, and storing millions of events per second	No (Microsoft Service)	Popular among Azure users with a large community.	Real-time, Durable, Scalable, Integrated with Azure	Supports different message formats

Alignment with SIMPL specifications (https://ec.europa.eu/newsroom/dae/redirection/document/86727)

In developing this component, an alignment with the following BBs from the referenced SIMPL implementation roadmap plan is needed:

- "*Data governance*": the data replication client can support the data lineage & data profiling capabilities identified by thus BB.

3.2.6 Building Block 06 | Data Workflow and component Orchestration

Table 23: Summary Table of Building Block 06

Minimum Ambition Level	1 – Awareness Twins
Kind	Software
Maturity Level	Plug & Play
Internal code	BB.06
Relevant MIM	MIM5

Description

A BB that provides Data Flow Orchestration and Control Flow orchestrations facilitates the ability to execute multiple separate but interrelated steps in a data and control flow in the right order. This BB takes the raw data and facilitates the data processing into valuable insights or actionable information. These workflows can be simple or complex depending on the requirements of the specific use case in place for your LDT and the amount of data and models involved. Data workflows are commonly needed to support in data (processing) pipelines, data integration, data analysis and data visualisation applications and thus relevant for all types of LDTs (starting from 1 – awareness twins). Control flow orchestration constitutes components that facilitate the coordination and management of tasks to execute multiple separate, but interrelated, steps in a workflow in the right order to support model interactions given the proper dependencies (which is needed for intelligent or Predictive LDTs). This BB is part of the core an LDT and plays a key role in ensuring the support between data, models and visualisations. It therefore has strong links with the following BBs in an LDT:

Interaction service

Based on the interaction service, a user induces changes that can trigger the orchestration of data inputs/outputs and intensities of models either predicting or simulating new scenarios.

Data query service

A type of data source may be requested and integrated from another system to be integrated through a query. Once induced this a task to consider when orchestrating a data workflow. In addition, a data query service may induce a task that follows the opposite route, which will then rely on a complex orchestration of data flows and control flow steps.

Message Broker

Sharing messages between BBs enables data exchange between simulation models and other LDT system BBs. The message broker contains the information to stream messages were based on events multiple distributed processes should interact with each other. This sequentially has a link with the data workflow and component orchestration given the relationship between models, data and visualisation.

Model abstraction service

Given that multiple models in an LDT interact, the data workflow and component orchestration have a strong link with the model abstraction service to serve a sequential step of task to e.g., feed models with the right data.

One of the main requirements to the data workflow and component orchestration is to limit the number of bottlenecks with regards to performance when striving for ambition levels of predictive and intelligent twins. As those ambition levels consists of numerous complex orchestration activities that need to be well coordinated while maintaining prominent levels of performance to execute tasks in a responsive manner e.g., for simulation and predictive scenarios within an LDT. This is for example is the case when multiple data sources and models are being consulted for complex simulations and predictions (e.g., calculation of traffic intensity or changes in the traffic system on mobility flow, air and noise pollution).

The components of a data workflow are:

- Data ingestion: The stage that brings in raw data into an LDT system from various sources, such as databases, APIs, files or data streams.
- Data transformation: The stage where the ingested data is cleaned, filtered, aggregated and enriched if needed to make it suitable for the data to be analysed or stored.
- Data processing: The stage of the actual computation or performance of analysis on the transformed data in order to make meaningful insights.
- Data storage: The stage where data is stored in data bases, data warehouses or data lakes for future usage. Preferably, when achieving higher levels of data should be stored locally and should require access to a central database to support higher performance of LDT.
- Data visualisation and/or reporting: The (final) stage where the data is presented in a visual manner for an end-user to understand and interact with the data.

The key components of a Component Orchestration are:

- Task scheduling: The stage in which the order of the tasks should be executed based on certain dependencies and availability of resources.
- Dependency management: The stage to define the relationships between tasks to determine the order of the task as such a task will not start until all dependencies are successfully completed.
- Parallel execution: The stage to optimise the execution of tasks by running them in parallel (concurrently) when possible, especially when there are independent tasks to be executed.
- Error handling: The handling of exceptions and errors that might occur when executing tasks such as inconsistencies in the data, network issues or failures between soft- and hardware components.
- Monitoring and Logging: Keeping track of all the tasks executions, its progress and logging tasks to act upon for troubleshooting or auditing.

Capabilities, functional requirements, mechanisms and interoperability guidance

Table 24: Capabilities Table of Building Block 06

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards
2 – Experimental Twins	13	Data Processing	Data		JSON, XML, SQL, Binary serialisation with protocol buffers
	30	Data flows and component orchestration	Integration		

3 – Predictive Twins	6	Data transformation	Data		JSON, XML, SQL
4 – Intelligent Twins	47	Technical transparency & explainability	Governance & Compliance	MIM5	N/A

Table 25: Mechanisms Table of Building Block 06

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1.	30	The solution should support task scheduling capabilities to be able to execute tasks whilst also considering dependencies and delays between tasks.	Unified APIs and standard data formats	JSON, XML
2.	30	Dependencies should be managed ensuring a task will not start until dependencies have successfully been completed.	(Automated) task scheduling mechanism	
3.	30	The solution should support parallel execution of tasks to optimise resources and scalability.	Parallel execution and task scheduling	N/A
4.	30	The solution should allow integration with various data sources, services and platforms as well as support the creation of custom tasks and connectors.	Unified APIs and standard formats. Platform agnostic tools to develop environments	JSON, XML, REST APIs, SQL, Binary serialisation with protocol buffers
5.	30	The solution should be able to manage errors, notify about these errors and automate recovery of failures.	Alerts and notifications.	Payload formats such as JSON and XML, or protocols (SMTP, HTTP/HTTPS, MQTT, TCP/IP)
6.	6	The solution should support data input/output operations and allow seamless integration with various data stores and formats.	APIs	JSON, XML, REST APIs, SQL, Binary serialisation with protocol buffers
7.	30	The solution should provide monitoring and logging capabilities to track executions, identify bottlenecks and troubleshoot issues.	Version control and logs that provide a clear history of updates and improvements	Log4j, Log4Net, Logback, JSON logging OpenMetrics, Prometheus, ELK stack
8.	30	The solution should provide a user-friendly interface to visualise workflows, track workflow performances, statuses and generate reports.	Implement a clear and intuitive interface	N/A
9.	6	The solution should support multiple programming languages and of data formats.	Unified API's and platform-agnostic tools to develop environments	Java, Python, Javascript/Node.js

10.	13	The solution should be able to integrate seamlessly with other data, tools, environments and services used commonly in data engineering, data analysis workflows but also to workflow related to different predictive and simulation models involved.	Unified API's and platform-agnostic tools to develop environments	
11.	13	The solution should be able to scale up effortlessly to manage large scale data workflows and support distributed tasks for example when different models are being consulted for a use case.	Platform agnostic mechanisms to support distributed processing platforms, containerisation, auto-scaling, caching, data partitioning, streaming processing	Binary serialisation with protocol buffers
12.	47	The solution should have good documentation, support and resources for users that can be accessed in a user-friendly manner.	Standardised documentation formats with meaningful definitions and descriptions	Markdown, reStructuredText, HTML
13.	6	The solution should allow to process of cleaning, formatting, filtering, aggregation, enrichment, joining/merging, converting, cleaning, normalisation, encoding and reshaping data from one data format or structure into another to make it suitable for analysis.	Unified APIs and support of data schema's	API based transformation, RESTful API design, JSON, XML, Binary serialisation with Protocol Buffers

Examples

Table 26: Examples of tools for Building Block 06

Name	Description	Open Source	Popularity & Community	Features	Standards Adherence
Apache airflow	Open-source platform used for data workflow and component orchestration.	Yes	High	Task scheduling, parallelism, extensibility, monitoring, scalability	Apache license 2.0, Supports different data standards or formats
Flyte	Data workflow and orchestration platform which provides support for data pipeline workflows and machine learning workflows.	Yes	Medium	Data ingestion, transformation, flow control and monitoring and logging	Supports different data standards or formats

Apache NiFi	Solution for data workflow management to automate flow of data between systems and helps design data pipelines for e.g., ML/AI algorithms.	Yes	Medium	Parallelism, GPU, monitoring	Apache License 2.0, Supports different data standards or formats e.g. JSON or YAML.
Apache Beam	Open-source data processing model and framework that supports both batch and stream processing of data.	Yes	High	Execute against multiple execution engines (portability)	Data source and programming language agnostic
Proprietary workflow solutions offered by cloud providers	Examples are AWS step functions that provides a fully managed service.	No	High	RESTful API, parallel execution	RESTful API design principles

Alignment with SIMPL specifications
(https://ec.europa.eu/newsroom/dae/redirection/document/86727)

Before final design and implementation of this component, an alignment with the following building blocks from the referenced SIMPL implementation roadmap plan is needed:

- "*Data orchestration*": The main building block for taking siloed data from multiple data storage locations, combining them, and making them available to data consumer applications.
- "*Container provisioning*": Allows the provisioning & deployment of container (images) in order to launch and stop the execution of container-based algorithms, generic applications or custom code. The containers may be preferred over the VMs for the end users in case the application portability is the most important aspect for them.
- "*Supporting – infrastructure orchestration*".
- "*Supporting – Distributed execution*".

3.2.7 Building Block 07 | Extended Reality (AR/VR services)

Table 27: Summary Table of Building Block 07

Minimum Ambition Level	4 – Intelligent Twin
Kind	Software
Maturity Level	No Existing Standard
Internal code	BB.07
Relevant MIM	MIM4, MIM5, MIM6, MIM7

Description

XR is an umbrella term for various immersive technologies, including VR, AR and Mixed Reality (MR). In all cases, XR stands for an immersive experience for a user, enabling to interact with virtual elements in a (semi-)virtual environment and sometimes with real elements simultaneously. The relationship between LDT and XR lies in its complementary roles when immersed in data-driven representations of real-world environments.

Potential benefits of integrating XR technologies in LDTs to facilitate interactivity and engagement between different stakeholders (e.g., urban planners, decision-makers, citizens and communities). It can provide a deeper understanding of the urban environment potentially serve better engagement, foster collaboration and inclusivity by actively participating in potential future developments. This can be supported by interactive exploration through data layers, making (real-time) changes to virtual city's elements and simulate various scenarios to optimise city's designs.

The main technologies of XR can be categorised as follows:

- **Virtual Reality (VR)**: technology that immerses users in a fully virtual environment, separate from the real world where they can interact and explore as if they were physically present. It uses a combination of hardware and software to transport users into a fully artificial 3D environment. Usually accomplished by wearing a headset although there are technologies that accomplish this without the user physically wearing any devices, for example in a VR cave. Since the virtual world has no physical limits, users can be transported across time and great distances while remaining stationary in the real world, enabling unique viewpoints that could not be achieved through any other means. VR is different from a 360-virtual display because the viewpoint changes based on the user's perspective. In some systems, the user's physical movements in the real world can be translated to virtual movement as well. This requires some kind of tracking technology, either solely for the headset or even of the user's entire body. The relationship between LDTs and VR lies in the possibilities to use VR as a visualisation and interaction layer for LDT data.
- **Augmented Reality (AR)**: overlays virtual elements onto the real world, enhancing the user's perception of reality. AR can be enabled by common devices such as smartphones and tablets or more exotic devices such as AR glasses. The key difference with VR is that in AR the real world is never fully obscured, thus it never fully immerses a user into a different perception of reality. However, it gains the capability to enhance real world physical objects, by overlaying extra information or even different visualisations on the real-world object. Therefore, it produces a more enhanced experience compared to the fully immersive with VR.
    - **Mixed Reality (MR)**: is an advanced form of AR, which seamlessly blends real-world elements and virtual elements. In MR, the user can interact with both simultaneously, creating a more immersive and interactive experience. This is an interesting technology for LDT's because it can better leverage the LDT's data to identify real-world elements, overlay extra information onto them and identify which user interactions are possible with the element through the LDT. The devices through which MR is possible are similar to AR but it requires greater capabilities from the devices themselves, such as high-definition cameras and faster processors to enable on-device object recognition software to run.

In relationship to other components of the LDT reference architecture, XR services can be used for multiple use cases where visualisations and interaction with users is valuable in the decision-making processes and are such based upon all LDT core BBs. Related BBs are listed below.

Integrated Environment

The XR as part of an LDT is only valid for decision-making purposes when the applications give a seamless user experience that is integrated with the governed and managed data services within the

LDT core. The integrated environment also refers to the advanced geospatial services and other visualisation services.

Collaboration & community management system, Case & scenario management

XR services are very much linked with the BB on Collaboration & Community management to facilitate city engagement activities and also can support Case & Scenario feedback for the Case & Scenario management system.

Capabilities, functional requirements, mechanisms and interoperability guidance

Table 28: Capabilities Table of Building Block 07

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards
2 – Experimental Twins	23	Advanced Visualisation & Geo dashboarding	Visualisation	MIM7	OGC, CityGML
4 – Intelligent Twins	25	Augmented Reality (AR)	Visualisation	MIM4, 5, 6	glTF, OpenXR, MPEG-V, ARML
	24	Virtual Reality (AR)	Visualisation	MIM4, 5, 6	glTF, OpenXR, MPEG-V, ARML

Table 29: Mechanism Table of Building Block 07

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1.	24	The XR should support seamless interaction from users.	Use of gesture recognition and voice commands	WebXR, OpenXR
2.	24	The XR should provide accurate real-time tracking of a user's position and orientation.	Use of positional tracking, skeletal tracking and object recognition	WebXR, OpenXR
3.	24	The XR should minimise latency to ensure an immersive experience.	Use of optimised hardware for rendering of the virtual environments	N/A
4.	24	The XR should prioritise user safety.	Environment scanning and collision detection to warn users	WebXR, OpenXR
5.	24, 25	The XR should support cross-platform development and use.	Unified API's and platform-agnostic tools to develop environments	WebXR, OpenXR, glTF, MPEG-V
6.	24	The XR should provide haptic feedback.	Use of vibrations and force feedback	OpenXR, Khronos Haptic API
7.	24, 25	The XR should support multi-user collaboration.	Network synchronisation and spatial mapping for the	OpenXR, WebXR

			same virtual environments	
8.	24, 25	The XR should prioritise user privacy and data protection.	Use of data encryption and explicit user consent on what data is collected, stored and processed	GDPR
9.	25	The XR should support object recognition.	Use of computer vision algorithms	OpenXR
10.	25	The XR should enable Simultaneous Localisation and Mapping.	Use of sensor fusion and map generation to map the environment and the user's position in it	OpenXR
11.	25	The XR should associate LDT data with virtual or real-life elements.	Use of spatial anchors and interoperable data models suited for XR	ARML (OGC)

Examples

First of all, DUET describes important illustrations of interactions with the platform from different persona like DUET users, citizens, policy advisors... linking this to e.g. gamification possibilities in its deliverable D4.5, see section 3.3 "Reporting Tools" on target types, duet dashboards, storytelling (e.g. using AR/VR technology) and gamification options. https://www.digitalurbantwins.com/_files/ugd/725ca8_5d77c1e261794216b5d141f308e9b9cb.pdf.

Table 30: Examples of tools for Building Block 07

Name	Description	Open Source	Popularity & Community	Features	Standards Adherence
Meta Quest	Meta Quest is the developer platform for Meta's (parent company of Facebook) XR devices.	No	One of the most extensive VR developer platforms in the world, considering user base	Voice commands, partial anchors, collision detection, haptic feedback API	Supports OpenXR
Unity	Provides extensive, cross-platform support for building AR, VR and MR applications for various platforms such as Oculus, HTC Vive, Microsoft HoloLens and mobile devices.	No	One of the most widely used game engines in the world	Cross-platform support, room-scale VR, physics engine, interaction SDK, spatial audio	Supports OpenXR, WebXR, glTF and most other standards through plugins
ARKit	Apple's AR development framework for iOS	No	Most popular framework for iOS developers	World tracking, facial tracking, image tracking,	N/A

	devices. It allows the creation of AR experiences for iPhones, iPads and Vision Pro headsets.			object recognition, light estimation, 3D audio	
ARCore	Google's AR development framework for Android devices. It allows creation of AR mobile apps for all devices that run the Android operation system.	Yes (commercial license)	Large user base	Motion tracking, environmental understanding, face tracking, depth API, cloud anchors, point clouds, vertical plane detection	Supports WebXR for web apps
AR.js	An open-source Javascript library that allows developers to create AR experiences for the Web	Yes	Popular in cloud-native applications with a large community.	Marker-based and Location-based AR features	Supports WebXR
OpenGL	Open-source graphics rendering AP that provides a set of functions for creating interactive 2D and 3D graphics applications	Yes	Widely adopted, with good documentation	Library to support development of 2D and 3D graphic applications. lows	API, support of various wrappers (like GLFW, SDL and Qt)
Direct3D	API developed by Microsoft as part of its DirectX technology suite	No	Support in Microsoft ecosystem	Provision of high-level libraries to simplify certain aspects like rendering of images and providing utility functions	Support in Windows environment, Open GL interoperability
Dataset: EU-wide 3D cities reference Open Street Maps based repository	Collaborative Dataset for developing an Open repository of buildings information for all EU cities	Yes	Open Street Maps under ODbL license	Provision of an EU wide reference for 3D applications and building information	CityGML

3.2.8 Building Block 08 | Federation Learning

Table 31: Summary Table of Building Block 08

Minimum Ambition Level	3 – Predictive Twins
Kind	Software
Maturity Level	No Existing Standard
Internal code	BB.08
Relevant MIM	MIM4, MIM5, MIM6

Description

Federated Learning is an advanced machine learning paradigm where a shared global model is trained across multiple decentralised edge devices or servers holding local data samples, without explicit exchange of the data (i.e., the model is brought to the data). This approach stands in contrast to traditional centralised machine learning, where all the training data is required to be present in one centralised location.

In the context of LDTs, this method can be particularly beneficial as it allows for training on a vast array of locally gathered data, which might pertain to various parts of the urban environment, without the need to transfer this data to a central location. This can enhance privacy and security while maintaining the benefits of collaborative learning.

Federated Learning services would typically include:

1. A server component that manages the overall process, including initiating the learning process, distributing the shared model, and aggregating updates.
2. Client components running on local devices/servers, which process the local data, train the model, and send updates back to the server.
3. Communication protocols for secure and efficient transmission of model updates.

Implementing a Federated Learning service within an LDT system would thus require careful selection and orchestration of tools and frameworks that support federated learning, as well as consideration of the appropriate networking and security infrastructure. Note that Federated Learning is a rapidly advancing field, with a variety of tools and frameworks (both open source and commercial) available. It is crucial to choose the tools that best suit your specific use case and data context.

Federated learning can be achieved using different libraries and tools (see below). There are multiple valid approaches towards supporting federated learning. One such approach is to abstract federated learning behind a model abstraction service that handles the orchestration and consider the federation service endpoints as one of the inputs of the orchestrator pseudo-model.

Capabilities, functional requirements, mechanisms and interoperability guidance

Table 32: Capabilities Table of Building Block 08

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards
3 – Predictive Twins	20	Federated Learning & Training	Analytics	MIM4, 5, 6	Evolving practices (i.e. AI act)
	21	Machine Learning & AI	Analytics	MIM4, 5, 6	

Table 33: Mechanism Table of Building Block 08

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1.	20, 21	The solution should address data privacy and security to prevent unauthorised access and data breaches.	Encryption, data anonymisation, differential privacy techniques	GDPR
2.	20, 21	The solution should support different machine learning libraries and models.	Federated Learning Frameworks	N/A
3.	20, 21	The solution should be able to handle various data formats and schemas from different sources.	Accommodate diverse data representations	JSON, XML, Binary serialisation
4.	20, 21	The solution should be able to process data closer to its source while reducing data transmission and latency.	Edge computing capabilities	APIs, Communication protocols (MQTT, HTTP/HTTPS., TCP/IP)
5.	20, 21	Make use of efficient communication protocols in order to minimise overhead of model updates.	Compressed model updates and selective aggregation	APIs, Communication protocols (MQTT, HTTP/HTTPS, TCP/IP.)
6.	20, 21	The solution should optimise resource allocation and model training to manage resource constraints efficiently.	Task scheduling, dynamic resource allocation	N/A
7.	20, 21	The solution should be able to address data versioning and synchronisation to handle data distributions and updates.	API versioning, Standards formats and Schemas, event-driven synchronisation, distributed data storage	JSON, XML, MQTT, WebSockets, AMQP
8.	20, 21	The Federated Learning service should be able to be work with/integrate with LDTs.	Reference architecture and design, Interoperability mechanisms	N/A

9.	20, 21	The solution should include techniques to explain decisions made by federated learning models.	Model usage guidelines	N/A
10.	20, 21	The solution should have scalability in mind, to which it can consider the federated learning system can scale to accommodate an increasing number of participants and data sources.	Model exchange formats, standardised communication protocols	ONNX, Tensorflow SavedModel
11.	20, 21	The solution should be able to deploy models and monitor to track model performance.	Performance monitoring and analytics	N/A
12.	20, 21	The solution should enable immediate decision support when Federated Learning models quickly adapts to changing events or environments.	Real-time monitoring and decision support	N/A
13.	20, 21	The solution should support collaborative governance processes to ensure fairness and accountability among stakeholders participating in the federated learning process.	Model usage guidelines	N/A

Examples

There do not seem to be any Federated Learning service components that fulfil all the requirements and can off-the-shelf be integrated into an LDT architecture. But there are several tools that are addressing some of the requirements and could be used in the composition of this BB.

Table 34: Examples of tools for Building Block 08

Tool	Open Source	Popularity & Community Size	Standards
TensorFlow Federated (TFF)	Yes	Large (part of TensorFlow community)	Evolving practices
PySyft	Yes	Medium	Evolving practices
FATE (Federated AI Technology Enabler)	Yes	Medium	Evolving practices
PaddleFL	Yes	Medium (part of PaddlePaddle community)	Evolving practices
IBM Federated Learning	Yes	Medium	Evolving practices
NVIDIA Clara FL	No	Small to Medium (specific to healthcare sector)	Evolving practices
Flower	Yes	High (to set up a federated learning framework)	Evolving practice

Alignment with SIMPL specifications
(https://ec.europa.eu/newsroom/dae/redirection/document/86727)

Before final design and implementation of this component, an alignment with the following building blocks from the referenced SIMPL implementation roadmap plan is needed:

- "*Container provisioning*": Allows the provisioning & deployment of container (images) in order to launch and stop the execution of container-based algorithms, generic applications or custom code. The containers may be preferred over the VMs for the end users in case the application portability is the most important aspect for them.

3.2.9 Building Block 09 | Integrated Environment

Table 35: Summary Table of Building Block 09

Minimum Ambition Level	2 – Experimental Twins
Kind	Software/ Guidelines
Maturity Level	No Existing Standard
Internal code	BB.09
Relevant MIM	MIM6

Description

The primary goal of an Integrated Environment is to create a unified and consistent user experience that enhances usability, efficiency, and accessibility. This is achieved through means of software integration, semantic interoperability and design best practice guidelines.

Key aspects of an Integrated Environment include:

- **Consistency**: The user interfaces and interaction patterns remain consistent throughout the different tools, promoting familiarity, recall and reducing the learning curve for users. Consistency ensures that users can easily navigate different sections of the software without confusion.
- **Smooth Workflows**: It ensures that workflows are well-defined and logically connected. Users can perform tasks or actions in a natural, intuitive, and uninterrupted manner, avoiding unnecessary context switches.
- **Data Sharing and Synchronisation**: Integrated Environments often involve seamless sharing and synchronisation of data across different tools or BBs. When users interact with various parts of the system, relevant data is shared and updated in real-time, maintaining data integrity and coherence.
- **Multi-Platform Support**: An integrated environment can extend across various platforms and devices, such as desktop computers, mobile phones, tablets or XR devices. The interface adapts to the specific platform while retaining consistency and usability.
- **Personalisation:** While maintaining consistency, integrated user experience can also offer personalisation options to cater to individual user preferences and requirements. This could include customisable layouts, themes, or preferences.

- **Accessibility:** The Integrated Environment is designed and developed in a way that makes it usable and inclusive for all users, including those with disabilities or impairments. Ensuring accessibility is essential to provide equal access and opportunities for all users to interact with and benefit from the system.
- **Collaboration and Communication**: It ensures smooth communication and collaboration between multiple users, enabling them to work together seamlessly and efficiently.
- **Error Handling**: An Integrated Environment considers error handling and recovery mechanisms. When errors occur within any BB of the system, they are managed gracefully and guide the user toward resolving the issue without disrupting the overall experience. This requires an interoperable error handling mechanism of all BBs.
- **Performance**: Performance optimisation is crucial in an Integrated Environment to ensure smooth and responsive interactions, avoiding delays and lags that might lead to frustration.

The functional dependencies of an integrated environment are widely spread over all use cases that apply to the LDT and to all visualisation components of which also the collaboration & community management system tools. On the technical side as well, the integrated environment has dependencies with specific BBs from the reference architecture.

Message Broker

Overall, an Integrated Environment aims to create a holistic and enjoyable experience for users, enhancing user satisfaction, productivity, and engagement with an LDT. The BB has dependencies on the Message Broker BB for ensuring interoperable communication between BBs.

Capabilities, functional requirements, mechanisms and interoperability guidance

Table 36: Capabilities Table of Building Block 09

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards
2 – Experimental Twins	26	Integrated User Experience	Visualisation	MIM6	WAI-ARIA, WCAG 2.0

Table 37: Mechanism Table of Building Block 09

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1.	26	The Integrate Environment should provide a consistent UI design.	Define and adhere to a standardised UI style guide	Platform specific design guidelines and LDT Toolbox design guidelines
2.	26	The Integrated Environment should support seamless Navigation.	Implement a clear and intuitive navigation system	UX design best practices
3.	26	The Integrated Environment should provide cross-platform support.	Use responsive design for different devices	Industry standard UI technologies like HTML5 and CSS3
4.	26	The Integrated Environment should prioritise accessibility.	Incorporate ARIA roles and accessible components	WAI-ARIA, WCAG 2.0

5.	26	The Integrated Environment should ensure data synchronisation.	Integrate with data synchronising mechanisms	REST API's, LDES
6.	26	The Integrated Environment should provide personalisation.	Provide user preferences and customisable settings	Allow users to save common preferences in the cloud
7.	26	The Integrated Environment should support collaboration features.	Enable real-time collaboration and communication between BBs and tools	Websockets
8.	26	The Integrated Environment should have effective error handling.	Display informative, contextual and actionable error messages	Follow UX design best practices for error reporting, OpenTelemetry
9.	26	The Integrated Environment should have multi-language support.	Implement internationalisation and localisation	I18n and L10n best practices
10.	26	The Integrated Environment should gather feedback and do user Testing.	A/B testing, monitoring	OpenMetrics
11.	26	The Integrated Environment should integrate with external systems and APIs of BBs.	Establish seamless integration with third-party services that implement interoperability standards	REST API connectors
12.	26	The Integrated Environment should provide contextual help and documentation.	Provide on-screen contextual documentation	Chatbots, input validation best practices
13.	26	The Integrated Environment should offer offline support.	Offer offline capabilities where possible and synchronise data on restored connection	Service workers
14.	26	The Integrated Environment should offer accessible documentation of its API specifications.	A tool that documents API implementations and lets users test them	OpenAPI

Examples

Table 38: Examples of tools for Building Block 09

Name	Description	Open Source	Popularity & Community	Features	Standards Adherence
WAI-ARIA	World Wide Web (W3C) recommended technical specification for accessibility best practices.	Yes	High	Guidelines for accessibility of web pages, dynamic content and UI components developed with web technologies	W3C
Prometheus	Monitoring and alerting toolkit for collecting, querying and alerting on time-series data from various systems and applications.	Yes	High	Data collection, a data model, query language (PromQL), data storage, visualisation and Grafana integration, alerting, scalability	OpenTelemetry, OpenMetrics
Jaeger	Distributed tracing system for monitoring and troubleshooting complex systems.	Yes	High	Telemetry, data storage, query service, a web-based UI for browsing data, collector for tracing data	OpenTelemetry
Google Material Design	A design language and system developed by Google. It is a comprehensive set of guidelines, principles, and best practices aimed at creating a consistent user interface across different platforms and devices. Material Design is used to design web applications, mobile apps, and other digital products.	Yes	High	Icons, components, motion and animations guidelines, accessibility, depth and elevation, layout grid, responsive design	Design guidelines
Swagger	A framework that enables developers to describe, document and visualise REST APIs through the OpenAPI standard. Give information about request and response format, authentication methods, etc.	Yes	High	YAML or JSON-based language to describe structure of API's, UI for documentation and testing API's, code generation	OpenAPI

Alignment with SIMPL specifications
(https://ec.europa.eu/newsroom/dae/redirection/document/86727)

In developing this component, an alignment with the following BBs from the referenced SIMPL implementation roadmap plan is needed:

- "*Support*" : support page, ticketing system, helpdesk
- "*Reporting*": platform usage, log info extraction
- "*Access control & trust*"

3.2.10 Building Block 10 | Interaction Service

Table 39: Summary Table of Building Block 10

Minimum Ambition Level	2 - Experimental Twins
Kind	Software
Maturity Level	No Existing Standard
Internal code	BB.10
Relevant MIM	MIM6

Description

The **interaction service**[26] is an important BB in an LDT environment as it supports the interaction of data, models and visualisations through a client interface. The interaction service keeps track of changes made in an LDT in the hypothetical context of a scenario – either by a user via an LDT client making changes in the virtual representation of a city such as closing roads, adding sound barriers, implementing circulation plans or by models that recalculate new intensities given an induced change such as new traffic intensities, updated air quality values, different water levels in sewage systems – and thus allows to conduct scenario-based analysis with LDTs.

The interaction service[27] does three things:

1) It keeps track of all the changes made so that a new scenario can be reconstructed based on the baseline scenario at any time by any component, thus remaining that the same 'state' of the data is respected throughout a e.g., a scenario.
2) All the changes are published on the message broker of an LDT core system so that other BB can detect changes and act accordingly.
3) It exposes an API to fetch or retrieve changes made after some point in time so that changes can be processed in a batch.

[26] DUET_D3.9 Digital Twin data broker specification and tools v2.0_v1.0 (digitalurbantwins.com)https://www.digitalurbantwins.com/technical-approach

[27] For testing and comparing several scenarios, an interaction service BB is not always required. Different scenarios could also be represented by using different input data which act as input for the models. The interaction service allows for more interactive interactions with the virtual world in a Digital Twin, and therefore offers more possibilities if you wish to conduct multiple scenarios and experiments in a short amount of time.

As such, the interaction service has a strong link with some BBs:

Data replication service

The link with the data replication service lies in the fact its ability to call upon a certain state of data to fulfil a scenario. As well as the output of a scenario and its accompanying identifiers to captured and used later for purposes such as replication of experiments or transparency in decision-making processes.

Message broker

The message broker is an important BB in relation to the interaction service as it serves an important function to publish the context information of the changes where other modules can act upon once detected.

Data workflow and component orchestration

The order and sequence of the task determines how the interaction services can either call upon the necessary data and use resources to complete its tasks.

How the interaction service works

To register changes in an LDT, the interaction service BB exposes an API that accepts changes in a certain format. An example for such a format is that changes are registered into a database with a scenario identifier. The interaction service provides endpoints for retrieving these changes per Scenario ID and starting from any given point in time. This allows users to reconstruct hypothetical situations in the virtual world corresponding to the scenario and run experiments with it. This is important to remain a certain state in which models reacting to induced changes are responding to the right state of the data.

The baseline scenario represents the original state that is not updated by changes. There should be a possibility to revert to the baseline by cleaning up all changes through the interaction service if this is activated via the client.

The interaction service BB has a strong link with other BBs in an LDT system. Aside from keeping the changes in store for later use, the interaction service also connects to the Message Broker to publish changes, adding contextual information (i.e., metadata) such as the scenario ID. The interaction service should also send updates messages (in batches), publishing changes after a point in time as a batch. This is important as it allows the possibility to respond to changes by subscribing to these change events with the Message Broker.

Capabilities, functional requirements, mechanisms and interoperability guidance

Table 40: Capabilities Table of Building Block 10

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards
2 – Experimental Twins	31	Interaction support.	Integration	MIM6	N/A

Table 41: Mechanism Table of Building Block 10

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1.	31	The solution should allow to keep track of all changes so that the new situation can be reconstructed from	Use of identifiers (i.e., scenario identifier) to	JSON, XML or binary serialisation through protocol

		the baseline scenario at any time by any component.	support message durability	buffers, applying FAIR principles
2.	31	The solution should be able to support to publish all messages on the message broker of the LDT core system so other changes can be detected.	Pub/sub mechanism	
3.	31	The interaction service BB should expose a simple API that allow to fetch changes so that changes can be processed.	APIs	
4.	31	The solution should support data source communication when referred to in a scenario.	Connection via data source identifiers through a data catalogue	
5.	31	The solution should be able to call upon models when referred to in a scenario.	Pub/sub mechanism	N/A
6.	31	The state of the data should be respected when a chain of modules is activated (i.e. called upon) through a scenario.	Pub/sub mechanism	N/A
7.	31	The solution should be able to fall back to the baseline scenario at any given time.	Identifiers and storage mechanisms	FAIR
8.	31	The interaction service should allow for topic-based communication that is triggered based on events.	Publish/subscribe mechanism	
9.	31	Send a message from the data query service to the model abstraction service with minimum storage utilisation per message in the message queue.	Binary serialisation	Protocol buffers
10.	31	The solution should be able to facilitate asynchronous communication (between publishers and subscribers).	Communication protocols	AMQP, MQTT, Websockets, REST, SOAP, Protobuf
11.	31	The solution should be scalable to accommodate increased message rates and when the number of participants grow.	Binary serialisation of data, standard messaging formats	JSON, XML, Protofbuf
12.	31	The solution should provide reliable message delivery, ensuring messages are not lost in transit.	Communication protocols	AMQP, MQTT, Websockets, REST, SOAP, Protobuf
13.	31	The solution should accommodate different communication requirements.	Communication protocols	TCP/IP, MQTT, HTTP/HTTPS

Examples

Currently, there are no industry standards or general solutions for the interaction service BB. However, it is possible to envision an (open) API specification along with some business rules and technical requirements for the creation of such a service. Ideally, the solution supports standard data messaging formats. Solutions using binary data formats as a payload require fewer bytes than data exchange in e.g. JSON or XML and thus should be considered when designing the interaction service. This should ensure a completely data driven approach in an LDT. It is important to consider the links with the other BBs in an LDT environment as the interaction service performs some important functionalities to support scenario-based analysis (e.g., to be useful in an interactive workshop setting with stakeholders). The interaction services support interdomain dependencies which require data and models to interconnect and form a computation process to enable simulation processes. A single-domain approach will not manage this and there are no standard software packages or frameworks that exist that support the capabilities needed if a Digital Twin is to be used for scenario-based analysis in a quick and responsive.

There are examples that support interaction of models and data in a human- readable format (like JSON or XML); however, when running scenarios with multiple configurations these tend not to perform sufficiently. For example, a central database to which a simulation model fetches its data and from which the visualisation interface reads, in terms of data speed, this might become a bottleneck. Instead, it is more efficient to stream data changes between the BB components directly. As such there are proven best practices that have designed interaction services as BB in an LDT to enable the interaction needed, e.g., a chain of simulations. An example of a type of interaction service in play is the Inter Model Broker Framework[28]. It uses protocol buffers to exchange messages in a binary format and induce changes across modules through a publish/subscribe mechanism. It improves the throughput of the data and reduces latency by reducing the data size transferred. Another example is the interaction service within the DUET framework[29].

Examples of high-level specifications from DUET are listed below, containing the need of interactions with other services such as the message broker, data query service, and how the interactions are supported.

[28] Lohman, W., Cornelissen, H., Borst, J., Klerkx, R., Araghi, Y., & Walraven, E. (2023). Building Digital Twins of cities using the Inter Model Broker framework. *Future Generation Computer Systems*, *148*, 501-513.https://doi.org/10.1016/j.future.2023.06.024

[29] DUET_D3.9 Digital Twin data broker specification and tools v2.0_v1.0 (digitalurbantwins.com)

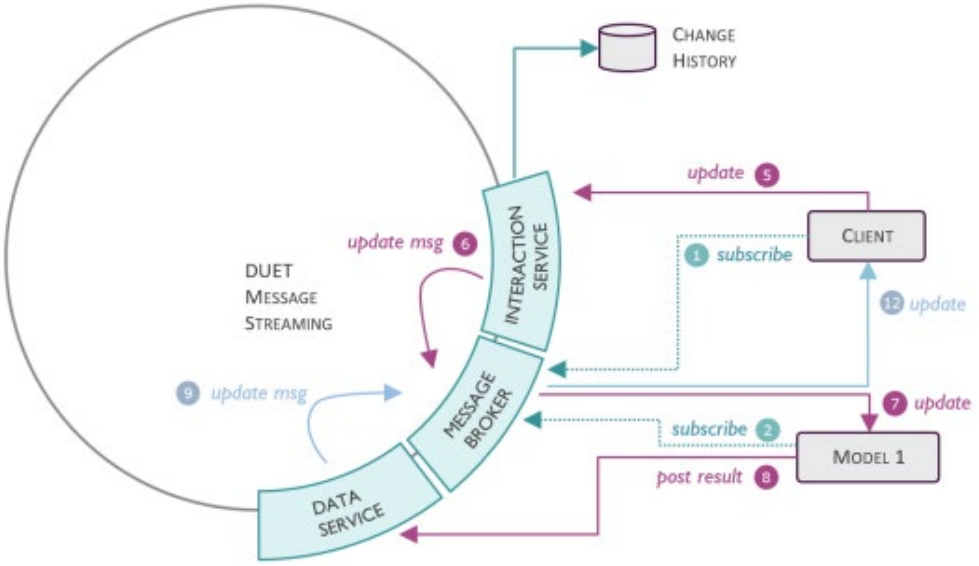


Figure 12: Interactions in an LDT in scope of interaction service of DUET framework

1. A client subscribes with the message broker to update events of an output of a certain model (i.e. prediction or simulation), which is a data sources associated with model 2.
2. Model subscribes with the message broker to update the events of a data source.
3. Model 2 subscribes to the message broker to update events of a model data sources associated with model 1.
4. Model 2 is then started and waits for incoming results which arrive at Model's 1 data source
5. Client 1 sends an update to the interaction service.
6. The interaction service publishes an update event associated with the update of client 1 on the message queue for the context data source.
7. The model 1 receives the updates message(s) from the Message broker for the client update and responds appropriately. This can mean that it can fetch the data needed for re-computation, it can activate a new run on the data, etc.
8. Model 1 writes the results to a database attached to the LDT framework.
9. When model 1 published its result (data source), a message is pushed to a message queue associated with the result data source.
10. Model 2 receives a result notification and fetches the data from a data service (could be from the data catalogue). Model 2 takes this data, incorporates these results in an ongoing model that is running.
11. Model 2 publishes its results to a data source. A message with these new results is pushed onto the message queue.
12. The client receives a notification of the changes to the data and responds appropriately.

3.2.11 Building Block 11 | Message Broker

Table 42: Summary Table of Building Block 11

Minimum Ambition Level	2 – Experimental Twins
Kind	Software
Maturity Level	Good Enough
Internal code	BB.11
Relevant MIM	MIM5, MIM6

Description

The Message Broker is a critical component of an LDT architecture. It abstracts over the message streaming algorithm that provides streaming and buffering capabilities for high velocity data. The Message Broker serves as an intermediary layer to manage the distribution of messages, ensuring that they are correctly delivered to the right recipients and, if necessary, ensures that they are properly transformed to meet the recipient's needs. It also guarantees that proper access rights are verified when models or other data consumers try to access the data (in collaboration with the access manager- see Annex 6.1).

It serves the following purposes:

- Enables communication between different components of the system without these components needing to know about each other's existence.
- Maintains message channels (topics, queues, etc.), allowing data models, interaction services, and visualisation services to publish and subscribe to messages on these channels.
- Provides a unified API, abstracting over the underlying message streaming service, making it easier for various parts of the system to interact with the message streaming service.
- Ensures reliable message delivery, even in the face of system failures. It should provide features like message persistence, delivery acknowledgments, and retry mechanisms to guarantee message delivery.
- Provides features for message filtering and routing so that subscribers receive only the messages in which they are interested.
- Scales to manage high volumes of messages while maintaining low latency.

A mechanism to make the message broker function is to operate it on the principle of a publish-subscribe mechanism and message queueing. Message queuing enables LDTs to facilitate data exchange between simulation models and other system components. They implement an asynchronous communication mechanism between processes, in which multiple processes do not communicate to the central queue (i.e., Message Broker) at the same time. This is supported by a publish-subscribe mechanism. With publish-subscribe, publishers (data producers) send messages to the Message Broker without knowing who the subscribers (data consumers) are to certain topics. The Message Broker then takes care of delivering the messages to all interested subscribers to a data source or model in an LDT (i.e., once a message is published for a certain topic, all subscribers will receive it). This enables one-to-one, one-to-many, or many-to-many communication, and enables event-based systems in which multiple distributed processes are interacting with each other such as in a LDT environment.

The Message Broker is used for the functioning of different components of the reference architecture but has specific relationships with the interaction service and model abstraction service.

The Message Broker must be chosen or designed to be interoperable with the Interaction Service and the Model Abstraction service. **This requirement may lead to the need for a (limited) abstraction layer over the chosen message streaming platform**. There are several message streaming solutions /message brokers available. There are several open-source and commercial message brokers available, each with their strengths and trade-offs. Some well-known options include RabbitMQ, Apache Kafka, Google Pub/Sub, Amazon SQS/SNS, and Azure Service Bus.

Capabilities, functional requirements, mechanisms and interoperability guidance

Table 43: Capabilities Table of Building Block 11

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards
2 - Experimental Twins	30	Data flows and component orchestration	Integration	MIM5, 6	
	31	Interaction support	Integration	MIM6	
3 - Predictive Twins	7	(Near) Real-time data processing	Data	MIM5, 6	LDES, Kafka, MQTT, Web Sockets, …
	8	Data, publishing and subscribing	Data	MIM5, 6	LDES, Kafka, MQTT, Web Sockets, …
	3	Ingest data	Data	MIM5, 6	
	4	Streaming data	Data	MIM5, 6	LDES, Kafka, MQTT, Web Sockets, …
	27	Alerts and Notification	Visualisation	MIM5, 6	
	35	Urban measuring, sensing & control	IoT	MIM5, 6	NGSI, HTTP, LDES, OGC
4 - Intelligent Twins	34	Supervised or unsupervised actuation, command & control	IoT	MIM5, 6	NGSI, HTTP, LDES, OGC

Table 44: Mechanism Table of Building Block 11

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1.	4	The message broker should be able to support for publish-subscribe models	Pub/sub mechanism	Protobuf
2.	4, 30, 31	The message broker should support the ability to queue messages when subscribers are not immediately available for processing	A method/function that allows communication between different components via the message broker.	Apache Kafka, RabbitMQ, etc.
3.	27	The message broker shall ensure reliable delivery and prevent message loss	A method/function that allows models to subscribe to different channels on the message broker. Error handling.	MQTT, Apache Kafka, RabbitMQ, etc.
4.	4, 30, 34, 35	The message broker shall be scalable, enabling to handle large number of messages involved when processes become more abundant. While also maintaining low latency.	Support multiple languages, performance considerations	Protobuf
5.	8, 30, 31, 34, 35	The message broker shall enable communication between different components of the system without these components needing to know about each other's existence.	Pub/Sub	Apache Kafka, RabbitMQ, etc.
6.	4, 7, 8, 30, 31	The message broker shall be able to maintain message channels (topics, queues, etc.), allowing data models, interaction services, and visualization services to publish and subscribe to messages on these channels.	Support for asynchronous method calls and call backs.	Asynchronous Programming Principles, Promise/Future-based APIs
7.	4, 8, 30, 31	The message broker shall provide a unified API, abstracting over the underlying message streaming service, making it easier for different parts of the system to interact with the message streaming service.	API	API guidelines

8.	3, 8	The message broker can handle the offering of features for message filtering and routing so that subscribers receive only the messages they are interested in.	Protocol support, filtering mechanisms	MQTT, Apache Kafka, Rabbit MQ

Examples

Table 45: Examples of tools for Building Block 11

Name	Open Source	Popularity (GitHub Stars)	Community Size	Standards Adherence
RabbitMQ	Yes	7k+	Large	AMQP, MQTT, STOMP
Apache Kafka	Yes	19k+	Large	Proprietary protocol (Kafka Protocol)
Google Pub/Sub	No (Service)	N/A	Google Cloud users	Publisher-Subscriber Model
Amazon SQS/SNS	No (Service)	N/A	AWS users	SQS/SNS Proprietary protocols
Azure Service Bus	No (Service)	N/A	Azure users	AMQP, SBMP (Service Bus Messaging Protocol)
Apache ActiveMQ	Yes	Medium	Medium	Java Message Service
ZeroMQ	Yes	High	Medium	Transport Protocols TCP, UD

Alignment with SIMPL specifications (https://ec.europa.eu/newsroom/dae/redirection/document/86727)

Before final design and implementation of this component, an alignment with the following building blocks from the referenced SIMPL implementation roadmap plan is needed:

- "*Paas Services*": messaging busses

3.2.12 Building Block 12 | Model Abstraction service

Table 46: Summary Table of Building Block 12

Minimum Ambition Level	2 – Experimental Twins
Kind	Software
Maturity Level	No Existing Standard
Internal code	BB.12
Relevant MIM	MIM9

Description

The Model Abstraction service can be an API or Software development kit that serves as a vital conduit in a Digital Twin setup, facilitating the control, orchestration, execution and composition of diverse models. Its purpose includes supporting different execution modes, enabling interoperability and managing model data. Additionally, it ensures secure access, offers scalability, supports error handling, and provides logging for performance tracking. This component acts as the backbone of any Digital Twin, bridging the gap between complex models and the user interface.

Within a Digital Twin, multiple models will be used to reflect various aspects of the urban environment, and the interactions of these aspects will need models to be chained together and with other assets such as the data they produce and consume, although they can be hosted within different model pipeline and workflow environments. There have been previous attempts to standardise model interactions by defining abstractions, but they have not led to a state-of-the-art standardised abstraction or open interface to be used as a given now. For that reason, the goal of this BB is to provide a glue between the different models aiming to handle them in a uniform manner. There can be different deployment methods, e.g., deploying all models in one framework which would make this simpler, but this cannot be used as a standard method in a world where models are delivered as a service, as a downloadable object, or as a licensed software package. In the end, the goal of this abstraction service could be seen as a kind of BPMN process engine that can link to existing frameworks or industry-proven APIs.

A Model Abstraction API/SDK should ideally support the following **abstraction features** (that then eventually can trigger the functionality in the other BBs such as the orchestration engine):

- **Execution Management**: Support for executing models in synchronous, asynchronous, batch, or real-time modes. Users should be able to initiate, control, and monitor model execution, and also cancel or pause it if necessary. This feature should also support different deployment modes, like local, distributed, edge, cloud, etc.
- **Parameter Configuration**: Support for defining and modifying the parameters of models. This can include hyperparameters, inputs, and outputs.
- **Data source specification**: The API/SDK should facilitate the specification of the data required for models as well as the output data. It should provide a way to bind input data and capture output data from model execution.
- **Model Lifecycle Management**: It should support various stages of a model's lifecycle, from training to deployment to monitoring to updating/retraining.
- **Model Monitoring and Logging**: It should support tracking model performance and health and provide logging functionality for tracking model execution details.
- **Interoperability**: The API/SDK should facilitate interoperability, i.e., the ability to work with several types of models, different modelling languages and frameworks, different data formats, etc. This may be harder to achieve in high-performance environments, so it is a nice-to-have. Compliant API implementations in different language/framework will work equally well.

Of course, cross-cutting concerns are to be considered such as access control, scalability, error handling and debugging. In the reference architecture, the model abstraction service operates together with following BBs: *Algorithms & Models, Data workflow and Component Orchestration, Asset Registry, Model & Data Catalogue, Interaction service.*

The Model Abstraction service is triggered by an orchestration workflow and shall use information that is related to the model and data sources that is registered in the Asset Registry and the Data and Model Catalogue.

Capabilities, functional requirements, mechanics and interoperability guidance

Table 47: Capabilities Table of Building Block 12

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards
2 – Experimental Twin	28	Model abstraction	Algorithms & Models	MIM9	De facto industry (see examples)
2 – Experimental Twin	29	Model hosting	Algorithms & Models		De facto industry (see example)

An important remark is that MIM9 is in an early draft at the moment of writing this report, and aims to make complex models interoperable, allowing more efficient analytics and impactful exchange of expertise, to allow cities to leverage each other's successes in data analytics. Thus, this BB is the first step towards that goal and can provide a uniform access to models by providing the elementary abstraction features mentioned.

Table 48: Mechanism Table of Building Block 12

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1.	28, 29	The API/SDK shall provide a uniform interface to interact with different models irrespective of their underlying implementation.	A unified set of methods/functions to load, execute, and interact with models	SDK Design Principles, Language-agnostic APIs (gRPC, REST)
2.	28, 29	The API/SDK shall allow models to publish their output to the message broker.	A method/function that allows models to publish their output to different channels on the message broker	LDES, MQTT, Apache Kafka, RabbitMQ, etc.
3.	28, 29	The API/SDK shall allow models to subscribe to data from the message broker.	A method/function that allows models to subscribe to different channels on the message broker	MQTT, Apache Kafka, RabbitMQ, etc.
4.	28, 29	The API/SDK shall support the tracking of the execution state of the models.	Functions to get/set the execution state of the models	State Machine Principles (e.g., SCXML)
5.	28, 29	The API/SDK shall be able to handle errors and exceptions gracefully.	Proper error and exception handling mechanisms	SDK/Programming Language-specific error handling principles.
6.	28, 29	The API/SDK should provide logging functionality.	Integration with a logging system.	Common Logging Libraries and Services.

7.	28, 29	The API/SDK should support model versioning.	A method/function to handle different versions of a model.	Semantic Versioning
8.	28, 29	The API/SDK should allow asynchronous execution of models.	Support for asynchronous method calls and call-backs.	Asynchronous Programming Principles, Promise/Future-based APIs.

Examples

To our knowledge, no usable standards or tools exist that provide an appropriate level of abstraction for the broad range of available model types. However, examples tuned to specific (typically ML) workflows exist:

Table 49: Examples of tools for Building Block 12

Tool	Open Source	Popularity & Community Size	Completeness	Standards
TensorFlow Extended (TFX)	Yes	High: Part of TensorFlow ecosystem. Active contribution and usage.	High: Supports full lifecycle of ML models, including abstraction and composition.	Adheres to the principles laid down by TensorFlow ecosystem.
Kubeflow	Yes	High: Active contribution and usage. Supported by many major companies.	High: Offers a comprehensive set of tools for deploying and managing ML workflows, including support for model composition (on top of Kubernetes).	Kubeflow components communicate via Kubernetes APIs.
MLFlow	Yes	Quite popular, especially among data scientists for simple tracking of ML experiments. Active and growing community.	Offers an abstract interface for packaging, executing and managing diverse ML workflows, being able to work with any ML library, algorithm, deployment tool or language.	Offers its own APU and model vision and is language agnostic. It contains " flavours" that glue to industry standards like PyTorch, TensorFlow).

This report also refers to the model abstraction approach implemented in the DUET project as an inspiration source.

3.2.13 Building Block 13 | Model Catalogue

Table 50: Summary Table of Building Block 13

Minimum Ambition Level	2 – Experimental Twins
Kind	Software
Maturity Level	Good Enough
Internal code	BB.13
Relevant MIM	MIM5

Description

The Asset Registry was already described as a fundamental BB to track all kinds of assets and their changes throughout an LDT's lifecycle, such as data, infrastructure, algorithm, model, and assets, and also more specific LDT assets such as cases, scenarios and experiments. As algorithms are more frequently used within Digital Twins to simulate the effect of changes, and even to predict future state, specific attention is needed to the use of algorithms and models within an LDT as the transparency of using digital assets like models is key in taking decisions in a societal and public context. For example, where the impact of the decisions has a high impact surface (citizens, businesses, etc.) within cross-influencing societal domains (environment, mobility, energy, safety, etc.).

With the rise of AI in the city landscape, special attention is needed to keep track of which models are being used and their purpose in order to align with the external facing MIM5 capabilities of fairness, transparency, explainability, and even accountability. A Model Catalogue plays a vital role in the management of machine learning models (and other) in a Digital Twin environment. It is responsible for the storage, versioning, and metadata tracking of machine learning models. Where an Asset Registry focusses more on the internal and technical aspects of the asset management and governance, the model catalogue exposes the algorithm assets in a more elaborate way towards external usability.

Some key features of a Model Registry/Catalogue include:

- **Model Versioning**: Like how code versioning systems like git work, a model registry should be able to keep track of different versions of models, allowing users to easily switch between different model versions and compare their performance.
- **Model Metadata Tracking**: A model registry should track important metadata for each model, such as the training data used, hyperparameters, model performance metrics, and more. This makes it easier for data scientists and engineers to understand the conditions under which a model was trained and how it is expected to perform.
- **Model Lifecycle Management**: A model registry should support the transition of models through various stages of a typical machine learning workflow, such as development, staging, and production.
- **Model Discoverability**: The registry should make it easy for users to discover models and share them with others, facilitating collaboration among teams.
- **Provenance**: The Model Registry should track the origin and transformation history of each model. This includes who created the model, the data used to train it, the changes it underwent over time, and where and when it has been deployed or used. This is important for traceability and for understanding the context in which the model was developed and used.

- **Transparency**: The Model Registry should provide clear and understandable documentation for each model, including its purpose, performance, limitations, and any biases it might have as well as datasets used for training, if applicable. This is important for users to understand how the model works, to use it appropriately, and to interpret its outputs correctly.
- **Accountability**: The Model Registry should log all actions related to each model, including who made the changes and when. This is important for accountability and for maintaining a trustworthy system.

These attributes are crucial for maintaining a high degree of trust in the models being used, particularly in a context where decisions that affect real-world outcomes, potentially impacting the lives of citizens, are being informed by these models. This is even more crucial when considering regulatory requirements and ethical considerations.

In an LDT context, the Model Catalogue could store different versions of models representing various aspects of the urban environment. These models could be deployed in a modular and interchangeable manner to support a variety of simulation and prediction scenarios. Different versions of a model could be compared to determine their effectiveness in various contexts.

Capabilities, functional requirements, mechanisms, and interoperability

Table 51: Capabilities Table of Building Block 13

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards
2- experimental twins	19	Algorithm & model management	Analytics	MIM5	
4- intelligent twins	49	Accountability information provisioning	Governance & compliance	MIM5	Algorithm Register

The basic functional requirements for an Asset Registry also exist for a model catalogue, applied to model assets. The following table focusses on additional functional requirements that are able to feed an algorithm register application as described in the examples section. An important remark is that Model Catalogues could be federated if the federation catalogues offer the same functionality, and a federation workflow is defined.

Table 52: Mechanism Table of Building Block 13

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1.	19	The catalogue should be able to manage and keep track of different versions of models.	Versioning mechanisms and storage systems	Version Control systems, naming conventions, containers
2.	19	The catalogue should provide authenticated and secure access to its content.	Authentication and authorisation mechanisms	OpenID Connect, OAuth 2.0, TLS/SSL
3.	19	The catalogue should track important metadata information for its models.	Use standardised vocabularies and data schemes to store them and serialise using e.g.,	Use of DCAT, schema.org, DAML, Google Model Cards, AIMMRS,

			JSON, XML, JSON-LD, etc.	algorithm register standards
4.	19	The catalogue should track the state of models during their typical ML workflow stages.	Use and integrate with SOTA stage descriptions and states from major tool vendor	Data Version Control, TFX, PMML, ONNX, Seldon, MLFlow
5.	19	The catalogue should provide discoverability of models and offer access to the results.	Use of standardised metadata and API for model access	Based upon the above-mentioned interoperability guidance practices for requitements 3 and 4
6.	19	The API/SDK shall allow models to publish their output to the message broker.	A method/function that allows models to publish their output to different channels on the message broker	LDES, MQTT, Apache Kafka, RabbitMQ, etc.
7.	19	The catalogue API/SDK shall provide uniform interface to interact with different models irrespective of their underlying implementation.	A unified set of methods/functions to load, execute, and interact with models	SDK Design Principles, Language-agnostic APIs (gRPC, REST)
8.	19,49	The catalogue needs to collect and offer important metadata on non-technical aspects such as bias, purpose, limitations, ethical elements, liabilities, etc.	Bias and fairness metrics, model explanation techniques, responsibility tracking, misuse/privacy/societal impact listing, data usage rights, IP rights, liability clauses	Fairness toolkits (Aequitas, Fairlearn, IBM AI Fairness 360, etc.), LIME/SHAP tools, identity systems, act guidelines
9.	19,49	The model catalogue should work together with other BB to offer an insight into the creation and usage of its models, grouping all the relevant metadata.	Integration with Asset Registry, workflow and orchestration... BBs , data catalogue, data storage systems, message broker, etc.	Algorithm standard definition can be used as guideline

Examples

A good example of a broader take on this with a focus on dissemination is the algorithm register standard that is currently under development. A concrete example of how a tool can be upon this standard to bring transparency to the citizens can be seen in the Amsterdam Algoritmeregister. This tool explains the following key elements that address MIM5 capabilities:

- How do the algorithms impact me and my everyday life.
- What is an algorithm and what is it used for.
- What ethical principles are followed when using algorithms.

It provides information, in a human readable format, on used datasets, data processing logic, fairness and non-discrimination aspects, human control in the loop and risk management. To get the necessary information, the tool needs to get data from the model catalogue in a standardised way. Within the

standard, there is an ongoing vocabulary and scheme definition to make sure that different model catalogues can feed the same information.

Multiple tools already exist in the market that hold information for machine learning models and can be used as the base composition to create a model catalogue for Digital Twins to automate the transparency and explainability required by MIM5. These tools can be used to feed the standard on algorithm registers for cities and communities. The model catalogue would rely on and connect with information from the Asset Registry and data catalogue to establish interfaces to not only feed other internal BBs (such as workflow orchestration or model abstraction), but also feed the tools that explain the use of the models to the citizens and other stakeholders.

Examples of the basic features for registries exist; however, most products are focusing on machine learning models. An overview of selected **algorithm tools** that can be used depending on the circumstances is highlighted in the table below.

Table 53: Examples of tools for Building Block 13

Product	Open Source	Popularity/Community	Standards Adherence
MLflow	Yes	High	Partial
Seldon Alibi	Yes	Medium	Partial
TensorBoard	Yes	High	Yes (TensorFlow)
Neptune.ai	No	Medium	Partial
Kubeflow	Yes	High	Yes (Kubernetes)
ModelDB	Yes	Low	Partial

Alignment with SIMPL specifications (https://ec.europa.eu/newsroom/dae/redirection/document/86727)

Before final design and implementation of this component, an alignment with the following building blocks from the referenced SIMPL implementation roadmap plan is needed:

- "*Data discovery*": data catalogue, metadata description. A model catalogue can have a lot of the same functionalities as a data catalogue and needs a metadata description. The SIMPL implementation could maybe be aligned with the tools above to feed it.

3.2.14 Building Block 14 | Model usage guidelines

Table 54: Summary Table of Building Block 14

Minimum Ambition Level	2 – Experimental Twins
Kind	Software/ Guidelines
Maturity Level	No Existing Standard
Internal code	BB.14
Relevant MIM	MIM5, MIM9

Description

Model usage guidelines are considered an essential component of the critical path for the development of LDTs as from the ambition level for Experimental Digital Twins and all levels above. This BB is not a technical BB and may be part of an LDT setup for a city of community, but it can also be applicable as a support tool that is part of an LDT Toolbox implementation.

They form a framework that allows users to determine appropriate modelling techniques in relation to the problem that needs to be solved including data-level considerations to be considered. This includes guidelines on how to implement models as well considering aspects on the algorithms/techniques to use, appropriate space/time resolutions for processing data, streaming requirements, data quality considerations and so on. The categorisation of models, algorithms, and data sources can be a key enabler for users to figure out which technologies are suitable in a given context, and what data requirements are involved.

Here is a suggested categorisation approach:

- **By Use Case**: This is the most straightforward categorisation, where models, algorithms, and data sources are associated with particular use cases. For instance, in an urban planning use case, 3D visualisation models and GIS data might be the most relevant.
- **By Domain:** Different models exist within certain domains, like urban planning, flood prediction, air quality impact simulation, and noise simulation.
- **By Data Type**: This categorisation can help users understand what type of data is required for different models and algorithms. For example, time-series data, image data, geospatial data, etc. would each require several types of models and algorithms.
- **By Task (or purpose)**: Models and algorithms can also be categorised by the type of tasks they perform, such as simulation, prediction, classification, clustering, anomaly detection, optimisation, etc. Users can select the appropriate model or algorithm based on the task at hand.
- **By Complexity**: Some models and algorithms require more computational resources (as multiple models can interact with each other e.g., traffic, air pollution and noise models) or more complex data pre-processing steps than others. Users can make a choice based on their available resources and complexity tolerance.
- **By Transparency**: Depending on the application, users may need models that offer varying levels of transparency or explainability. For instance, simple linear regression models are highly transparent, whereas deep learning models might be considered "black boxes".
- **By Regulation Compliance**: Depending on the geographical location and nature of the use case, different regulations may apply. For instance, GDPR in Europe has certain requirements for data handling that might affect the choice of data sources and models.
- **By Technology:** A lot of algorithms exist that are modelling physical behaviour based on understanding of the nature of the problem. But more data-driven AI algorithms are entering the game, and even become challengers of the physical models. Especially looking at the specific technology aspects and their regulatory dynamics (such as the AI Act), it is clear to establish model usage guidelines that take this into account and correlate this (e.g., within an Asset Registry) to the transparency and accountability expressed within MIM5.

In terms of data requirements, the following considerations can be kept in mind:

- Volume: The size of the dataset needed.
- Velocity: The speed at which data needs to be processed.

- Variety: The types and sources of data that are required.
- Veracity: The reliability or trustworthiness of the data.
- Value: The usefulness or relevance of the data to the case or scenario.

A comprehensive catalogue detailing these categories can help users make informed decisions when selecting technologies and managing data for their LDT applications.

Capabilities, functional requirements, mechanisms and interoperability guidance

This report proposes a tool that helps an LDT user navigate the complex labyrinth of data and algorithms. By selecting a use case or a typical problem, the tool can suggest appropriate techniques along with associated data requirements and concerns (based on a data requirement featuring the above categorisation).

Table 55: Capabilities Table of Building Block 14

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards
2 – Experimental Twins and above	55	Model governance & compliance	Governance & Compliance	MIM5, MIM9	N/A

This BB is as contributor to the capabilities required by MIM5, as it assists in the explainability and transparency. The model usage guidelines can also be used to make complex data models more interoperable in the end, and thus creating standards for, future models to be more tuned to the needs, which is an enabler for MIM9.

Table 56: Mechanism Table of Building Block 14

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1.	55	The BB should have a comprehensive database of use cases/problems and their associated modelling techniques.	A relational or graph database could be used to store this information	The database schema should adhere to common database design principles to ensure data consistency, integrity, and reliability
2.	55	The BB should allow users to search for use cases/problems and retrieve the corresponding modelling techniques.	A search function with advanced filtering and sorting options could be implemented	N/A
3.	55	The BB should provide detailed information about each modelling technique, including its technology, strengths, weaknesses, complexity, and data requirements.	The tool should have a mechanism to fetch and display this information upon user request	N/A
4.	55	The BB should provide guidelines or suggestions on the data-level requirements and considerations	The tool should have a mechanism to fetch and display this	N/A

		linked to the use of each modelling technique.	information upon user request	
5.	55	The BB should have a user-friendly interface that makes it easy to explore use cases/problems, modelling techniques, and their data requirements.	A clean and intuitive UI design is essential	The user interface should adhere to common design principles for usability and accessibility
6.	55	The BB should be able to handle updates to the database (e.g., new use cases/problems, modelling techniques, etc.) without disrupting its functionality.	A backend system that allows for database updates without downtime is needed	N/A
7.	55	The BB should provide educational resources for users who are not familiar with certain modelling techniques or data-level considerations.	Integration of learning resources or links to reputable learning platforms could be beneficial	N/A
8.	55	The BB should have robust error handling and provide clear, helpful error messages to the user.	Proper error handling mechanism and error message design is necessary.	N/A
9.	55	The BB should provide an API for other services to fetch its data and make use of its functionality.	A RESTful API, or GraphQL could be used to achieve this	The API design should follow common practices like using HTTP verbs, status codes and should provide a well-documented interface
10.	55	The BB should be able to scale to accommodate growing user demand.	The system architecture should be designed to handle increased loads, possibly by using cloud-based solutions or adopting a microservices architecture	N/A

Examples

No existing tool currently fulfils these requirements, especially on categorisation. However, (open source) tools can be listed and used as starting points, or are at least partially feature complete, and they need to be evaluated in detail if they can serve the purpose or if the tool can be built on top of them. Most of these tools come from the machine learning world and are used also to feed other BBs like model catalogues.

- *TensorFlow Model Garden:* this tool provides pre-trained models with usage guides.
- *PyTorch Hub:* this tool provides pre-trained models with usage guidelines.
- *H2O.ai:* provides automated machine learning models which include a set of pre-defined models.

- *ModelDB:* system to manage ML models and associated experiments.
- *Seldon:* open-source platform that helps organisations deploy ML models in Kubernetes.
- *MLFlow:* platform to manage the ML lifecycle, including experimentation, reproducibility and deployment.
- *OpenML:* an open platform to share datasets, algorithms and experiments.

3.2.15 Building Block 15 | Algorithms & Models

Table 57: Summary Table of Building Block 15

Minimum Ambition Level	2 – Experimental Twins
Kind	Software
Maturity Level	Good Enough
Internal code	BB.15
Relevant MIM	MIM1, MIM4, MIM5, MIM6

Description

An LDT should enable the development towards inclusion of different algorithms and models related to the challenges of smart cities and communities. While for most of the individual domains different models exist that allow stakeholders to understand the current status and what-if scenarios of each domain individually, the LDT's strength lies in making these algorithms & models uniformly available to all stakeholders to address evidence-based cross domain decision making. As such, in an LDT context, the idea is that multiple domains can interact. The role of the algorithms and models BB is therefore a crucial enabler of the strength of an LDT. The role of algorithms & models can be described according to the following[30]:

- to interpolate spatially where no direct measurement data is available;
- to integrate the presentation and analysis of complementary data sets;
- to cross-correlate data from different domains;
- to infer properties/attributes that are not directly measured;
- to convert measurements and state info into KPIs; and
- to extrapolate (predict) business as usual (BAU) and what-if scenarios.

However, it is important to note that no single way to create an all-encompassing approach of modelling all the different actions and activities taking place in an urban environment in an LDT exists. Rather, a multitude of model components exists that can be combined to enable the possibility to capture all these complex interactions and activities. As such, the BB algorithms and models describe the type of models that should be compatible in an LDT to add value and the possibility to access and integrate those models in an LDT environment allowing stakeholders to understand, plan, analyse, predict and make more data-driven decisions in order to improve urban developments (i.e., city's

[30] D3.4 Smart City domains, models and interaction frameworks v2 (p.10)

behaviour and its various subsystems under different conditions). They play a pivotal role for ambitions level for Experimental Twins and higher to enhance the capabilities envisaged when using LDTs.

As mentioned, models have a variety in type and are developed differently. There are many providers of models' solutions that could be deemed fit to work in an LDT environment. Therefore, an LDT should ensure the possibility to integrate different type of models in an LDT environment. For example, it is envisaged that each city will already have an own traffic model in place that should be able to integrate in an LDT environment. This could also be complemented with other models offered by model developers (such as air quality models calculating emissions or noise pollution models). In addition, supporting tools for developers to create environments where models can be developed from scratch (e.g., machine learning models) should also be considered when integrating in an LDT environment. Thus, a type of connector should be developed that allows for an entrance (e.g., API) so an LDT can communicate with different models and module environments (i.e., supporting tools). This should be done in such a way that a data analysis or model prediction development can be performed directly on data coming from an LDT. In addition, models in an LDT environment are often not standalone, but interact with each other (such as traffic models, with noise and air pollution models). Therefore, these models need strong orchestration support to ensure responsiveness and low latency of complex model interactions.

Thus, there are strong interactions with other BBs when setting up algorithms & models to work in an LDT environment:

Model usage guidelines

The model usage guidelines support the selection of the correct models in determining appropriate modelling techniques in relation to the problem that needs to be solved including data-level considerations to be considered. It therefore helps in the consideration which type of algorithm or model to choose and thus integrate in an LDT environment.

Asset Registry

The Asset Registry facilitates the discoverability and the support of finding the correct information of datasets and models and algorithms.

Model abstraction service

The model abstraction service has an important link with the algorithms & models as it serves as an important orchestrator of enabling multiple models to interact within an LDT environment.

To specify the added value of the BB on algorithms and models, it is important to distinguish the several types of algorithms & models that can be used in an LDT environment. The differences in types of models and its rightful application lie in their learning approach, flexibility in conducting predictive tasks, the use cases the algorithms and models are applied to, the chosen KPIs and model interpretability. As such, several types of algorithms & models can be distinguished in an LDT environment: 1) Simulation; 2) Prediction; and 3) Machine learning & AI.

Simulation

Simulation models involve techniques to create virtual representations of a cities environment to test and visualise potential outcomes of different scenarios. It uses mathematical models and algorithms to simulate behaviour and interaction of various elements within an urban environment. Simulation enables to end-user to understand the consequences of different actions and decisions in a controlled environment. This can also include complex dynamic system modelling capturing behaviours and interactions within an urban environment over time. Examples of simulation models for an LDT environment are:

- Traffic simulation models:
    - Microscopic traffic simulation models:

    - Macroscopic traffic simulation:
- (eEnergy consumption levels:
    - Building energy simulation:
    - District energy simulation:
- Agent Based models:
- Environmental models:
    - Air quality models:
    - Water management models:
    - Noise pollution models:
- 3D geospatial models: Smart grid models:
- Infrastructure resilience models:
- Social interaction models:
- Waste management models:

Prediction

Prediction models are statistical models that use historical data and models to predict or forecast future events, trends, or outcomes. It allows users to make informed estimates about future events based on historic data and statistical methods. Prediction models learning approach is based on using a predefined set of mathematical equations or rules. The model parameters are estimated using statistical techniques. The model's performance is evaluated based on how well it fits the historical data. Examples of prediction models are:

- Traffic flow and congestion prediction
- Modal split predictions
- Impact of new mobility predictions
- Energy consumption prediction
- Air quality prediction
- Water demand and supply prediction
- Public transportation demand prediction
- Real estate market prediction
- Urban growth and land use prediction
- Weather induced event prediction
- Waste generation and collection prediction

Machine learning & AI

Machine Learning & AI models are models that learn from the data and improve their performance over time. These types of models use patterns in the data to predict or make decisions without being explicitly programmed for a certain predictive task. These models learn from data through training and validation. Iteratively they adjust their parameters to minimise prediction errors and their ability to generalise to new, unseen data. They are considered flexible in the sense that they can adapt to changing patterns and handle complex relationships in the data. Thus, can handle a wide range of tasks and types of data. The different type of machine learning & AI models are:

- Supervised learning
- Unsupervised learning
- Semi supervised
- Reinforcement learning
- Deep learning algorithms
- Natural Language Processing
- Ensemble Learning
- Nearest Neighbour
- Decision tree

- Bayesian algorithms

How to implement algorithms & models in an LDT environment?

The BB on models and algorithms is a BB that show the models that can be used in an LDT. Depending on the use case and objectives of the LDT different type of models can be called upon and interact with each other. The way these models can be called upon is through packaging models inside a container enabling the possibility to deploy models and access models in the cloud. The models run outside an LDT core environment and are interconnected using an API. It allows users to bundle a model or algorithm and all its dependencies, libraries, and model configurations together into a single container. Where, when and which models are deployed when requested will be controlled by a container orchestrater (e.g., Kubernetes) via a set of suitable scripts will then regulate the interaction of an LDT system and models.

It is important to note that models in an LDT environment are controlled by a model abstraction service and corresponding API, which makes deployable models upon request possible. The interaction service is responsible for executing the deployment, calling upon models whilst also considering resource requirements. By packaging the model in a specific format, the container technology can use a set of scripts to which a docker orchestrator can receive message (indirectly) through the Model Agent API to request to start, stop or receive status checks of the models use. A start request includes context information of a model (its parameters, data, etc.) that is pushed and passed to the model. A model's response message includes a unique instance id to control or get information of the model. When a model is deployed, it should respond to messages of the orchestrator, additional information or properties, data, links can then be passed into to context information section with the start request of the model.

However, the way models interact depend on how tightly these models are intertwined. In Figure 9 for example, it is noteworthy that many components of various levels of a system are intertwined and looping back (either direct or indirect) to modules of their input (e.g., travel demand depends on generalised cost and the generalised cost depends on travel demand).

All these smaller modules that can make up for example a transportation model module in its own right. In an LDT environment, one of the most important challenges for the BB algorithms & models is to find a compromise between the decomposition of models and associated modules (which allow for more flexibility and interactions), and the effort needed to integrate these via APIs so that developers or analysts can work with them. Thus, it may be necessary to provide 'handbooks' or 'guidelines' of models developed to allow the design of models suitable to fit the needs of the challenges tackled with an LDT to allows e.g., to indicate calculation sequences of different models for certain analysis. As such, when integrating different models it is important to carefully consider the circular dependencies, the modules involved within a workflow, which will require involvement of the right expertise.

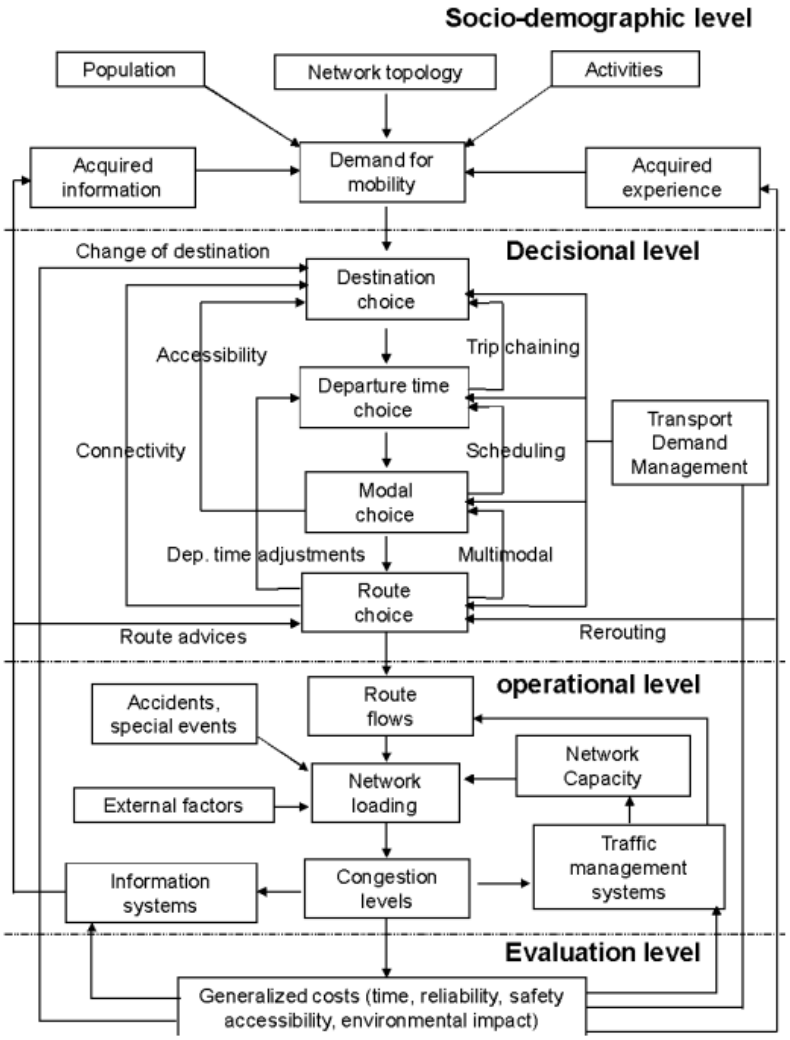


Figure 13: Transportation system structure and model modules

Capabilities, functional requirements, mechanisms, and interoperability guidance

Table 58: Capabilities Table of Building Block 15

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards
2 – Experimental Twins	16	Simulation	Analytics	MIM 1, 4, 5,6	NGSI-LD
3 – Predictive Twins	15	Prediction	Analytics	MIM 1, 4, 5, 6	NGSI-LD
3 – Predictive Twins	21	Machine Learning & AI	Analytics	MIM4, 5, 6	

Table 59: Mechanism Table of Building Block 15

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1.	15, 16, 21,	Models should be deployable in the cloud.	Containerisation	Docker
2.	15, 16, 21,	Models should be able to define their own channels or topics for exchanging information in an LDT system.	APIs, context information	JSON, XML
3.	15, 16, 21,	A set of suitable scripts should organise the interaction with the LDT system and the models in the cloud.	APIs	HTTP
4.	15, 16, 21,	Models are deployable upon request and should receive message to start (deploy), stop (delete) the model.	APIs	JSON
5.	15, 16, 21,	A start request from a deployed model should include model context information (i.e., parameters, data).	RESTful APIs	RESTful HTTP
6.	15, 16, 21,	Model should have IDs assigned so that the container of the model can be located .	API	JSON
7.	15, 16, 21,	Models should be supported in exchanging data to and from the models.	API	JSON
8.	15, 16, 21,	Models should be accurate, reflect confidence in their predictions which represent real-world urban systems and their behaviours.	Evaluation metrics	N/A
9.	15, 16, 21,	Models should be scalable to accommodate and integrate the vast amount of data generated	Orchestration mechanisms	Kafka, RabbitMQ
10.	15, 16, 21,	Models in an LDT environment should be adaptable to incorporate new data, handle evolving conditions and respond to events.	API	REST API
11.	15, 16, 21,	Model related information (i.e., about its performance or context) should be provided to aid on insights into the decision-making process.	Metadata, transparency guidelines	AI act
12.	15, 16, 21,	A link with an orchestrator should enable model deployment in LDTs to be executed in the right order accompanied by using right usage of resources.	Orchestration mechanisms	Kubernetes
13.	15, 16, 21,	A LDT should support the interaction of different type of models and provide a connection to integrate different models	Orchestration mechanism, connectors	APIs

14.	15, 16, 21,	Use of models in a LDT environment need to capture all the effects that empirical analysis has deemed relevant (scientifically proven and accepted)	Scientific literature	N/A

Examples

Table 59 describes a list of examples of various tools, applications, developer environments that support the development of algorithms and models that could deem fit to be integrated in an LDT environment. Some cover multiple capabilities (e.g., simulation and machine learning & AI) at the same time, other require additional plugins, but at the base have a link with the BB algorithms and models. It is important to note that the fit of these tools are dependent on the role of the data and which use case or objective of the LDT it will support.

Table 60: Examples of tools for Building Block 15

Tool	Open Source	Popularity & Community Size	Standards
TensorFlow	Yes	Large (part of TensorFlow community)	API
OpenML	Yes	Large	API
Eclipse MOSAIC	Yes	Medium	API
Eclipse SUMO	Yes	Medium	API
MATSim	Yes	Medium	API
Apache BEAM	Yes	Medium	API
NoiseModelling – Noise-Planet	Yes	Medium	API
PTV Visum	No	High	API
PTV VISSIM	No	High	API
CESIUM	Yes	High	API
OmniTrans	No	High	API
SRM1, SRM2[31]	Yes	N/A	API
OPS	Yes	N/A	API
Predictor-LimA	No	High	API
LOTOS-EUROS		Medium	API
TraMod	Yes	Medium	API

[31] https://www.digitalurbantwins.com/_files/ugd/725ca8_5060e85784164f52a3b0ff3348d7e951.pdf

CityFlows	Yes	Medium	API
MatlabTrafficToolBox	Yes	Medium	API
Jupyter Notebooks	Yes	High	API
QGIS	Yes	High	API
Keras	Yes	High	API
CitySim	No	High	API
EUReCA	Yes	Medium	API
OpenTrafficSim	Yes	Medium	API
UrbanFootprint	Yes	Medium	API
ATMO-Street	Yes	Medium	API
ATMOSYS	No	Medium	API
OpenIDEAS	Yes	Medium	API
ActivitySim	Yes	High	API
Microsoft Azure	No	High	API
RapidMiner	Yes	High	API
ArcGIS Hub	No	High	API

Alignment with SIMPL specifications
(https://ec.europa.eu/newsroom/dae/redirection/document/86727)

Before final design and implementation of this component, an alignment with the following building blocks from the referenced SIMPL implementation roadmap plan is needed:

- "*Application sharing*": calculation algorithms, machine learning models, software & apps

3.2.16 Building Block 16 | Asset Registry

Table 61: Summary Table of Building Block 16

Minimum Ambition Level	1 – Awareness of Twins
Kind	Software
Maturity Level	Plug & Play
Internal Code	BB.16
Relevant MIM	MIM2, MIM5

Description

A Digital Twin, in accordance with the GEMINI principles, necessitates a dual mandate of trustworthiness and efficiency, particularly in an environment where it is heavily reliant on external assets like data, simulation models, environment models, and more. Given the dynamic nature of these assets, it is crucial for a Digital Twin to incorporate a system for asset governance that is needed up till ambition level 4.

Asset governance (of digital assets) is defined as the capability (implemented by a process) to manage (technical) data schemas, data vocabularies, business glossaries, and their interrelations. An essential component of this governance is the management of asset metadata and the provision of lineage and provenance records for the utilisation of these assets, tracing back to the decisions made based on them. Thus, it enables to track quality and data lifecycles and is key to important aspects identified by MIM5 such as accountability, context, and various levels of transparency.

A base feature of this system is the Asset Registry, which serves as an index cataloguing the assets involved in the Digital Twin. While the asset documentation can exist external to an LDT instance, the registry is integral for traceability and discoverability. The Asset Registry provides an encompassing overview of the available datasets, schemas, vocabularies, algorithms, and other usable assets.

Expanding on the specific asset types:

Datasets: For any dataset employed within a Digital Twin, comprehensive understanding regarding its trustworthiness and quality attributes is paramount. This entails complete transparency into its lineage, ensuring integrity, and a detailed description of the data and its quality for accurate assessment of its suitability for the intended use. Moreover, it is imperative that the dataset's schema, format, and semantics are explicitly defined.

Schema, Format, Vocabulary: The Asset Registry should possess the capability to list available formats, ensuring assessment of data usability. Concurrently, it should be equipped to track the data schemas and their associated vocabularies (if applicable) for understanding schematic and semantic compatibility with algorithms, processors, and/or visualisation tools.

Ontologies: The Asset Registry should incorporate ontology management, including the ability to import/link to externally generated ontologies and artifacts. It is crucial to avoid ad-hoc asset governance with informally defined glossaries.

Algorithms: Documenting algorithms in terms of their data inputs and outputs can facilitate their usage and composition with data sets or other algorithms. Providing basic descriptions of algorithms and defining their reliability, quality expectations, and limitations is crucial.

Other Assets: Besides algorithms, such as simulation or machine learning models, other data-transporting or data-transforming assets may also be made available. Proper documentation of these assets in the registry aids their efficient and appropriate usage. Also specific LDT concepts (like cases and scenarios) must be expressed as assets within this registry.

The Asset Registry is a part of the core technology of a Digital Twin, and thus has quite some links with other BBs, for example:

Case & Scenario management

This BB has a clear responsibility to registers its assets (cases, scenarios, experiments) into the Asset Registry, but also use assets within as pointers for its interaction with other BBs.

Data and Model Catalogues

While data and model catalogues serve more the business purpose of discovering and using the items they catalogue, the Asset Registry is more integrated with the software BBs and also registers digital assets that are not necessarily needed in catalogues. The assets in the registry support tracking of the

different versions of the assets and their usage inside the workflows and lifecycle of the Digital Twin. For that purpose, the Asset Registry can certainly feed the catalogues withs specific information that needs to be exposed towards catalogue users.

Data workflow and component orchestration

Asset registries can provide provenance details on how the assets are being used in certain executions (e.g., in scenario experiments). For that, the orchestration component can add provenance pointers towards the asset usage history, which can be maintained in the Asset Registry, and thus can contribute to more technical transparency from the asset viewpoint.

Capabilities, functional requirements, mechanisms and interoperability guidance

This report focusses on the key aspects of an Asset Registry. The factual governance and management of data and the management of models and algorithms is more a concern of the data catalogue and model/algorithm catalogue, respectively. But the Asset Registry's task is to disclose this data for potential users giving the insight in type, schema, semantics, format, provenance, lineage, intended use, and qualitative aspects for data sources. Similarly for models and algorithms, aside from discoverability, the Asset Registry can provide details such as the model publisher, any training sets used, assumptions about input data sets, output quality expectations, intended use, transparency concerns, fairness concerns and privacy concerns.

The requirements below can be optionally extended to include the publishing of other resources such as ontologies, schemas, formats, vocabularies, and transformations.

Table 62: Capabilities Table of Building Block 16

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards
1 – Awareness of Twins	2	Ontology management	Data	MIM2	W3C, OSLO, OGC, …
	1	(Meta)Data Schema management	Data	MIM2	Data model definitions (e.g. OSLO, Fiware Smart Data Models)
	11	Data source Management	Data	None	DCAT-AP
2 – Experimental Twins	56	Semantic governance & compliance	Governance & Compliance	None	N/A
4 – Intelligent Twins	46	Procedural Transparency	Governance & Compliance	MIM5	N/A
	47	Technical transparency & explainability	Governance & Compliance	MIM5	N/A

The Asset Registry is an important enabler for MIM2, as it is used to store the data models (schemes and vocabularies) and aids in the provenance to use them throughout the lifecycle of the Digital Twin.

Moreover, it is key in providing the transparency capabilities of MIM5. This report pays more attention to this in the examples section below and how it is instrumental in feeding the new standard for algorithm registers identified as the key goal of MIM5 used in open smart cities.

The below table lists the main requirements for this BB, with suggestions on the technical mechanisms to address them, and hints on existing interoperability practices or standards.

Table 63: Mechanism Table of Building Block 16

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1.	1, 11, 2	The Asset Registry should allow for adding and discovering data sources based on type, schema, semantics, format, provenance, lineage, intended use, and qualitative aspects.	Use of APIs for data source discovery and management	RDF, DCAT-AP, OpenAPI, GraphQL
2.	11	The Asset Registry should support querying and browsing of data sources using the aforementioned parameters.	Use of query languages with data source interrogation capabilities	SPARQL, GraphQL
3.	11, 56	The Asset Registry should allow for adding and discovering models and algorithms based on publisher, training sets used, assumptions about input data sets, output quality expectations, intended use, transparency concerns, fairness concerns, and privacy concerns.	Use of APIs for model and algorithm discovery and management	RDF, OpenAPI, DCAT-AP, GraphQL
4.	11	The Asset Registry should support querying and browsing of models and algorithms using the aforementioned parameters.	Use of query languages with model and algorithm interrogation capabilities	SPARQL, GraphQL
5.	46	The Asset Registry should facilitate logging and tracking of asset usage.	Implementation of a data usage logging and tracking system	N/A
6.	47	The Asset Registry should allow for documentation of assumptions and expectations of models and algorithms.	Use of APIs to facilitate the documentation of assumptions and expectations	N/A

Examples

Below is a list of tools that can comply with the needs of an **Asset Registry**.

Table 64: Examples of tools for Building Block 16

Tool	Open Source	Popularity & Community Size	Features Completeness	Standards
CKAN (Comprehensive Knowledge Archive Network)	Yes	High	High	OpenAPI Specification (OAS)
Apache Atlas	Yes	Medium	High	Open metadata standards
Amundsen (Lyft)	Yes	Medium	Medium	OpenAPI Specification (OAS)
Alation	No	High	High	Proprietary standards
Collibra	No	High	High	Proprietary standards
AWS Glue	No	High	High	AWS standards
DataHub (datahubproject.io)	Yes	Medium	High	Flexible metadata model

As an example of an open-source tool**, DataHub** is an open-source metadata platform that enables data discovery and search, collaboration, governance, and end-to-end observability, created at LinkedIn and open-sourced in February 2020. It functions by having metadata providers push the information and updates to the platform via API's or messages, rather than letting the platform setup scrapers for domain-specific data sources. It can also be deployed in a federated infrastructure, where different metadata services are owned and operated by different stakeholders. It is both freely available as an open-source project or as a managed SaaS platform through the company Acryl Data.

Please note that the open-source tools, while they may have lower direct costs, may also require more expertise and effort to set up and maintain. Remember that the availability and capabilities of open-source tools may change over time, so it is essential to check for the latest developments. The choice of tool should therefore be based on your specific requirements and capacity for implementation and support.

Integration with an algorithm register tool

Smart cities have become more aware of the risks of using intelligence on data within their cities, and their increased accountability concerns. For that purpose, cities like Amsterdam, Barcelona, Brussels, Eindhoven, Mannheim, Rotterdam, and Sofia have formed a community to realise MIM5 using a tool to bring transparency, context and explainability towards the citizen (and other stakeholders), thus inducing fairness and accountability. The standard for this is being developed[32] within the context of Eurocities. LDTs bring this intelligence to a next level, using data and algorithms to suggest policy actions, or even controlling current city behaviour live-time, and thus use several of these assets.

[32] https://www.algorithmregister.org/standard

The Asset Registry and its artefacts bring context, understanding (technical and non-technical level) on the assets used in a digital way, and thus can play an instrumental role in the automation of algorithm register tools as keeping these tools manually up to date will not scale in highly intelligent cities and communities using LDTs. For that reason, this BB is not only mandatory for MIM5 in general, and the capabilities mentioned above, but also for realising tools like algorithm registers that give a face to MIM5.

Alignment with SIMPL specifications (https://ec.europa.eu/newsroom/dae/redirection/document/86727)

Before final design and implementation of this component, an alignment with the following building blocks from the referenced SIMPL implementation roadmap plan is needed:

- "Data Discovery": Data catalogue, metadata description, search engine
- "Application Discovery": application catalogue, metadata description, search engine

3.2.17 Building Block 17 | Synthetic Data Generation Tool

Table 65: Summary Table of Building Block 17

Minimum Ambition Level	2 – Experimental Twins
Kind	Software
Maturity Level	Good Enough
Internal code	BB.17
Relevant MIM	MIM1

Description

Synthetic data generation tools are software tools or frameworks designed to create artificial or simulated data that closely resemble real-world data. These tools use algorithms and statistical models to generate data with similar statistical properties, distributions, and relationships as the original data. Synthetic data generation tools are useful for various purposes, such as data augmentation, privacy protection, and testing machine learning algorithms and models. An example of a popular synthetic data generation tool is that of synthetic populations to which it can act as input for example Agent-Based Simulations, mode-choice prediction etc.

Within an LDT environment when running different cases, scenarios and experiments the access to real world datasets is often limited or challenging, especially for datasets that contain characteristics of human behaviours such as traffic movement. Synthetic data generation tools can provide significant value to provide high level insights with datasets that resemble actual data without compromising privacy or security. More specifically, using synthetic data different simulations and analyses can be conducted, it allows the augmentation of existing datasets and the creation of additional data points enabling scalability and enhancing the Digital Twin's accuracy, model training a validation of machine learning and Ai models as synthetic data can complement real-world datasets, testing and evaluation of performance of algorithms and multiple scenario simulations.

Synthetic data generation tools are software services that can be offered on a marketplace and accessed via a connecting service to an LDT. They have a strong link with BBs such as the data query service or Asset Registry in which references to synthetic data generation tools can be made.

Capabilities, functional requirements, mechanisms, and interoperability guidance

Table 66: Capabilities Table of Building Block 17

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards
2 – Experimental Twins	33	Context Information	IoT	MIM1	
3 – Predictive Twins	9	Synthetic data generation	Data		

Table 67: Mechanism Table of Building Block 17

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1.	33	It should provide support listing the context information, specifying the structure and types of data to be generated to ensure consistency with real data formats.	Data model formats, API	NGSI-LD
2.	9	The tool should be able to capture the statistical distribution and diversity of the real-world dataset.	Iterative Proportional Fitting or other methods to maintain statistical properties that meet with the real-world dataset	N/A
3.	33	The tool should be able to generate and handle varying sizes of datasets.	Low latency, data models and formats	NGSI, JSON, XML, CSV
4.	9	To protect privacy, the tool should have mechanism to anonymise sensitive information ensuring no personally identifiable information is included.	Differential privacy, k-Anonymity and l-diversity, data perturbation, sampling replacement	GDPR
5.	9	To maintain accuracy and validity of simulations, the synthetic data should closely resemble the real data.	Iterative Proportional Fitting	N/A
6.	9	The tool should enable a user to define specific attributes, statistical properties and rules for data generation.	UX	N/A

7.	9	Depending on the domain, the solution should support domain specific features and data types.	Domain specific approaches may be required	N/A
8.	9	The solution should support generating data that reflects the diversity, temporal and spatial variation and heterogeneity present in real-world data, capturing various patterns, dynamics and relationships.	Random sampling and noise injection, rule-based generation, GANs, VAEs, copulas, MARKOV chains, Bayesian networks, differential privacy, various hybrid approaches	N/A
9.	33	The solution should have built-in data validation mechanisms to check for defined constraints and adheres to specific data models.	Statistical property preservation, data type validation	N/A
10.	9	The solution should be able to generate synthetic data quickly.	Parallel processing, efficient data models, data precomputation	N/A
11.	9	The solution should support versioning and reproducibility to main consistency of generated datasets over time and across different runs.	Version control mechanisms	N/A
12.	33	The solution should provide comprehensive documentation and user support.	Software documentation	N/A
13.	9	Synthetic data should support compatibility with data analytics and ML libraries.	Interoperability frameworks, APIs	REST

Examples

Table 68: Examples of tools for Building Block 17

Product	Open Source	Popularity & Community	Standards Adherence
Gretel	No	High	Partial
Synth	Yes	Medium	
Populationgenerator (TNO)	No	Medium	Partial
Databaker	No	Medium	Partial
OpenSynthetics	Yes	High	Partial

Syndata	No	Medium	Partial

Alignment with SIMPL specifications
(https://ec.europa.eu/newsroom/dae/redirection/document/86727)

Before final design and implementation of this component, an alignment with the following building blocks from the referenced SIMPL implementation roadmap plan is needed:

- "*Data Processing*": data anonymisation

3.2.18 Building Block 18 | Data Storage

Table 69: Summary Table of Building Block 18

Minimum Ambition Level	2 – Experimental Twins
Kind	Software
Maturity Level	Good Enough
Internal code	BB.18
Relevant MIM	MIM2

Description

Data Storage (DS): Data storage refers to the retention of data, provided and recorded through various recording devices or provided by users. Databases are designed to store and manage vast amounts of data in a systematic manner, allowing users and applications to access, query, and manipulate information.

From the physical storage perspective, databases usually come in various forms of Hard Disk Drives, In-Memory storage, and Cloud Storages. Hard Disk Drives use magnetic storage for storing data on disks such as CDs, DVDs, and Blu-ray discs. In-Memory storage stores data in the system's memory rather than on magnetic disks to provide rapid retrieval access. Cloud Storage store data on the remote machines (servers) accessible through the internet.

To clarify the barrier between the data storage BB of an LDT and the relation to the European Digital Infrastructure Consortium (EDIC)[33] activities, it is worth to elaborate and distinguish the relevancies. EDIC is dedicated to synchronising two fundamental aspects – data and models – with data space. In pursuit of this objective, EDIC aims to establish a comprehensive platform facilitating European stakeholders to exchange language data and models seamlessly. On one hand, the platform empowers stakeholders to fine-tune and develop a Large Language Model, enabling the creation of data-driven products, turnkey solutions, and customised language technologies and services. This cohesive approach ensures innovative advancements in language technology, fostering collaboration and innovation throughout the European community.

[33] https://digital-strategy.ec.europa.eu/en/news/digital-assembly-2023-digital-open-and-secure-europe

Regarding data storage, an LDT can provide a logical data layer for its own relevant data as part of its Toolbox to integrate and transform data to any external subsystems, including any subsystem which uses the common European Data Spaces data sharing platform.

For such functionality, three aspects need to be considered. Firstly, metadata modelling is essential for representing the storage structure efficiently. Secondly, a virtual data access point is established, facilitating easy accessibility to the stored data. Lastly, data transformation, which has already been addressed as part of the data replication BBs, completes the necessary components to enable the smooth and effective operation of an LDT data storage.

Regarding the physical storage aspect, a data space concerns the space for data storing that can be cloud-based, for example. Indeed, the data storage BB does not concern the data space aspect, meaning that an LDT data storage can be hosted in any preferred space.

Databases from Structure Perspective

In data storage management, three architectures have emerged applicable for any data or technology-driven application, including Digital Twin: Data Lake, data warehouse, and data lakehouse. Each of each serving distinct purposes and catering to specific data requirements.

- **Data warehouse:** A data warehouse collects raw data from various sources into a centralised repository (storage) and organises it into relational database format[1] i.e., structured data format.
- **Data lake:** A storage repository which can store vast amounts of data in its natural format, such as raw, unstructured and structured data. This data model is designed to store data in its most natural form i.e., raw, unstructured, semi-structured or structured, without the requirement of predefined schemas or data transformations without prior need of schemas or data transformations. Raw data is usually collected from various sources, such as sensors, logs, databases, and applications and it comes in various heterogeneous formats, including JSON files, CSV files, and SQL scripts.
- **Data Lakehouse:** Striking a balance between the best aspects of data warehouses and data lakes, the revolutionary concept of a data lakehouse emerges. The key aim is to address the challenges and limitations associated with traditional data warehouses and data lakes to support various forms of data, raw, structured, and unstructured. Data lakehouses utilised similar data structures from data warehouse and combines with the low-cost storage and flexibility of data lakes, enabling organisations to store and retrieve data efficiently while mitigating potential data quality issues[34].

Big data challenges for database design and implementation

The diversity of data sources in Digital Twins gives rise to a significant challenge related to big data, as outlined in the big data taxonomy[35]. This challenge refers to “variety”. In the context of big data, variety refers to the differences in data formats and models within the collected data, stemming from the diverse range of data sources.

Specifically in level 3 and 4 of maturity level of an LDT, variety of data sources become even more obvious as historical data serves as the ground truth or the ideal expected result against innovative approaches for comparison, prediction or real-time simulation algorithms. But also, many IoT-based data will be added as time-series and event-based data sources, which demands consideration of even more structured data formats to the data storage. This leverages the emergence of data lakehouse,

[34] https://www.ibm.com/topics/data-lakehouse

[35] Ceravolo, P., Azzini, A., Angelini, M., Catarci, T., Cudré-Mauroux, P., Damiani, E., Mazak, A., Van Keulen, M., Jarrar, M., Santucci, G. and Sattler, K.U., 2018. Big data semantics. *Journal on Data Semantics*, *7*, pp.65-85.

which is proposed to encompass a broader range of data source types, spanning from unstructured to structured data.

Data storage also deals with volume and velocity as two other prominent big data challenges. The first challenge, volume, pertains to effectively managing and indexing massive amounts of data. The second challenge, velocity, involves efficiently handling the speed at which new data is stored, modified or gets updated: Data Operations on database.

On the design level, the primary focus lies in addressing the challenge of variety, while during implementation, the emphasis shifts to tackling the challenges of volume and velocity both on the storage but also in database querying where the latter is discussed in the Data Query Service BB.

Database Model implementation: Code-First Approach (vs Design-First Approach)

The database design adopts a code-first approach, which is applied in domain-driven design, placing a strong emphasis on the application's domain during the database design process. Unlike the traditional method of first creating the database and then adding entities and relations, the code-first approach involves initially coding/implementing entities and relations within the programming platform.

Working with the code-first approach empowers developers to work at a higher level of abstraction compared to conventional methods like the Design-first approach. The code-first approach functions as an object-relational mapper, establishing an object-oriented layer between the relational database and the object-oriented programming language, thus eliminating the need to write SQL queries.

Federated access to data in a system of systems landscape

Aligned with the SIMPL specs (https://ec.europa.eu/newsroom/dae/redirection/document/86241) for the architecture vision, and IDSA/Gaia-X standardization, data storage needs to be interpreted within a federated landscape, using abstracted access to the data using e.g., the query and replication service components from the reference architecture. "Local" storage can be seen as a LDT application-specific store taking part in the federated access layer or attached to application functionality directly.

Table 70: Capabilities Table of Building Block 18

Ambition Level	Capability Nr	Capability Name	Category	MIM to consider	Standards
2 – Experimental Twins	10	Data storage	Data	MIM2	
2 – Experimental Twins	13	Data processing	Data		
2 – Experimental Twins	5	Data replication	Data		LDES
2 – Experimental Twins	12	Data time travel	Data		LDES, Temporal
3 – Predictive Twins	6	Data transformation	Data		SQL, GraphQL, SPARQL, REST, LDES, NoSQL

The data storage needs to interact with other building blocks as below from the reference architecture.

Asset Registry

Asset Registry locates and accesses various data sources across an LDT system. Data storage as a data access layer on a higher level can receive information from Asset Registry on potential queries via Data Query Service as intermediator.

Data Query Service

Data Query Service is applicable for the queries received from any inter or intra subsystems, and in case of any query received from access point of data storage, it needs to be revoked.

Capabilities, functional requirements, mechanisms, and interoperability guidance

Table 71: Mechanism Table of Building Block 18

Nr	Capability Nr	Requirement Description	Mechanism	Interoperability Guidance
1.	14	The data model among database entities needs to be able to extensible and the historical changes on database model needs to be trackable and revertible.	Code-First Approach	MIM2
2.	14, 5	The data storage shall be able to store data in different formats (structured, un-structured, semi-structured).	RDBMS, Object-Oriented, NoSQL, Hierarchical format	MIM2
3.	14, 5	The data storage shall be able to store data in different formats (structured, un-structured, semi-structured).	Either data lake and data warehouse or data lakehouse	
4.	14,16	The data Storage shall be able to manage data operation .	Repository database pattern can be employed	
5.	10	Cross-Sector accessibility (data access point).	APIs	Restful

Examples

Table 72: Examples of tools for Building Block 18

Tools	Open Source	Description	Popularity	Features	Standards
Microsoft Entity Framework	Yes	object-relational mapper (ORM) for Microsoft's .NET	High	Possibility of using LINQ for database operation, Strongly Object-Oriented, Code first approach	

Hibernate	Yes	object-relational mapper (ORM) for Java	High	Easy ORM, Easy transaction definition, possible to integrate with spring jpa	
PostgreSQL	Yes	Free and open-source relational database management system	High	Extensibility, Reliability, ACID, JSON, Parallel Query Execution	MIM2
MySQL	Yes	Open-Source Relational database management	Large open-source active community support	Portability, Performance Optimisation, Security and Scalability	SQL
MongoDB	Yes	Cross-platform open-source document-oriented database	High	Scalability, Flexibility, and Ease-of-use	GDPR, ACID, PCI
Apache Cassandra	Yes	Open-Source, distributed, NoSQL database management system	High	High availability, Fault-tolerance, Scalability, Distributed, Security and Observability	TLS/SSL encryption
Redis	Yes	Open-Source in-memory storage, used as a distributed, key-value database	Active open-source development	Speed, Simplicity, and support for wide range of data structure	TLS/SSL encryption
SQLServer	No	Relational database management system for storing and retrieving of the data	Microsoft proprietary	High Performance, OLTP support, BI integration support, JSON, Availability and Security	

Alignment with SIMPL specifications (https://ec.europa.eu/newsroom/dae/redirection/document/86727)

Before final design and implementation of this component, an alignment with the following building blocks from the referenced SIMPL implementation roadmap plan is needed:

- "*Paas services*": different kinds of databases
- "*Supporting- infrastructure management*"
- "*Data Sharing- data store connector*"
- "*Data governance*"
- "*Cloud computing & edge computing- storage provisioning*"

4. How to evolve from the LDT Toolbox design to implementation phase?

An LDT Toolbox design and the technical specifications as described in Chapter 3 provide the overall framework to be implemented by the EU within the overall ecosystem of EU digital solutions to support cities and communities in their digitalisation journey.

This chapter describes a process to evolve from the specifications level to definitions of the procurement needs for the LDT Toolbox and the necessary steps, illustrated by an example to identify the needs for the guidelines and/ or software tools from an LDT Toolbox.

4.1 Guidance for BB procurement steps

The objective of the Toolbox is to provide valuable solutions for cities and communities to accelerate the development of an LDT that corresponds as much as possible with the proposed reference architecture to ensure that their LDT solution can evolve according to the overall best practices from (L)DT implementations, smart city architecture and digital solutions for data sharing and e-governance. An LDT of a city will be a specific implementation, based upon reusable components from their own IT landscape (for instance using their cloud data storage platform) in combination with digital solutions from an LDT Toolbox and from other EU initiatives (e.g., DS4SSCC, data spaces), as clearly indicated in the reference architecture. Which tools are needed, depends highly on the specific use cases and ambition level that a city envisions, but for all implementations, the city's IT system landscape needs to be designed in such way that it can easily incorporate future functionalities without major impact on the IT landscape. In that perspective, the BBs of an LDT Toolbox are defined to be able to analyse the necessary functionalities for the LDT capabilities, referring to associated standards and complying with interoperability standards as described in the MIMs.

The abstraction level of describing the specifications on the level of BBs is needed to demonstrate how the LDT as a System of Systems application functions as described by the interactions in the reference architecture, but also to ensure that tools that are delivered to support functionalities for a BB, respond to common standards and functionalities and serve as the glue to functionalities enlisted. In order to go to a following step to define the procurement needs on the level of a software solution for a BB, there are two important observations to consider.

Firstly, the Maturity Level for a BB must be taken into consideration. The Maturity Level (*plug & play*, *good enough* or *no standard exists*) indicates the level of maturity of existing tools that comply with defined standards and requirements for the BB. For certain BBs, no standards or policies are defined yet whereas for others there is already a level of standardisation realised, but still requiring integration effort. The Maturity Level gives a first indication, but it is necessary to identify for each BB to what extent the market is aware of the standardisation and functional requirements as defined for the BBs to ensure that the proposed tools are fit-for-purpose within the context of the LDT.

An example is the case & scenario manager BB, for which no standards or example tools have been identified. Currently, scenario management is often integrated by the software provider as functionalities that are delivered as part of a specific model. However, the LDT reference architecture defines the case & scenario manager and the models as different BBs so that a city can apply a defined scenario from their case & scenario manager for different models that can be developed by different software providers. In order to achieve this interoperability, the standards for integration and interoperability as well for the case & scenario manager as for the models need to be agreed upon to make sure that the tools are developed in a way that they are reusable within different use cases across multiple LDTs. Thus, it is important to not only to look at the current market status of available tools (e.g., what models exist already), but also to evaluate which dependencies on standardisation requirements are necessary to consider. This does not mean that no models can be procured by a city yet, but it indicates that tools need to be procured as part of the LDT Toolbox alongside technical

analyses and design to ensure that the evolution of the LDT Toolbox contributes to the development and alignment with standardisation bodies. An example for the elaboration of specifications for a technical design for a BB of a tool is given in the description of the case & scenario manager BB where a data model is presented with suggested high-level APIs. This design work needs to be further elaborated and reviewed with the possible clients (advanced cities/regions) and standardisation stakeholders.

Secondly, the technical specification, as defined on the level of the BBs, document the capabilities and functional requirements for LDT purposes. However, when procuring a software tool, generic specifications related to security, compliance, legal, scaling and interoperability, and functional requirements need to be detailed and documented which are based upon the tool's scope and how it will be used. The BBs are defined as generic components with no reference to specific domains and cover different functionalities that can be addressed by one or many tools. Therefore, it is important to define the scope of the tool that will be procured. When a city implements an LDT, they will identify the domains that need to be covered. Thus, it is important to know if the procurement of a tool is meant to support a domain-specific use case or multiple domains, as it needs to be aligned with specific domain- standards and domain best practices. This means that a general assessment needs to be done to specify the purpose of the tools, in line with the overall reference architecture and other BBs, and document detailed functional requirements and specify the integration requirements based upon the use cases and functionality from related BBs that is required. Further elaboration on legal, security, compliance and technical requirements need also to be in line with industry standards, as is the case for any other city tool and can be inspired by the specifications from the **Smart Cities Reference Architecture Methodology**[36] and **the Reference framework for integrated management of an SSC** and **Procurement guidelines for SSC**[37]**.**

Considering the above-mentioned observations, it is advised to take a **top-down holistic approach to describe how to achieve the level of procurement details for an LDT Toolbox BBs**.

1. **BB assessment within the context of the reference architecture.**
   For each BB, a deep dive on the market maturity has to be performed and on the proposed standards and mechanisms. A thorough market analyses of existing solutions will provide more insights on classifying the different functionalities that are delivered by different tools. Moreover, the overall standardisation and mechanism proposed can be analysed to get insights in the current state of how the proposed standards are applied and can function within the context of an LDT reference architecture.

2. **Tool assessment and definition.**
   The purpose of this phase is to analyse which (type of) tools can be procured and define the usage scope of a tool to be able to detail the procurement specifications, especially on the level of functional requirements and integration requirements of a procurable solution. A possible approach to identify **which tools are most valuable for cities** is to start by identifying specific use cases that are of much importance for the cities and analyse the processes they want to support. When describing high level use cases, it becomes apparent what type of tools are needed and which interaction with other BBs is being addressed. It can be useful to document in detail the interactions on the level of the reference architecture or document generic BPMN flows which serve also as guidelines for cities when implementing their use cases. Based upon the use cases, an assessment can be done on which tools are needed and analyse what is already available in the market. The Community Organisation and Tool Development Body can identify **which gaps exist in the market** to comply with the desired requirements and standards and can **determine the set**

[36] IEC SRD 63188:2022 | IEC Webstore
[37] Publications – United for Smart Sustainable Cities (U4SSC) (itu.int)

of tools, guidelines or technical designs to be procured for the LDT Toolbox BBs. The definition of the use cases and interactions within the reference architecture provide the input to define the exact scope and purpose of the tools and for detailed, specific functional, technical and integrations requirements, in addition to the overall functional requirements and technology-related standards that are suggested for each BB described in Chapter 3.

3. **Tool specifications related to LDT Toolbox processes.**
   The deployment of an LDT Toolbox, as described in Section 2.2, will deliver specific requirements on generic processes regarding the validation process, testing requirements and actual provisioning and support process for the tools. For all tools, there must be overall **user acceptance criteria** defined, **testing requirements**, **support processes** (user community, software installation and user documentation, etc.) and **the lifecycle management processes** (how to deal with version control, bug fixing, etc.) that are based on a common approach in line with the organisational and technical setup of an LDT Toolbox. These elements are also part of the detailed procurement specifications for tools (can be guideline documents or software tools) for an LDT Toolbox.

4.2.1 Example on BB procurements steps

Example: what actions are needed to be able to ***procure a Predictive Model*** within an LDT and how can we relate that to the procurement of essential toolbox tools like ***model usage guidelines*** and ***model abstraction***?

This example starts from the context of a city that wants to procure a predictive model (see for some examples in the BB.15 description) and the data needed to assist the model training and inference (e.g., to train the machine learning model towards the local circumstances). The city wants to add prediction to its existing Open Urban Platform and pave the way to the step-by-step implementation of an LDT that is future-proof. Following this, there needs to be alignment on the procurement strategy and the support the Toolbox components can give in addressing some essential challenges to be future proof. For example, in integrating this model and future models in a seamless way with data and user experience assets used within the smart city range of IT systems. Lastly, it is important to note that this flow will have an impact on the procurement of some of the tools themselves, such as the ones based on the model usage guidelines and the model abstraction components.

4.2.2 Context of a City

There are several key challenges a city needs to address to add a predictive model to its Open Urban Platform and User Experience IT environment, users being city administrators, domain experts, or even citizens. Prediction models are not easy to procure, firstly because they virtualise a complex physical domain that has lots of parameters to consider. The best model for predictive analysis depends on several factors, such as the type of available data, the objective of the analysis, the complexity of the problem and the desired accuracy of the results.

There is no one solution that fits all these factors, and the best model to choose depends on data used, type of model (process-centric or data-centric models), technology used (e.g., for machine learning linear regression, neural networks, clustering, decision trees, etc.) and other parameters. As described in the ITU guidelines above within the “Six dimensions of a Digital Government”, it is key to be data-driven, and this is essential in procuring models, as the degree of availability of data now and in the future can have a very important impact on the choice of certain classes of predictive models, especially for the data-centric models. The quality of machine learning models decreasingly depends on the evolutions of the algorithm itself, but on the data that is fed to the algorithm (e.g., in continuous retraining scenarios).

In the context of a city, it is also very difficult to predict the future challenges to be solved, and thus to procure a model that is ready for that. For example, in urban mobility and transportation there are macro and micro models that all aim to solve different levels of challenges in the city and without domain knowledge, it is very difficult to take future-proof decisions. The ideal situation for a city would be to be able to use models-on-the-fly, for example in **model marketplaces or models-as-a-service[38] scenarios**, where predictive simulations can be run when they are needed without procuring the model itself. This allows cities to solve their problems with the most recent applicable technology and relieves the model suppliers to keep lots of model instances up to date. The future of procuring and using models could be buying access to model marketplaces, temporary licenses to models, subscriptions to model services. This link with model marketplaces and models-as-a-service is clearly indicated in the reference architecture.

An example of a process to address this flow is given in the BPMN diagram below. It is a high-level business process illustrating how data and predictive models can be used within a decision-making context within a smart city, for which 4 swim lanes have been identified:

- Swimlane 1 covers the scoping of the smart city challenge, its associated use cases and policy context. This can lead to specified cases and scenarios to be defined within BB.02. It is important to choose a business glossary, identify the needed and available data sets and how a view on predictions using the data can support the decision making.
- Swimlane 4 translates this towards the choice of datasets and models that can then be used to perform model serving and lead to fair and inclusive decision making, e.g. using BB.03 to engage citizens.
- The choice of datasets and models leads to their procurement and governance swim lanes represented in lanes 2 and 3. The procurement of the model can be seen as buying and integrating the model itself (from e.g., a model marketplace) or enabling the use of models-as-a-service or subscription licenses. Either way, it is crucial to evaluate the assets and bring these assets under governance and continuous management within the digital cycles.

[38] (PDF) Model as a Service (MaaS) (researchgate.net)

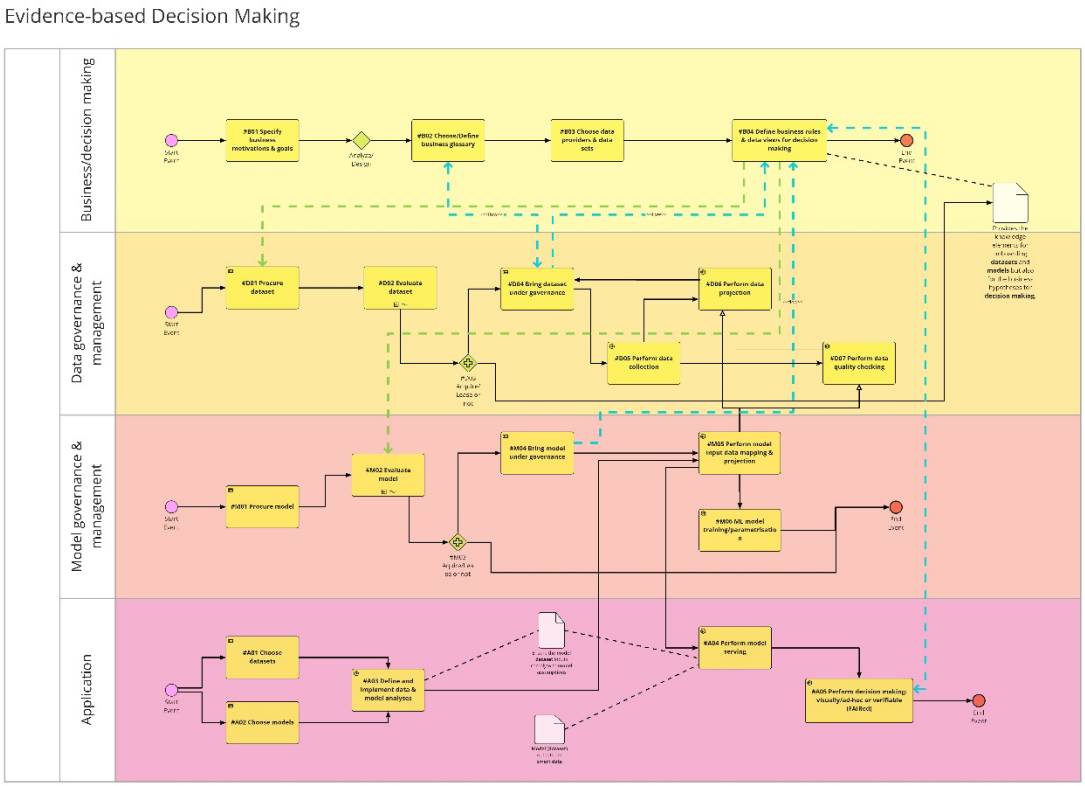


Figure 14: Example of a BPMN flow setting up a LDT application based on groomed decision-making motivation and goals and including procurement and governance of the essential data and model assets.

Source : https://vloca-kennishub.vlaanderen.be/Draaiboek:Evidence-based_Decision_Making

It can be concluded that the procurement of a predictive model for a city can be very bound to the specific goals and local conditions, and that it is not straightforward to align that with procurement of data, applications, and other IT assets. In the ideal case, cities can procure access to predictive models as services or temporary subscriptions. The availability and discoverability of models and descriptions of their specific parameters needs to be tackled by (federated) catalogues (e.g., federation of BB13 from a collection of LDT implementations, or from data spaces and marketplaces that are also able serve model access). But in any case, access to these models should be defined to make it easy, and the experiences with model usage should be made explicit. That is why the LDT Toolbox BB BB.14 (model usage guidelines) and BB.12 (model abstraction service) should lead to the realisation of tools that can accelerate these ambitions in a stepped way.

4.2.3 Guidance for procurement LDT Toolbox BBs, related to Models

As mentioned above, the Toolbox can bridge the autonomous world of models, algorithms and data towards the decision-making process in an interoperable way. Continuing the context of the city challenge described above, this report highlights 2 BBs at low maturity that are essential in bridging the worlds of models, city domains and workflows. These are the model abstraction service (BB.12) and the model usage guidelines (BB.14), both in the lowest maturity stage “no standard exists”. These BBs describe the power of their glue to scale LDTs, which are cross-domain by nature and therefore require several predictive models to serve their purposes.

First, the **model usage guidelines BB** is elementary in mapping the domain-needs and their challenges towards applicable technologies and algorithms for prediction, as the problems in one city will probably be contextual different due to environmental, data, legal and other parameters than in

another city or region. The proposed BB brings all this knowledge together and bundles this in a knowledge portal. This BB could be further evolved into a tool using the following steps:

- Gather knowledge for categorisation, techniques, challenges, among others from existing Digital Twin domain experts, the tools provided in the examples (e.g., TensorFlow Garden), model suppliers, among others, and create a vocabulary and data scheme to capture the information.
- Create a knowledge hub tool that can be used federated and feed it with the information gathered in the previous round. Make sure it can be updated, and that governance is installed on top of it.

This tool can then be used as a first step to understand what type of predictive modelling can be fit-for-purpose for the expressed challenge, narrowing down the search for procurable models that fit the criteria (e.g., in model marketplaces on during market analysis). This tool could be fed by model providers, model users, and other stakeholders, but it will need moderation and should keep its abstraction level and neutrality. This tool can also leverage the tendering process by using its terms and definitions to make it easier for companies to understand the problem and to enter into dialogue with the market on the criteria. New technologies, algorithms and more qualitative data is emerging every day, so the information gathered during tendering processes can flow back into the model usage guideline tool.

The **model abstraction service BB** plays a similar role, but not in the selection of model technologies based on common problem descriptions nor in the execution part of an LDT. Again, the maturity of this BB is “no standard exists” and thus the pragmatic choices will need to be made and inspired by existing tools that bring interoperable frameworks that are extended with some abstractions to fulfil the functional requirements mentioned. This BB could further evolve into a tool through two steps:

- First, gathering experience from users of the proposed examples and their features, and how they can address the functional requirements, adding experience from model suppliers that already are selling their models within these established frameworks. The gathered information can then create a concrete proposition for an API that is flexible enough for existing models to align on.
- Second, supporting the API with agents to glue it with the existing API models. These agents can then be designed, implemented, and offered, for example in the form of an SDK or open-source software packages.

The **Algorithm & Models BB** serves in understanding the different type of models that exist (simulation, prediction and machine learning & AI). This is because there are several accompanying complexities that need to be understood when considering intertwining models in an LDT context, and how to integrate models in a LDT context. The level of maturity and purpose of each model varies. However, it is important to note that no single way exists to create an all-encompassing approach of modelling all the different actions and activities taking place in an urban environment in an LDT. Rather, a multitude of model components exists that can be combined to enable the possibility to capture all these complex interactions and activities. As such, the BB algorithms and models should be further evolved to a tool where users of an LDT interested in integrating models can find:

- Descriptions of the type of models that should be compatible in an LDT should add value and information about the possibility to access and integrate those models in an LDT environment. Moreover, it should help stakeholders to understand, plan, analyse, predict, and make more data-driven decisions to improve urban developments (i.e., city’s behaviour and its various subsystems under different conditions).
- Knowledge, such as best practices and examples of use cases from cities that have implemented predictive models in an LDT environment. This includes lessons learned on intertwining models and sharing ideas or solutions on how to scale integrating multiple models

and examples of existing solutions or libraries (both open and proprietary) that offer predictive models that can be integrated in a LDT environment. Examples of such are listed in the algorithms and models BB.15.

- Interoperability guidance on how to integrate models from various sources (e.g., open-source libraries or model marketplaces) and the links with BB such as model usage guidelines and the model abstraction service.

Overall, the algorithms and models BB can be supported by some type of environment or page with interactive functionalities to support sharing of knowledge.

4.2.4 Guidance for procuring a model by a city, in relation to the LDT Toolbox

As mentioned, accessibility and integration of the right predictive model in an LDT environment highly depends on the context, objectives and needs of a city. As such, the responsibility for choosing which model and how the model should be executed (e.g., fitted to meet a city's objective) is under the city's authority in which the BB on model usage guidelines provide support. The BB on model abstraction service should facilitate the integration of models in an LDT environment from a technical point of view and thus integrating and intertwining models in an LDT environment in a standardised way. As such, the BB on the model abstraction service should be developed in a manner which allow greater interoperability to integrate multiple models depending on the city's need.

To provide guidance in the procurement of right predictive modelling tools that fulfil the importance of a BB in an LDT environment, an example outline is given on actions to consider when wanting to procure algorithms and models that could function in the context of an LDT. Choosing the right prediction, simulating, machine learning or AI model revolves around several steps to ensure the chosen model(s) align with the specific goals, data and complexities of the LDT that represent the city's challenges. The following steps should be taken into consideration:

Identify use cases & interactions in the LDT

a) Define the prediction objective: clearly articulate the specific prediction task (*operational-tactical-strategic*) what is needed to be addresses with an LDT (e.g., traffic flow, noise pollution, energy consumption or a combination of those tasks simultaneously).
b) Define interactions with other relevant BBs: determine integration requirements to ensure correct applicability of the model in an LDT environment (i.e., follow model usage guidelines) and technical interoperability with an LDT environment (i.e., follow principles of the model abstraction service)

Define minimum requirements

a) Functional:
   i. Specify what the algorithm or model should do and how it should perform (e.g., what is the preferred outcome, according to which KPIs, and determine how the model should be monitored and evaluated).
   ii. Determine features, capabilities and possible usability and UX requirements (e.g., user interactions).
   iii. Define a glossary to determine the right features and usage of terms are being implemented accordingly.
   iv. Note down all terms used.
   v. Note down the metrics used (NO2, CO2, etc.).
   vi. Determine geographical region.
   vii. Determine standards to adhere to in term of model interpretation (examples are noise levels, air quality).

b) Technical:

i. Determine the type of model (e.g., simulation, prediction or machine learning and AI). Depending on the type of model chosen, attention should be paid to the type of data acquisition is necessary. As the quality of the data greatly determines the prediction objectives.

ii. Determine model complexity and how the model should be integrated in an LDT environment, in particular emphasis should be made on ensuring model interoperability. Interactions with other BB should be foreseen, notable the model usage guidelines, model abstraction service and asset registry.

Determine sources

a) Data:

i. Ensure sufficient data availability to match identification of a use case.

ii. Ensure access to the right data (e.g., APIs, extracts, downloads or pipelines).

iii. Use data first to validate the problem – evaluate the data.

iv. If dataset is deemed relevant, ensure access to the data under the right data governance framework. This holds a strong link with the BB on asset registry.

v. Determine the necessary inputs for the model to determine the effort needed to get the right data. For example, in terms of data acquisition - finer grained models typically need data at a higher spatial resolution which in the end affects model validity. It is therefore advised to start with models that can be supported by larger, validated datasets (e.g., road network if traffic models are the starting point for analysis). Depending on the model complexity, and thus chosen models, additional data sources are needed, which need to find a way.

vi. Feature engineer necessary additional data features.

b) Model:

i. Align on business needs and prediction objectives in order to determine the right suitable model and its associated maturity.

ii. Train, score and evaluate multiple versions of the chosen model to determine model fit-for-purpose.

c) Applications:

i. decide whether the model will be created in-house, with or by external experts or consultants that will be hired or via built-in models from third party vendors (e.g. via model marketplaces) that will be procured or via a combination of the three. Consider factors such as the complexity of the modelled system or problem, availability of expertise, budget and timeline.

ii. Ensure transparency of the model to evaluate performance of the model not only visually but also validate on the input and output of the model, the link to prediction objective.

Vendor evaluation (if applicable)

- Reputation
- Maturity of the model
- Licensing and costs
- Support maintenance
- Integration requirements
- Customisation and extensibility

Interoperability guidance

Depending on the model complexities and model interactions (e.g., when multiple models run in sequence), the chosen software tools need to integrate with existing LDT environments. Therefore, it is necessary to specify the integration capabilities, APIs, data formats and interoperability requirements with other BBs.

Validation and Quality Assurance

A model is not run once but is an interactive task. Ongoing continuous validations tests are needed to ensure the predictive model accurately represent the real-world urban systems it is meant to simulate and is still aligned with the chosen predictive objectives.

Ongoing support and updates

Maintain ongoing communication with the source of the predictive model, especially if updates or modifications are anticipated based on changing urban conditions, requirements or regulatory changes.

Training and documentation

Provide training to users who will be working with an LDT and how the models can be used effectively. Detailed documentation should be made available to aid users in utilizing the model. These should be registered in the modal usage guidelines.

5. Conclusion

This report explains how an LDT Toolbox is designed and details the technical specifications for the prioritised BBs. This will be the starting point for the development of an EU LDT Toolbox in the upcoming procurement. Based on the purpose and key features which an LDT Toolbox needs to fulfil, our proposal for the delivery solution of the Toolbox is to make it available to the EU communities as a platform business model. There is a recommended set of requirements that should be considered for the deployment phase, namely:

1. Design driven by Local Communities requirements and by European priorities and values.
2. Tools are co-created by Local Communities, Industry, and Academia.
3. Generated IP is managed in a way to ensure core tools can continue to be available on an Open-Source Basis.
4. Involvement of an active community in the EU LDT Toolbox steering, maintaining, and developing.
5. Effective governance processes are put in place.
6. Long-term Self-sustainability of an EU LDT Toolbox.
7. MIMs Plus compliant and Standards based EU LDT Toolbox.

This report makes a proposal for the **design of a European LDT Toolbox** with a detailed description of capabilities, requirements, and proposed standards for the advanced Building Blocks (BBs) to accelerate more advanced cities and communities in the LDT development to evolve to higher ambition levels. The reference architecture for a LDT is explained to indicate how the different BBs work together and interact in order to deliver the required capabilities that are needed for a certain ambition level.

The overall LDT Toolbox design will, besides technical building blocks also contain strategic guidelines and tooling to help cities and communities in accelerating the development of LDTs. An essential requirement for the successful deployment of the LDT Toolbox and its BBs is the need for standardisation and compliance with interoperability standards. The objective of the LDT Toolbox is to achieve horizontal interoperability so that LDTs can interoperate on a horizontal scaling and vertical interoperability within the broader scope of digital solutions that are developed in the overall EU ecosystem as for instance the DS4SSCC and data spaces initiatives.

The selected BBs and description of the LDT reference architecture highlight the **importance of abstraction** to manage interoperability in a **system of systems approach** and assures that proposed tools can be re-used and connect in the end specific instances of Digital Twins and can operate together with other digital BBs defined in the EU projects. An LDT Toolbox can thus focus on deployment of BBs that deliver specific added value for the LDT implementations. The deployment of the toolbox and its BBs does contain certain risks and challenges that are further detailed in the overall roadmap proposal for the deployment of the LDT Toolbox (Deliverable 5.03). In general, it must be noted for all BBs that there is currently a lack of standardisation for specific data and model requirements and APIs for the defined BBs. In addition, the overall interoperability requirements definition (MIMs, data spaces, etc.) is still ongoing. The challenge is to not only procure software tools but that the specification of the BBs themselves can become the new de facto standards for LDTs.

Accordingly, the link between the ambition and the (preferred) deployment scenario(s) lies in utilising the insights gained from analysing cities and their level of progress. This information can be harnessed to support their development and guide them progressively towards the next ambition level. At the start of the deployment, it is important to notice that the overall view of a coherent LDT system design is specified and standardisation on the LDT development practices are made clear, so a city or community has a good understanding of the complete view on developing their LDT project.

6. Annexes

Annex 6.1 Non-prioritised Building Blocks- high level description

The setup of a LDT requires a coherent setup of different technical BBs that is explained in Building Block 01: LDT reference architecture.This annex describes the technical BBs for which no detailed technical specifications are provided in Section 3 of this document.

Table 73: Non-Prioritised Building Blocks

BB Name	Category	Description
Access Manager	Security	A BB that provides the tools to dynamically restrict access to different functionalities, inputs and outputs of the LDT. This includes but is not limited to API's, UI's, data sets, configurations, etc. Can consist of "simple" access management methods such as Role Based Access Managements or more advanced methods such as Policy Based Access Management which is common in Data Spaces *Also refer to SIMPL BB "Access Control & Trust" from* https://ec.europa.eu/newsroom/dae/redirection/document/86727
Analytical Visualisation Components	Visualisation and UX	A BB that provides analytical visualisations of the data available in the LDT to the stakeholders of the system. Dashboard collection library or components. For visualisation of necessary KPIs related to the output of an LDT, a frontend (web-based), backend, web publisher and map tile generator are software components that constitute abilities to visualise and contextualise information via an LDT. The web client enables the user to interact with the content and services of the LDT through a graphical user interface. Its main goal is to provide a seamless and user-friendly experience for accessing information on current and future scenario's generated through the interaction between data and models. *Also refer to SIMPL BB "Data Processing- data visualisation" from* https://ec.europa.eu/newsroom/dae/redirection/document/86727
Community management tools	Visualisation and UX	Tools & practices for building up a community and communicating with it. This includes blog posts, social media posts, chat groups, content sharing, etc.
Context Broker	Data Services	A Context Broker is an API that can integrate data from multiple systems, creating a holistic view of information by describing several types of data and providing an interface to view and interpret said data. It is also a gateway for performing actuations.
Data Catalogue	Data Services	A Data Catalogue is a list of available datasets. Essential features of a data catalogue include searching, browsing metadata, license information and access details to the datasets themselves. The data catalogue can also manage the training data sets. *Also refer to SIMPL BB "Data Discovery" from* https://ec.europa.eu/newsroom/dae/redirection/document/86727

Data space Connector	Data components	A dataspace connector enables data integration, data exchange, or interoperability between different data sources and services, without requiring data to be moved or copied into a central repository first. *Also refer to IDS data space connectors from https://internationaldataspaces.org/offers/ids-components/*
Data space connector: data service	Data components	Data service components of a data space connector enable the integration of data services into the connector. This can be done at multiple levels, depending on the needs.
Data Space Connector: model service	Data components	A model service component for dataspaces allows to publish a model/algorithm as a service into a data space connector.
Encryption tools	Security	Encryption tools allow for the secure exchange of digital assets and safeguarding the integrity of both data and other assets. Aside from that, encryption tools can be used in the context of privacy preservation, using it to conceal identity-sensitive fields in the data for instance. *Also refer to SIMPL BB "Security" from* https://ec.europa.eu/newsroom/dae/redirection/document/86727
Event Based Data Publishing Service	Data components	A service that allows to publish data sources as a series of update events or version objects to enable full data time travel capabilities.
Failover	Security	Failover infrastructure plays a critical role in Digital Twins when used in an operational context where high availability is required. Especially when LDT technology is used to control real-world assets, exceptional care needs to be taken towards ensuring high availability and operational continuity.
Geospatial Visualisation	Visualisation and UX	A BB that provides the tools to create geospatial visualisations which focus on the relationship between data and its physical location
Identity Manager	Security	A BB that provides the tools to uniquely identify and authenticate stakeholders in the LDT ecosystem, such as users, organisations, models, etc *Also refer to SIMPL BB "Access Control & Trust" from* https://ec.europa.eu/newsroom/dae/redirection/document/86727
IoT Agent	Data components	Component responsible for receiving all messages from IoT devices and sensors and distributing the across subscribed consumers. Can include IoT device management features as well.
IoT and Edge	IoT & Edge	IoT and edge infrastructure required for sensing, monitoring and controlling the urban environment. This should discuss some patterns and concerns regarding the deployment and dependencies of these components.

Security & Risk Assessment Tool	Security	Security assessment tooling as a web-based questionnaire that can lead to a risk score.
User Manager	Security	A BB that provides tools for user management, aka the management of accounts linked to specific identities and access policies. *Also refer to SIMPL BB "Access Control & Trust" from* https://ec.europa.eu/newsroom/dae/redirection/document/86727

Annex 6.2 Deep dive on LDT Capabilities

Table 73 provides a detailed description of the identified capabilities resulting from the analysis of stakeholders' needs and requirements. Although it includes functional, strategy and legal capabilities, the design of the Toolbox will only focus on the technical and functional capabilities. In fact, legal and strategic capabilities are either being addressed in other DEP-funded projects (e.g., DS4SSCC; CitCom.ai) or will be analysed in two upcoming DEP-funded LDT projects to "Scale the deployment of an enabling digital infrastructure" and "Increase the Awareness and Readiness of EU Communities", as part of a tender to "Advance Initial Stages for the Transformation of Smart Communities".

Table 74: Details of Capabilities

Domain	**Domain-type**	**Capability** (Nr.|Name)	**Capability Description**
Functional	Data	1|(meta)data schema management	The ability to manage data schemes and their usage coming from different data models inside the Digital Twins' pipeline.
Functional	Data	2|Ontology management	The ability to register and search knowledge graphs and ontologies.
Functional	Data	3|Ingest data	The ability to ingest data from a variety of sources including but not limited to historical data, IoT and event sources, streaming data sources using proper adapters. This includes dataspace connectors.
Functional	Data	4|Streaming data	The ability to transfer of large volumes of data continuously and incrementally between a source and a destination without having to access all data at the same time.
Functional	Data	5|Data replication	The ability to replicate and store data (historical, streaming) in proximity to other Digital Twin components. Storage can be done in fit-for-purpose systems such as Context brokers, Historical databases, WFS services.
Functional	Data	6|Data transformation	The ability to convert data types and properties through cleaning, structuring and enriching raw data to make if suitable for further processing and analytics.
Functional	Data	7|(Near) Real-time data processing	The ability to manage and act on the captured data with minimal latency.

Domain	Domain-type	Capability (Nr.|Name)	Capability Description
Functional	Data	8|Data, publishing and subscribing	The ability to publish data on the one hand and subscribe to updates on the other in a generic and scalable way.
Functional	Data	9|Synthetic data generation	The ability to generate synthetic data based on patterns and rules in existing sources
Functional	Data	10|Data publishing	The ability to publish data resulting from experiments through a service or an API.
Functional	Data	11|Data source management	The ability to manage data sources, referring to formats, schemas and or ontologies they conform to as well as metadata regarding (non-exhaustive) the lineage, quality, nature and intended use of the data.
Functional	Data	12|Data time travel	The ability to have a full historic trace of the data and context data to ensure repeatability of experiments.
Functional	Data	13|Data processing	The ability to process large volume of data.
Functional	Data	14|Data Storage	The ability to store large volumes of data in a fit-for-purpose data store.
Functional	Analytics	15|Prediction	The ability to estimate that a specified event will happen in the future or will be a consequence of other events.
Functional	Analytics	16|Simulation	The ability to create approximate imitation of a process or a system using past historical information, physical models, video, audio, and animation, what-if-scenarios.
Functional	Analytics	17|Algorithms & Models	The ability to perform mathematical and statical calculations to enable physics-based and other mathematical models
Functional	Analytics	18|Reporting (general)	The ability to generate configurable and customisable reports to get insights into the data. This includes the ability to report on cases and scenarios and their results, sharing them with stakeholders and/or the broader public.
Functional	Analytics	19|Algorithm and model management	The ability to store, manage and retrieve the algorithmic codebase, business rules and metadata that describe a simulation model.
Functional	Analytics	20|Federated Learning and Training	The ability to train (AI) models in a federated way, typically with privacy concerns in mind.
Functional	Analytics	21|Machine Learning and AI	The ability to host and run machine learning and AI models.
Functional	Visualisation	22|Basic Visualisation	The ability to graphically or parametrically (that is, through parameters and values) visualise data through simple charts, graphs, simple dashboards, tables, hierarchical and basic 3D views of the assets. This includes dashboards.

Domain	Domain-type	Capability (Nr.|Name)	Capability Description
Functional	Visualisation	23|Advanced Visualisation and Geo dashboarding	The ability to graphically or parametrically (that is, through parameters and values), visualise data through complex charts and graphs, dashboards fetching raw and process data from multiple systems, complex 3D models and animations. This includes the ability to create dashboards visualisations with overlayed data from different systems.
Functional	Visualisation	24|Virtual Reality	The ability to provide a simulated experience that can be similar to, or completely different from, the real world.
Functional	Visualisation	25|Augmented Reality	The ability to provide an interactive experience of a real-world environment where the objects that reside in the real world are enhanced by computer-generated perceptual information such as visual, auditory, haptic etc., environment.
Functional	Visualisation	26|Integrated user experience	The provisioning of an integrated and consistent UI for all components of the Digital Twin.
Functional	Visualisation	27|Alerts and Notification	The ability to display and manage alerts, messages, message queues, triggers, and notifications
Functional	Integration	28|Model abstraction	The LDT needs to provide a clear interaction scenario for algorithm suppliers to plug into, so that e.g., the MIM5 capabilities above can be delivered. This capability also implies the ability to decouple models from their data and configuration to enable interchanging data sources, properly parameterise the models and compose models.
Functional	Integration	29|Model hosting	The capability of hosting a model as a reusable asset or as a service.
Functional	Integration	30|Data flows and component orchestration	The ability to coordinate the dataflows, interaction and models in a Digital Twin.
Functional	Integration	31|Interaction support	The capability of interacting with the environment (2D, 3D, VR or Analytical) to create new hypothetical situations in the virtual world that may impact the outcome of a scenario/experiment
Functional	Integration	32|Case, scenario and experiments management	The ability to create, manage and share LDT cases. This includes creating different scenarios in each case and creating, running and keeping track of experiments and their outcomes within these scenarios.

Domain	Domain-type	Capability (Nr.|Name)	Capability Description
Functional	IoT	33|Context information management	Context information management ensures comprehensive and integrated access, use, sharing, and management of data across different solutions and purposes. It manages the context information coming from IoT devices and other public and confidential data sources providing cross cutting context data and access through a uniform interface.
Functional	IoT	34|Supervised or unsupervised actuation, command and control	The ability to manipulate real-world systems/assets in response to the outcome of a model or decision taken in a Digital Twin.
Functional	IoT	35|Urban measuring, sensing and control	The ability to sense, measure and control the urban environment.
Functional	Security	36|Privacy-sensitive data interaction	The Digital Twin needs to be able to interact with consent-driven personal data stores when using personal data within the Digital Twin. In that sense, it can be seen as an end-user of personal data for a certain purpose and thus needs to be able to interact as such.
Functional	Security	37|User and Access Management	The ability to invite users, manage users, define and manage user roles as well as the possibility to control access to the different key assets by the administrator as well as the users that own these assets.
Functional	Security	38|Encrypted asset exchange	Cross-layer asset encryption for secure exchange.
Functional	Security	39|Reliability and resilience	The ability to operate at high reliability/SLA levels.
Strategic		40|Marketplace interaction	Provide a way to interact with marketplaces / data spaces.
Strategic		41|Citizen engagement and Case and scenario feedback gathering	The ability to capture feedback from stakeholders in the context of an LDT case.
Strategic		42|Collaboration and community management	The ability to interact with Communities around Europe to share experience, practices, requiring advice and ask for feedbacks. The ability to collect knowledge base and provide Co-pilot experience.
Strategic		43|Reference case management	The ability to build a knowledge base for different use like dissemination, and inspiration.
Strategic		44|Technical Solutions Catalogue Management and Benchmarking	The ability to benchmark solutions using evaluation framework like SonarQube and/or with a defined set of performance KPI. Ability to perform GAP analysis

Domain	Domain-type	Capability (Nr.|Name)	Capability Description
Strategic		45|LDT Roadmap Management	The ability to define a roadmap for LDT deployment in function of the ambitions and the capabilities of the local authority.
Legal		46|Procedural transparency	The ability to disclose to different stakeholders the type of choices made, parties involved, risks and mitigation actions when using data and algorithm assets.
Legal		47|Technical transparency and explainability	The ability to disclose to different (technical) stakeholders the owner, signature, provenance, versioning, of the assets uses in decisions and to disclose how algorithms and data are used and chained to come to a result from a technical perspective (how does it work).
Legal		48|Usage context information provisioning	Provide descriptions in what context the Digital Twin assets are used (in essence, couple Digital Twin scenarios to the transparency items mentioned above).
Legal		49|Accountability information provisioning	As LDTs are used in the public domain, provide a way to describe how used assets are validated, (publicly) reviewed, respect human digital rights, sovereignty, respect regulations.
Legal		50|Security and Risk assessment	The ability to protected Digital Twins from unintended or unauthorised access, change or destruction. Security concerns equipment, systems and information, ensuring availability, integrity and confidentiality of information.
Legal		51|Quality Assurance	The ability to manage the quality for LDT solutions and facilitate compliance.
Legal		52|Transaction management with private actors	Create awareness in public stakeholders so that they can contractually ask for the right things in public procurements and in contract negotiations.
Legal		53|Public procurement	Create standardised fair, reasonable and non-discriminatory terms of reference and conditions for local administrations that procure private ICT solutions for use in projects involving LDT.
Legal		54|Data management and compliance	The ability to perform mature data management with a focus on general legal compliance, data privacy, data licensing, data usage, accountability.
Legal		55|Model governance and compliance	The support for governance and certification of models, algorithms and other assets.
Legal		56|Semantic governance and compliance	The support for governing semantics and ontologies across LDT deployments and the ability to test conformance of the use of proper ontologies in asset publishing.

ISBN: 978-92-68-06527-3

Publications Office
of the European Union