European data space
for smart communities

# Deliverable 3.4

# DS4SSCC Deliverable 3.4 Models validated

WP 3 – Model usage phase

**Authors:** Corné Helmons (Rotterdam)
**Reviewers:** Niek Verschure ('s-Hertogenbosch), Anni Joela (Tampere), Margriet Visscher (Future Insight)
**Delivery date:** 7 July 2026 (revised 29 September 2026)
**Dissemination level:** Sensitive
**Type:** Report

Funded by the European Union

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or of the granting authority. Neither the European Union nor the granting authority can be held responsible for them

# Revision History

| Author Name, Partner short name | Description | Date |
|---|---|---|
| Corné Helmons, Rotterdam | City Project Manager | 7-July-2026 |
| Corné Helmons, Rotterdam | Revision v1.1 after OASC review: validation evidence per model added (Ch. 5), registration chapter condensed with cross-references (Ch. 6), Figure 2 redrawn from actual registrations | 29-September-2026 |

Abbreviations

| Abbreviation | Definition | Abbreviation | Definition |
|---|---|---|---|
| DS4SSCC | European data space for smart communities (DS4SSCC) | DT-aCoGo | Digital Twinning as Common Good |
| OASC | Open & Agile Smart Cities & Communities | OUP | Open Urban Platform |
| WP | Work Package | PC | Pilot Coordinator |
| BIM | Building Information Modeling | WFS | Web Feature Service |
| OGC | Open Geospatial Consortium | ETL | Extract, Transform, Load |
| LOD | Level of Detail | UMG | Unreal Motion Graphics |
| REST | Representational State Transfer | | |
| API | Application Programming Interface | CIM | Common Information Model |
| CRS | Coordinate Reference System | DPRA | Delta Plan on Spatial Adaptation (Deltaplan Ruimtelijke Adaptatie) |
| JWT | JSON Web Token | NAP | Normaal Amsterdams Peil (Dutch height reference) |
| PET | Physiological Equivalent Temperature | PPB | Purple Polar Bear |
| TPG | The People Group | WMS | Web Map Service |
| LGN | Landelijk Grondgebruiksbestand Nederland (national land-use dataset) | WUR | Wageningen University & Research |

# Table of contents

1. Executive Summary ..... 5
2. Introduction ..... 6
3. Problem definition ..... 8
3.1 Consistency between data services and model execution ..... 8
3.2 Limited transparency in API-based model access ..... 8
3.3 Dependence on platform-specific execution environments ..... 9
3.4 Risk of inconsistent reuse across cities ..... 9
3.5 Need for integrated validation within the model execution workflow ..... 9
4. Validation approach ..... 10
4.1 Expert-driven validation in the platform environments ..... 10
4.2 Validation checks ..... 12
4.3 Outcomes of a check ..... 12
5. Validation results per model ..... 13
5.1 Rana – flood model (Nelen & Schuurmans) ..... 13
5.2 DatMobility – traffic model (Goudappel) ..... 15
5.3 Tygron – heat stress model ..... 17
5.4 Summary across models ..... 20
6. Exposure of the validated models in the OUP ..... 22
7. Conclusions and next steps ..... 23
7.1 Conclusions ..... 23
7.2 Remaining limitations and recommendations ..... 23
7.3 Next steps ..... 24
Annex A – Validation evidence documents ..... 25

# 1. Executive Summary

This deliverable reports the validation of the three analytical models used in the DT-aCoGo digital twin ecosystem: Rana (flood modelling, Nelen & Schuurmans), DatMobility (traffic modelling, Goudappel) and Tygron (heat stress). It builds on the model execution workflow established in Deliverables 3.1–3.3, in which CIM-aligned datasets are retrieved from the Open Urban Platform (OUP), processed in platform-specific modelling environments and exposed as reusable services through APIs.

In this deliverable, validation means checking that this chain remains consistent from input to output. Four types of checks were applied to each model: **(V1)** interpretation of the input data, **(V2)** spatial and temporal consistency of the results with the Digital Twin, **(V3)** exposure and access through the OUP hub structure, and **(V4)** plausibility of the model output. The main test focus was the (semi-)automated data delivery supported by the CIM and the OUP, which determines whether analyses can be scaled across projects and cities; output plausibility rests on several years of use of the same models in earlier Rotterdam projects. The checks were carried out by the model providers in their native expert environments during the integration work and are documented in their technical reports (Annex A).

All three models passed the tests of (semi-)automated data delivery and run end-to-end within the ecosystem. The validation also surfaced concrete issues that were resolved or bypassed with documented procedures. Examples are height offsets that arise when Rana flood results are converted from the Dutch RD/NAP reference system to the geocentric system of 3D Tiles; missing fields in the DatMobility scenario input, which are estimated or ignored; and a single injected credential per Tygron API registration, which was resolved with a separate registration per city hub. Chapter 5 documents each check with its finding, resolution and remaining limitation.

The validation also identifies limitations that remain at the end of the project. None of the providers can yet register each analysis result in the OUP automatically (a workaround is in place: results called from the Unreal Digital Twin are always served through the same data-service link, while the underlying dataset is replaced), protected OUP data sources cannot yet be consumed directly by the model platforms, and cities cannot inspect the DatMobility traffic model themselves without a separate contract. Because the first two recur across providers, they are addressed as recommendations to the OUP rather than to the individual models (Chapter 7).

API registration and hub governance are described in detail in D3.1 and D3.3; this deliverable only summarises the registrations as they were validated (Chapter 6). The integration of the validated services into the Unreal Engine visualisation environment is addressed in Deliverable 3.5.

# 2. Introduction

The DT-aCoGo project aims to establish a scalable and interoperable digital twin ecosystem in which data, analytical models, and visualisation environments are connected through a federated architecture. As described in previous deliverables, the Open Urban Platform (OUP) serves as the central data space, enabling datasets and services to be shared across stakeholders, while analytical models are executed within dedicated domain-specific platforms. A schematic description is shown in Figure 1 below:

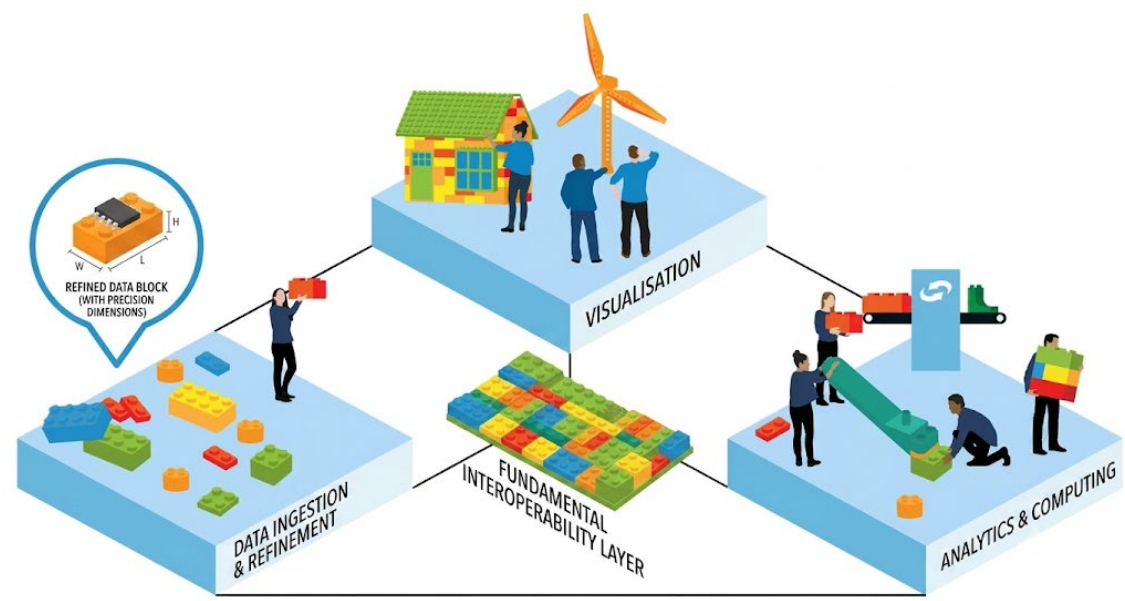



*Figure 1: Schematic architecture DT-aCoGo*

The foundations of this ecosystem have been established in earlier deliverables within Work Package 3:

- Deliverable 3.1 defined the interoperability framework between OUP and sectoral analytical models, including the use of a Common Information Model (CIM) to ensure semantic alignment of heterogeneous datasets.
- Deliverable 3.2 validated lifecycle-based temporal filtering, establishing time as an intrinsic property of the data through attributes such as *ValidFrom* and *ValidTo*, ensuring consistency between analysis and visualisation.
- Deliverable 3.3 operationalised the modelling workflow by defining how datasets are **prepared, validated, and published as authoritative data services in OUP**, and how analytical models consume these services and expose their functionality through standardised APIs.

Within this framework, a **model execution workflow** is established in which CIM-aligned datasets are retrieved from OUP, processed within platform-specific analytical environments, and returned as validated outputs to the digital twin ecosystem. This ensures that data, analysis, and visualisation remain consistently aligned across all components.

Where Deliverable 3.3 focused on the preparation and delivery of model input data and the finalisation of the Digital Twin as an authoritative data state, this deliverable addresses the subsequent phase: the validation of analytical models. In this context, validation refers not only to verifying the correctness of model outputs, but also to ensuring **consistency between model inputs, analytical processes, and resulting data services**.

The validation process is closely linked to the architectural principle of separating data, analysis, and visualisation. Models consume **CIM-aligned data services from OUP**, execute calculations within their native platforms, and return results that are published as reusable services. These services are subsequently registered and exposed as APIs, enabling controlled reuse within the federated hub structure.

By embedding validation within this workflow, the DT-aCoGo approach ensures that analytical models are not treated as isolated components, but as integral parts of a **coherent digital twin system**, in which lifecycle consistency, API standardisation, and data governance are maintained across all stages.

Compared with the first version of this deliverable (July 2026), this revision adds the validation evidence per model (Chapter 5), based on the technical reports delivered by the model providers (Annex A). It also condenses the description of API registration into a summary with references to D3.1 and D3.3 (Chapter 6), and replaces Figure 2 with a diagram drawn from the actual registrations in the OUP.

The deliverable is structured as follows. Chapter 3 defines the validation problem. Chapter 4 describes the validation approach and the four types of checks applied. Chapter 5 presents the validation results per model. Chapter 6 summarises how the validated models are exposed through the OUP. Chapter 7 presents conclusions, remaining limitations and next steps.

# 3. Problem definition

Within the DT-aCoGo ecosystem, the integration and deployment of analytical models builds upon a foundation in which datasets are already semantically aligned through the Common Information Model (CIM) and published as authoritative data services within the Open Urban Platform (OUP).

While this significantly reduces the complexity of data integration, challenges remain in ensuring that analytical models operate consistently and reliably within the established model execution workflow.

## 3.1 Consistency between data services and model execution

Although datasets are harmonised and made available as CIM-aligned services, analytical models interpret and process this data within their own platform-specific environments. Differences in internal model logic, parameterisation, and execution mechanisms can lead to variations in results, even when the same input data services are used.

This creates a challenge in maintaining consistency between data, analysis, and resulting data services, which is a core principle of the DT-aCoGo architecture.

Without a structured validation approach, discrepancies between expected and actual outcomes may remain hidden once models are exposed through APIs.

## 3.2 Limited transparency in API-based model access

Within the DT-aCoGo framework, analytical models are exposed through standardised APIs registered in OUP. While this enables scalability and reuse, it also abstracts the internal workings of the models.

From the perspective of the digital twin ecosystem:

- Models are accessed as black-box services
- Only input parameters and output results are visible
- Intermediate calculation steps and model assumptions remain internal to the platform

As a result, API-based interaction alone cannot establish whether model outputs are consistent with the authoritative data state of the Digital Twin, or whether deviations are caused by data quality, parameter settings or model logic. Validation therefore has to take place within the native modelling environments, and not exclusively at API level.

## 3.3 Dependence on platform-specific execution environments

A key characteristic of the DT-aCoGo ecosystem is that analytical models are executed within platform-specific environments (e.g. flood modelling, traffic modelling, heat stress analysis), while OUP functions as a data space and governance layer rather than an execution environment.

While this separation ensures flexibility and modularity, it introduces dependencies:

- Each platform has its own configuration logic and execution workflow
- Model behaviour can vary depending on platform-specific settings
- Validation mechanisms differ across platforms

This heterogeneity makes it difficult to define a single, centralised validation approach across all models.

## 3.4 Risk of inconsistent reuse across cities

The DT-aCoGo architecture enables the reuse of analytical models across multiple cities through the federated hub structure. However, differences in:

- data completeness and quality
- spatial resolution and level of detail
- temporal scenario definitions (based on lifecycle attributes)

can lead to variations in model outcomes when the same model is applied in different contexts.

Even when data is aligned through the CIM, these contextual differences may affect model behaviour. Without proper validation, models may produce results that are technically correct within the platform but inconsistent with the intended Digital Twin scenario.

## 3.5 Need for integrated validation within the model execution workflow

The challenges described above demonstrate that validation cannot be treated as a separate or external process. Instead, it must be embedded within the end-to-end model execution workflow, as defined in Deliverable 3.3.

Specifically, validation must ensure that:

- CIM-aligned input data services are correctly interpreted by the models
- analytical processes produce results consistent with the defined temporal and spatial context
- outputs can be published as reliable data services and exposed via APIs
- the full chain—data → model → result → API—remains coherent

This leads to the central problem addressed in this deliverable:
How can analytical models be validated in a way that ensures consistency between CIM-aligned data services, platform-specific model execution, and API-based integration within the DT-aCoGo digital twin ecosystem?

To make the answer verifiable, Chapter 4 translates this question into four concrete types of checks, and Chapter 5 reports the outcome of these checks for each of the three models.

# 4. Validation approach

Validation in DT-aCoGo is carried out where model behaviour can be fully inspected: in the native modelling platforms, by the domain experts of the model providers, working with harmonised input data from the OUP. Validation covers the complete chain shown in Figure 2 (Section 4.1): from input data in the common model, through model execution and registration in the provider hubs, to consumption by the city hubs and the Unreal Digital Twin.

## 4.1 Expert-driven validation in the platform environments

Each model provider validates its model in its own expert environment, where input data, parameters and intermediate results are fully accessible:

- **Rana** (Nelen & Schuurmans): a project environment per scenario (for example Merwehaven) with a fixed folder structure for source, before, during and after data, flood models and results, and documented processes to prepare the elevation model, the friction and infiltration rasters, build the 2D flood model and simulate rainfall.
- **DatMobility** (Goudappel): the municipal traffic model in OmniTRANS, maintained by Goudappel. This expert environment is only available to municipalities with a maintenance contract with Goudappel; within DT-aCoGo, model-level validation was therefore performed by Goudappel.
- **Tygron**: the Tygron Platform client application, in which the Digital Twin sessions created through the Proxy API can be opened to inspect the Heat Stress Overlay, the imported data and the future-plan Measures.

Deviations found in these environments can originate from the interpretation of the CIM-aligned input data, from parameter or scenario settings, or from differences in platform-specific execution logic. They are analysed and resolved iteratively before model functionality is exposed through the OUP.

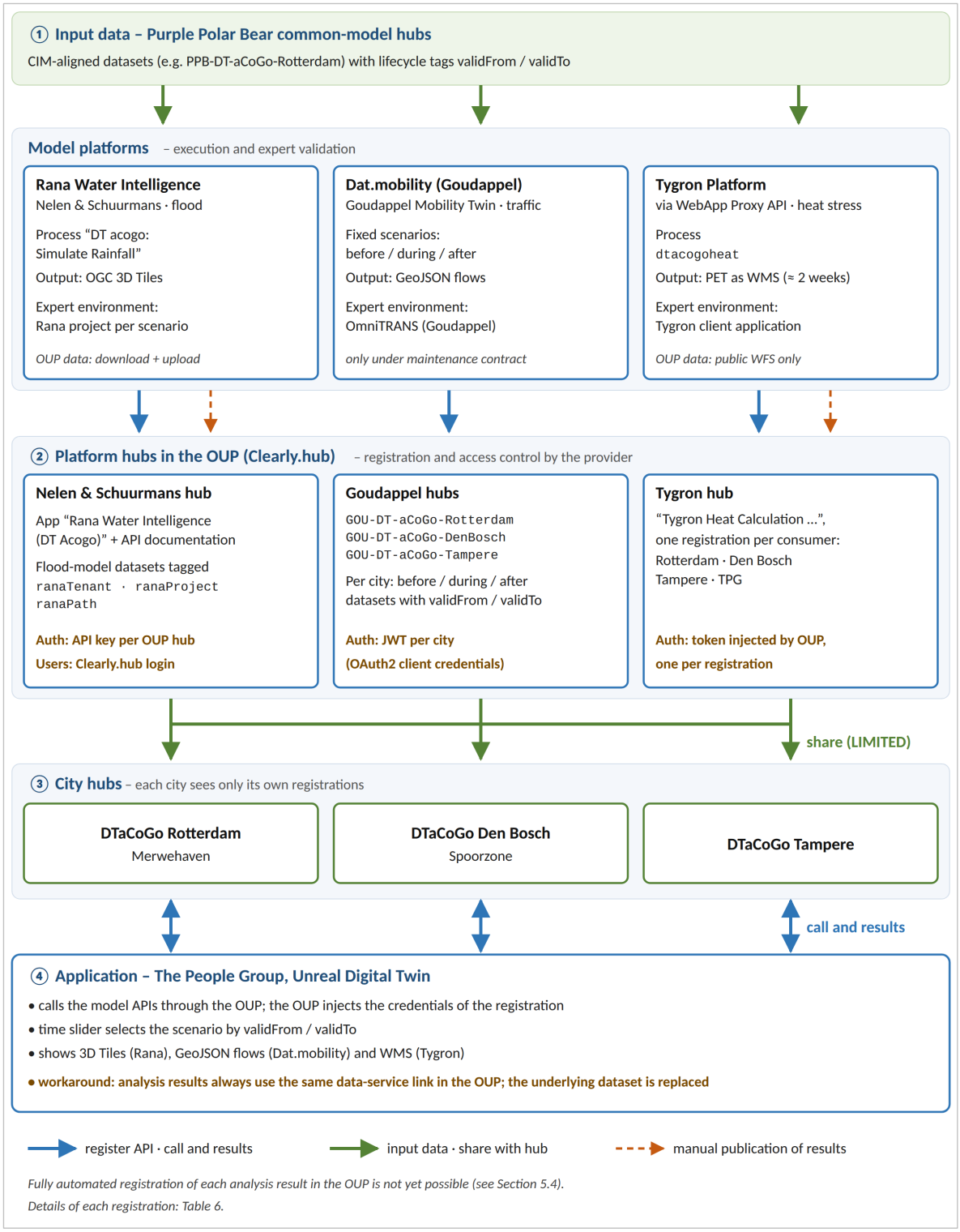



*Figure 2: API flow of the analytical models as implemented and validated, drawn from the registrations documented by the model providers (Annex A)*

Figure 2 has been checked and validated by the technical leads of the Municipality of Rotterdam and Future Insight.

## 4.2 Validation checks

To make validation verifiable and comparable across heterogeneous models, four types of checks are distinguished (Table 1). Checks V1–V3 address the consistency of the data–model–service chain; check V4 addresses the plausibility of the model output itself.

The main focus of testing in DT-aCoGo was the (semi-)automated data delivery supported by the CIM and the OUP (checks V1–V3), because this determines whether analyses can be scaled across projects and cities. Output plausibility (V4) was not re-established from scratch: Rotterdam has worked with all three model providers for several years in earlier projects, in which the model outcomes proved accurate and valuable for real project use, each time with manual, project-based data preparation. DT-aCoGo was the first time the input data was delivered through the OUP. V4 therefore rests on that track record and on the assessment of the models in D3.6.

*Table 1: Validation checks applied to each model*

| ID | Check | Question | Evidence used |
|---|---|---|---|
| **V1** | Input interpretation | Are the CIM-aligned inputs from the OUP interpreted by the model as intended (availability, format, attributes, units, coordinate system)? | Data requirement specifications, input contracts, dataset checks |
| **V2** | Spatial and temporal consistency | Are results positioned correctly in space (coordinate system, height reference) and in time (scenario selected through validFrom / validTo)? | Coordinate system agreements, scenario tags, time-slider behaviour |
| **V3** | Exposure and access | Can model functions and results be called and retrieved end-to-end through the OUP, by the intended hub only? | Registrations, authentication configuration, documented runs |
| **V4** | Output plausibility | Are the model outcomes accurate and valuable for real project use? | Track record in earlier Rotterdam projects; ethical and functional assessment of the models (D3.6) |

## 4.3 Outcomes of a check

Each check results in one of the following outcomes, which are used in Chapter 5:

- **Confirmed** – the check was passed without changes.
- **Resolved** – an issue was found and fixed in the model configuration, the integration or the agreements between parties.
- **Workaround** – an issue was found and bypassed with a documented procedure; the underlying cause remains.
- **Open** – a limitation that remains at the end of the project and is carried forward as a recommendation (Chapter 7).

# 5. Validation results per model

This chapter reports, for each model, what was checked, what was found and how it was resolved. The findings are taken from the technical reports delivered by the model providers in June 2026 (Annex A) and from the configuration of the OUP hubs. The output plausibility check (V4) is reported at the end of each section.

## 5.1 Rana – flood model (Nelen & Schuurmans)

Rana is the most fully integrated of the three models. Flood models are built in the municipality's own Rana environment from common-model data, and rain simulations can be started from the Unreal Digital Twin through the Rana API. Rotterdam (Merwehaven) serves as the reference case.

*Table 2: Validation findings for the Rana flood model*

| Check | What was checked | Finding | Resolution | Status |
|---|---|---|---|---|
| V1 | Availability of common-model inputs for the Merwehaven scenario | The dataset check in the common-model page of Nelen & Schuurmans (Figure 3) showed three of five inputs available as a service. The physical land-use map was not active, and the elevation map had been delivered directly, without a link. | Elevation map supplied directly to Rana. Land use derived from the national land-use dataset LGN, produced and maintained by Wageningen Environmental Research (WUR); for the future situation the same processing rules were applied to the underlying datasets. | **Resolved** |
| V1 | Modelling requirements | 2D hydrodynamic modelling requires a projected coordinate system in metres (not degrees), rasters of 0.5 × 0.5 m and explicit no-data values; land-use classes must map onto the Rana standard categories. | Requirements and conversion steps fixed in the replicable workflow “From data to flood models” (D3.3, annex). | **Resolved** |
| V1 | Retrieving OUP data directly into Rana | Harvesting OUP metadata into the Rana Catalog is not yet possible: Clearly.hub offers no secured CSW endpoint, and the Rana harvesters only support username/password authentication. Four integration options were assessed. | Data downloaded from the common model and uploaded to the Rana project. | **Workaround** |
| V2 | Temporal consistency of future scenarios | Models for the ‘during’ and ‘after’ situation may only use data valid on the scenario date, and only span the scenario area. | Scenario date applied as a filter on validFrom / validTo and merged with the ‘before’ situation (workflow “From data to flood models”). | **Resolved** |
| V2 | Position of flood results in the 3D Digital Twin | Converting results from RD (EPSG:28992) with NAP heights to the geocentric system of 3D Tiles introduces offsets that are not negligible compared to the water depth, because layers from different parties use different projection and geoid assumptions. | Consistent projection and vertical reference definitions required for all 3D layers. | **Open** |

| Check | What was checked | Finding | Resolution | Status |
|---|---|---|---|---|
| V2 | Representation of shallow water | The 10 × 10 m water surface is drawn over a much finer ground model. Where the water depth approaches zero, both surfaces coincide and cause flickering. | Three improvements identified: depth-dependent transparency based on the 0.5 m depth raster, water volume per cell, or separate static terrain and time-varying water tiles. | **Open** |
| V3 | Linking flood models to OUP datasets | The OUP cannot register individual Rana files. | Convention: flood models stored in `dt-acogo/floodmodels/` and identified through the OUP tags ranaTenant, ranaProject and ranaPath. | **Resolved** |
| V3 | End-to-end simulation from Unreal | The documented sequence of API calls (start process, poll job status, retrieve 3D Tiles) was executed from the Unreal plugin, for example on the flood model “Goudsesingel huidig” on 13 March 2026. | Presets for design events T200, T500 and T2000 defined for the user interface. | **Confirmed** |
| V3 | Publishing results back into the OUP | Automated publishing is blocked: machine-to-machine access only allows Rana to create datasets in its own hub. | Results published manually through a Rana Publication and registered in the target hub with an API key. Results of analyses called from the Unreal Digital Twin are served through a fixed data-service link whose underlying dataset is replaced. | **Workaround** |
| V3 | User login through the OUP | Clearly.hub supports OAuth2 for single-page applications, but not for back-end applications such as Rana. | Custom configuration with Future Insight; Clearly.hub added as identity provider in Rana; subscription proof verified against the Clearly.hub public key. | **Resolved** |

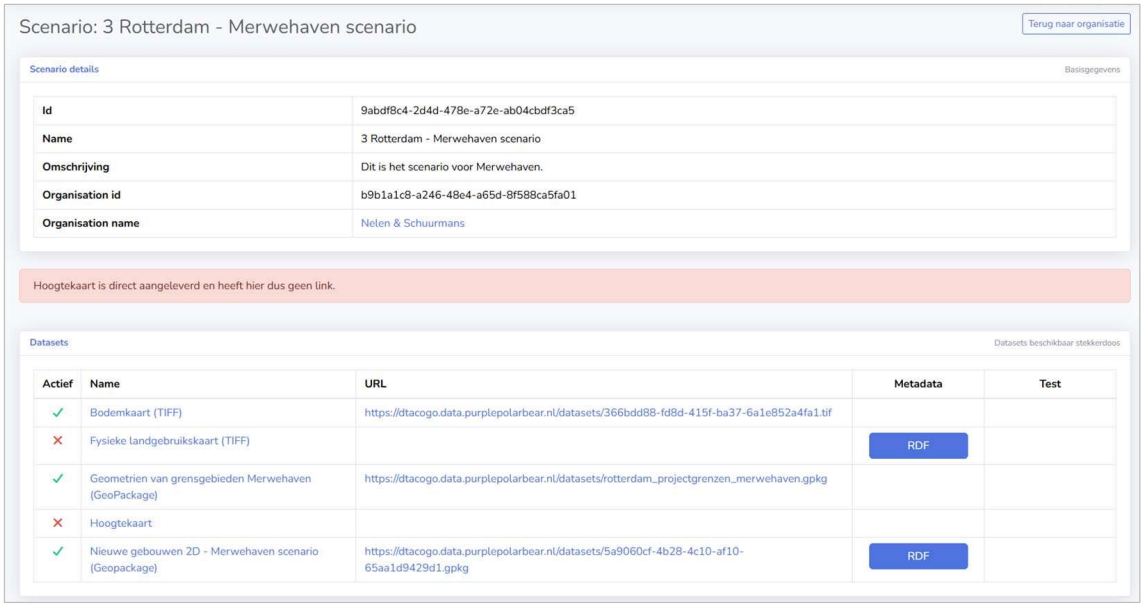



*Figure 3: Dataset check for the Merwehaven scenario in the common model: available inputs (✓), inactive land-use map and directly delivered elevation map (✗) (source: Rana, Deliverable I part 1)*

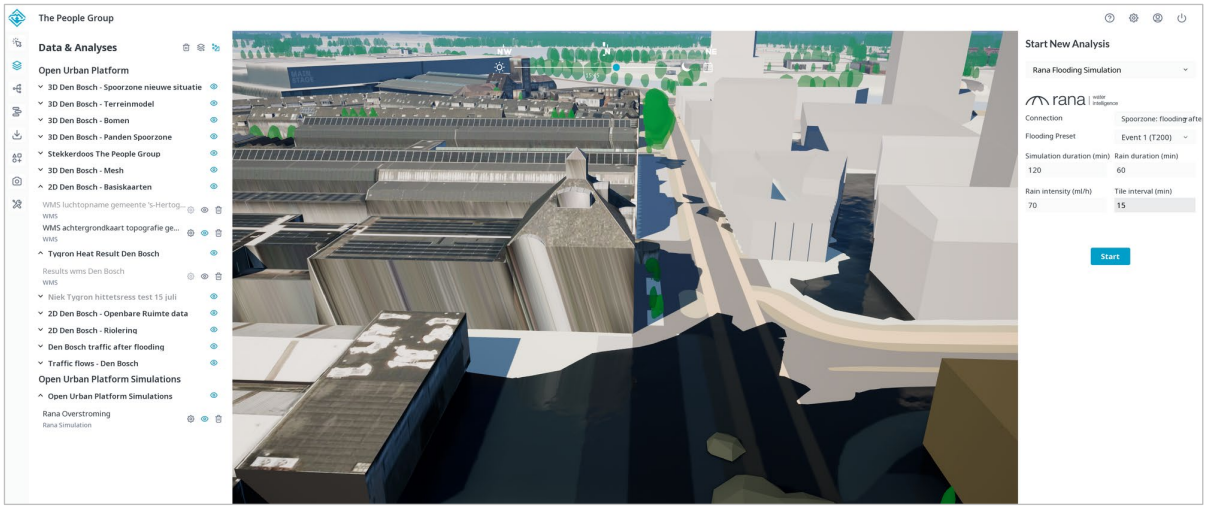



*Figure 4: Rana flood result for the Spoorzone in Den Bosch ('after' situation, design event T200: 70 mm/h for 60 minutes, 120 minutes simulated) in the Unreal Digital Twin; the result is listed as "Rana Overstroming" in the Data & Analyses panel (source: The People Group, Unreal Digital Twin)*

**Output plausibility (V4).** Confirmed on the basis of prior use. Rotterdam has applied Rana flood models in earlier projects, with manually prepared input data, where the outcomes proved accurate and valuable for real project use; the model basis, assumptions and error sources are assessed in D3.6. In DT-aCoGo the same modelling approach was fed with input data delivered through the CIM and the OUP, which passed the checks above. Figure 4 shows the result of a simulation for the Spoorzone in Den Bosch ('after' situation) with design event T200, started from and visualised in the Unreal Digital Twin: water accumulates between the buildings of the new situation, and the result is listed as "Rana Overstroming" in the Data & Analyses panel.

## 5.2 DatMobility – traffic model (Goudappel)

DatMobility is integrated through fixed scenarios. Municipalities deliver a study area and the expected changes in socio-economic data; Goudappel calculates a before, during and after scenario in its own modelling environment and publishes each scenario as a conditional-flow service (OGC API Features) in a dedicated OUP hub per city.

*Table 3: Validation findings for the DatMobility traffic model*

| Check | What was checked | Finding | Resolution | Status |
|---|---|---|---|---|
| V1 | Interpretation of the scenario input | Input contract of two datasets: the study area as valid GeoJSON (automatically extended with influence and boundary zones) and a JSON 'delta' of socio-economic data (e.g. Spoorzone: 39 new homes, 82 residents, 2,739 m² commercial space, 80 jobs, 380 public parking spaces). Missing fields are estimated where possible (e.g. residents from new homes) or ignored; the field commercial_space_unit is not used. | Fixed JSON structure agreed and published through the common model. Estimated values are not yet flagged to the user. | **Resolved** |

| Check | What was checked | Finding | Resolution | Status |
|---|---|---|---|---|
| V1 | Inspection of the traffic model | The municipal traffic model is not exchanged through the OUP, and the expert environment (OmniTRANS) is only available under a Goudappel maintenance contract. Local models are calibrated against traffic counts (D3.6). | Model-level validation performed by Goudappel. | **Open** |
| V2 | Temporal consistency of scenarios | Each city hub holds a before, during and after scenario with validFrom / validTo tags, e.g. Merwehaven 'before' valid from 2025-01-01 to 2027-01-01 (Figure 5) and Spoorzone 'after' from 2030-01-01 to 2035-01-01. | The Unreal time slider selects the scenario valid on the selected date. | **Confirmed** |
| V3 | Access isolation per city | Scenario datasets are registered with LIMITED findability (e.g. only findable by DTaCoGo Rotterdam) and a city-specific JWT authentication method (OAuth2 client credentials). | – | **Confirmed** |
| V3 | Retrieval of city-wide results | Networks for a municipality are large; retrieval in batches and filtering by bounding box are not suitable for the Digital Twin. | Retrieve with offset=0 and limit=10000 and omit the bounding box; expose only the condition parameters in the user interface. | **Resolved** |
| V3 | Conditional flows from Unreal | The Flow scaler panel in Unreal (Figure 6) calls `/collections/segments/items` with date, hour, temperature, weather and trip purpose, and plots the returned GeoJSON on the Digital Twin. | – | **Confirmed** |

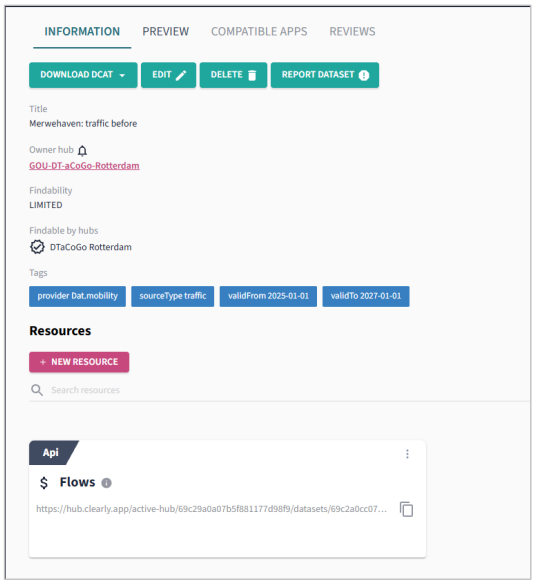



*Figure 5: Scenario dataset "Merwehaven: traffic before" in hub GOU-DT-aCoGo-Rotterdam, with LIMITED findability and validFrom / validTo tags (source: Dat.mobility guide)*




*Figure 6: Flow scaler panel in the Unreal Digital Twin, calling the DatMobility conditional-flow service (source: Dat.mobility SDK Light)*

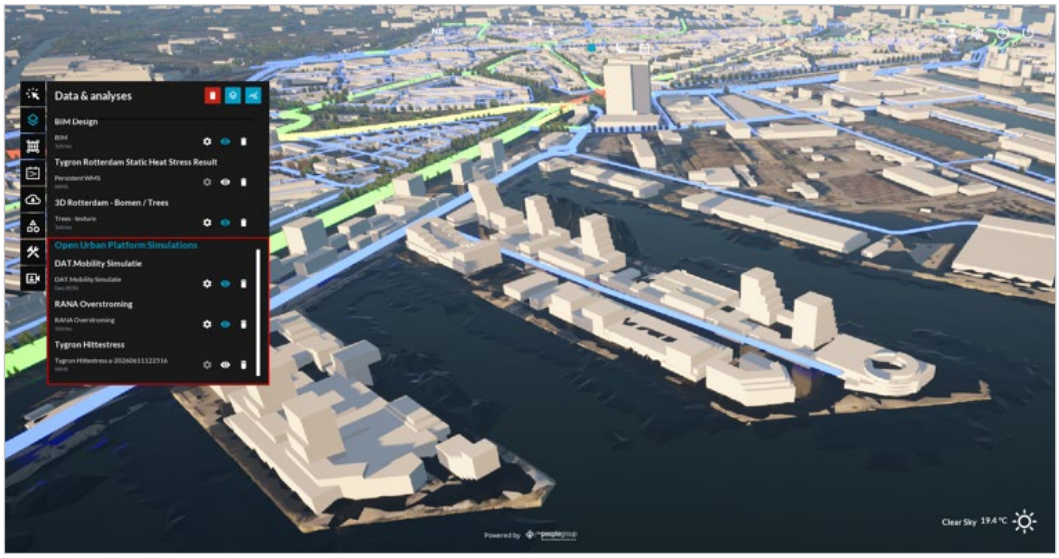



*Figure 7: Traffic flows for Merwehaven in the Unreal Digital Twin; the Data & Analyses panel lists the results of all three models: DAT.Mobility (GeoJSON), Rana (3D Tiles) and Tygron (WMS) (source: D3.5)*

**Output plausibility (V4).** Confirmed on the basis of prior use. The traffic models of Goudappel have been used by Rotterdam in earlier projects and are calibrated against traffic counts (D3.6). In DT-aCoGo the scenario input was delivered through the CIM and the OUP, which passed the checks above. Figure 7 shows the resulting traffic flows for Merwehaven, plotted on the Digital Twin.

## 5.3 Tygron – heat stress model

Tygron is connected through a Proxy API based on the open-source Tygron WebApp. A single request creates a Digital Twin session on the Tygron Platform for a configurable area (500 × 500 m by default), runs the heat stress calculation according to the DPRA methodology and returns the Physiological Equivalent Temperature (PET) as a WMS.

*Table 4: Validation findings for the Tygron heat stress model*

| Check | What was checked | Finding | Resolution | Status |
|---|---|---|---|---|
| V1 | Loading OUP data into the heat stress model | Loading data from protected OUP data sources proved infeasible. Through the Proxy API only WFS services without additional authentication can be used, such as the Rotterdam tree WFS (Figure 8). Apart from names and building types, attributes must be numeric. | Data requirements specified: trees as points or polygons with a numeric height mapped to FLOOR_HEIGHT_M; plans as non-overlapping polygons with DEFAULT_FLOORS and floor height, imported as Measures. | **Workaround** |
| V1 | Tree data from the common model | Loading tree data from the Purple Polar Bear common model was developed but not connected. | Trees loaded through a pre-configured GeoPlugin in the prepared municipal projects, or through a public WFS. | **Open** |
| V2 | Coordinate system of results | The project agreed on EPSG:4326 as common coordinate system, while Dutch partners commonly use EPSG:28992; the WMS must be requested in the correct system. | `&crs=4326` or `&crs=28992` appended to the WMS link. | **Resolved** |
| V3 | Credentials when called through the OUP | A 'limited' registration injects one credential for every caller; 'public' and 'paid' registrations are not possible because Tygron manages its own licensing. | Separate registration per consumer, each with its own token: Tygron Heat Calculation Rotterdam, Den Bosch, Tampere and TPG. The API cannot be publicly discoverable. | **Workaround** |
| V3 | Persistence of results | Results exist in a temporary session of about two weeks; afterwards the WMS link no longer works, and the OUP does not detect such broken links. | PET results exported manually to the Tygron GeoShare (Figure 9) and published as a persistent WMS, registered in the OUP as an open or protected resource. | **Workaround** |
| V3 | Automatic registration of results in the OUP | The OUP does not pass information about the calling hub or user, so the provider cannot determine where results should be registered. | Results of analyses called from the Unreal Digital Twin are served through a fixed data-service link whose underlying dataset is replaced; full automation recommended to the OUP (Chapter 7). | **Workaround** |
| V3 | API conformance | The Proxy API follows OGC API Processes but is not yet fully conformant (schemas, link references for auto-discovery). | Sufficient for live use within the project. | **Open** |

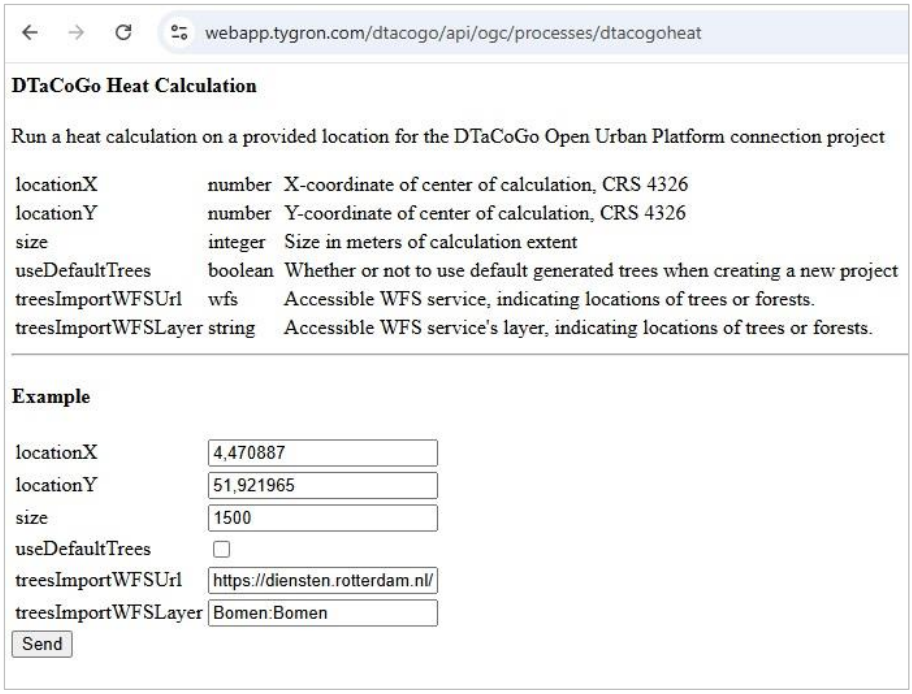



*Figure 8: Parameters of the process dtacogoheat in the Tygron Proxy API, with the Rotterdam tree WFS as input (source: Tygron WebApp Proxy API description)*

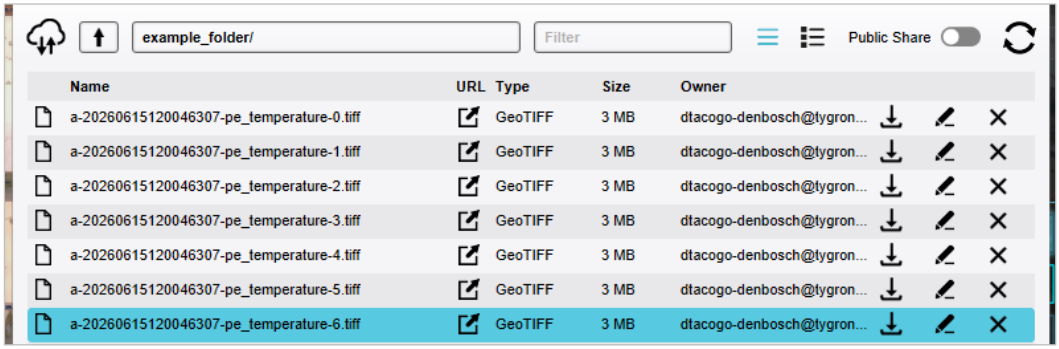



*Figure 9: PET results of a Den Bosch calculation of 15 June 2026 exported to the Tygron GeoShare, one file per time frame (source: Tygron data registration instructions)*

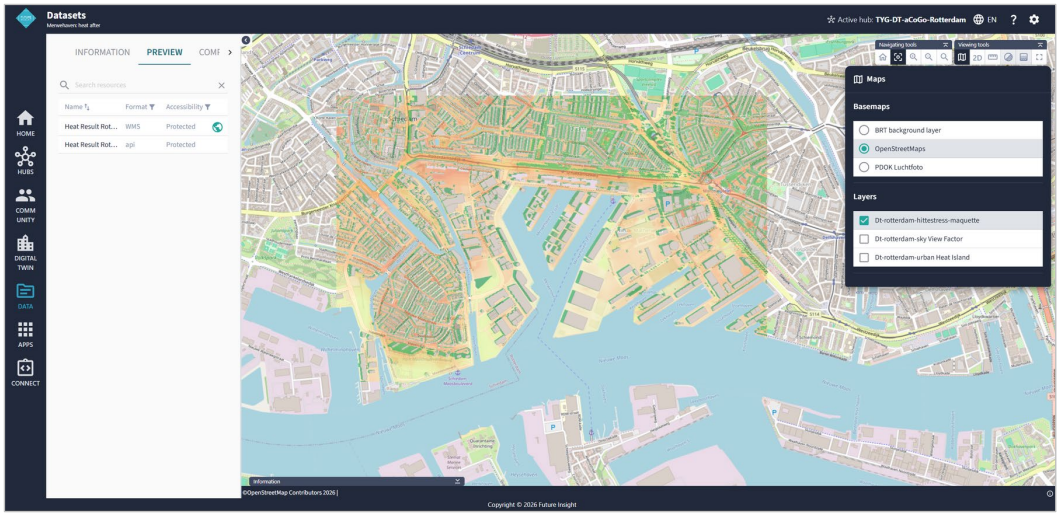



*Figure 10: Heat stress result for Merwehaven ('after' situation), registered in the OUP as a protected WMS and previewed in the OUP viewer (source: D2.1)*

**Output plausibility (V4).** Confirmed on the basis of prior use. Rotterdam has applied the Tygron heat stress calculation, which follows the DPRA methodology, in earlier projects with outcomes that proved valuable for real project use (D3.6). In DT-aCoGo the calculation was fed with input data delivered through the OUP, which passed the checks above. Figure 10 shows the resulting heat stress map for Merwehaven in the 'after' situation, as registered in the OUP.

## 5.4 Summary across models

Table 5 summarises the outcomes per check and model.

*Table 5: Outcome of the validation checks per model*

| Check | Rana | DatMobility | Tygron |
|---|---|---|---|
| **V1** Input interpretation | Resolved; direct harvesting from the OUP by workaround | Resolved; model inspection only by Goudappel (open) | Workaround; protected sources and common-model trees open |
| **V2** Spatial and temporal consistency | Temporal consistency resolved; 3D position and shallow water open | Confirmed | Resolved |
| **V3** Exposure and access | End-to-end confirmed; write-back by workaround (fixed data-service link) | Confirmed | End-to-end through per-hub registrations; write-back by workaround (fixed data-service link) |
| **V4** Output plausibility | Confirmed on the basis of prior project use (D3.6) | Confirmed on the basis of prior project use; calibrated against counts (D3.6) | Confirmed on the basis of prior project use (D3.6) |

Three findings recur across providers. They point to the platform rather than to the individual models:

1. **Write-back of results.** None of the providers can register each analysis result in the OUP automatically, because machine-to-machine identities are bound to the provider's own hub and the OUP does not pass the context of the calling hub or user (Rana, Tygron). As a workaround, results of analyses called from the Unreal Digital Twin are always served through the same data-service link in the OUP, while the underlying dataset is replaced.

2. **Protected input data.** Model platforms cannot yet consume protected OUP data sources directly; data is downloaded, delivered directly or published as a public service (Rana, Tygron).
3. **Spatial reference.** The coordinate system and vertical reference must be agreed across all parties that deliver 3D or map layers (Rana, Tygron).

# 6. Exposure of the validated models in the OUP

The registration of model APIs in the OUP, the federated hub structure and the governance mechanisms are described in D2.1, D3.1 (Chapters 7 and 8) and D3.3 (Chapter 5), and are not repeated here. Table 6 summarises the registrations as they were validated in Chapter 5 (check V3); Figure 2 shows how they connect.

*Table 6: Registrations of the validated models in the OUP*

| Model | Platform hub | Registrations | Authentication | Findable by |
|---|---|---|---|---|
| **Rana** | Nelen & Schuurmans hub | App "Rana Water Intelligence (DT Acogo)"; dataset "Rana API documentation"; flood-model datasets tagged provider Rana / sourceType Flooding | API key per OUP hub (machine-to-machine); Clearly.hub as identity provider for users | Hub "Digital Twin a CoGo – DS4SSCC" (LIMITED) |
| **DatMobility** | GOU-DT-aCoGo-Rotterdam, -DenBosch and -Tampere | Per city: before, during and after scenario datasets, and the GMT documentation | JWT per city (OAuth2 client credentials) | Own city hub only (LIMITED) |
| **Tygron** | Tygron hub | Tygron Heat Calculation Rotterdam, Den Bosch, Tampere and TPG | authenticationToken injected by the OUP, one per registration | Corresponding city hub and The People Group |

The integration of these services in the Unreal Engine visualisation environment is addressed in Deliverable 3.5.

# 7. Conclusions and next steps

## 7.1 Conclusions

- **The model chain is validated for all three models.** For Rana, DatMobility and Tygron, input interpretation, spatial and temporal consistency, and exposure through the OUP were checked, and the issues found were resolved or bypassed with documented procedures (Chapter 5). All three models passed the tests of (semi-)automated data delivery through the CIM and the OUP; the plausibility of their outcomes rests on several years of use in earlier Rotterdam projects.
- **Validation had to take place in the native environments.** Most issues, such as the height offsets of flood results, shallow-water artefacts and gaps in the scenario input, were only visible in the expert environments and not at API level. This confirms the premise of Chapter 3.
- **Integration depth differs per model, and validation follows it.** Rana is fully integrated, DatMobility works with fixed scenarios and Tygron through a proxy. A single central validation procedure is not realistic; a shared set of checks (V1–V4), applied per model, is.
- **Most remaining limitations sit at platform level.** Write-back of results, access to protected data and agreements on spatial reference recur across providers and require changes in the OUP rather than in the models.

## 7.2 Remaining limitations and recommendations

Table 7 lists the open findings from Chapter 5 with a recommendation and the party it is addressed to.

*Table 7: Remaining limitations and recommendations*

| Limitation | Recommendation | Addressed to |
|---|---|---|
| Each analysis result cannot be registered in the OUP automatically (workaround: fixed data-service link with replaced dataset) | Pass hub and user context with calls to limited APIs, and allow OUP-side configuration to register result links from provider responses | OUP / Clearly.hub |
| One credential per API registration | Allow hubs or users to add their own credentials for limited and paid resources, so providers can register one generic endpoint | OUP / Clearly.hub |
| Transient results and broken links go undetected | Support temporary dataset registrations with an expiry date and periodic availability checks | OUP / Clearly.hub |
| Protected OUP data cannot be harvested by model platforms | Provide a secured, tenant-scoped metadata endpoint (e.g. CSW with client credentials) or a data-space connector | OUP / Clearly.hub, Rana |
| Height offsets between 3D layers | Agree one coordinate system and vertical reference for all 3D layers and document it in the recipes | All data and model providers |
| Shallow-water artefacts in 3D | Implement depth-dependent transparency or separate terrain and water tiles | Rana |
| Estimated scenario inputs not visible | Flag estimated or ignored input fields in the scenario results | DatMobility |
| Cities cannot inspect the traffic model | Arrange expert access or a validation report per scenario | Municipalities, Goudappel |
| Common-model tree data not connected; Proxy API not fully OGC conformant | Connect the common-model tree data and complete OGC API Processes conformance | Tygron |

## 7.3 Next steps

The validated services are integrated in the Unreal Engine visualisation environment in Deliverable 3.5. A next step proposed by Rana and Dat.mobility is a combined flood–traffic recipe, in which water depths from a Rana simulation are used to adapt the road network of the Dat.mobility model, executed through an orchestrator component.

# Annex A – Validation evidence documents

The findings in Chapter 5 are based on the following technical reports, delivered by the model providers to the consortium:

- **Rana Water Intelligence (Nelen & Schuurmans), 29 June 2026:** Deliverable I – Gathering data and setting up model (parts 1 and 2) and flowchart “From data to flood models”; II – Authentication user flow; III – Add app to Open Urban Platform; IV – SDK Light: DT-aCoGo flood modelling; V – Retrieving / publishing data via Open Urban Platform; VI – Lessons learned and next steps (part 1: combined flood and traffic modelling recipe; part 2: improvements on 3D water visualisation).
- **Dat.mobility (Goudappel):** Dat.mobility – traffic calculation module (guide); SDK Light: DT-aCoGo traffic model.
- **Tygron, 24 June 2026:** DT-aCoGo Tygron data requirements specification; DT-aCoGo Tygron WebApp Proxy API description; DT-aCoGo Tygron data registration instructions; DT-aCoGo Open Urban Platform recommendations; Tygron Support wiki, Expert mode tutorials 1–4.

# About the European Data Space for Smart Communities

**The European Data Space for Smart Communities (DS4SSCC-DEP) initiative is a pivotal deployment following the preparatory action for a Data Space for Sustainable and Smart Cities and Communities (DS4SSCC).** Emphasising sustainability aspects and diversity in communities, DS4SSCC developed a multi-stakeholder data governance scheme, created a blueprint for the European DS4SSCC, delivered priority datasets, and developed a roadmap towards a mature DS4SSCC.

DS4SSCC-DEP's vision revolves around creating a well-governed federated territorial, place-based data space for smart communities available for developers and infrastructure providers, aligning with the prospects outlined in Europe's Digital Decade objectives. This approach distinguishes itself from sectoral data spaces by encompassing diverse domains, and underpinning governance across all levels of society. The key objectives of the project are:

- **Objective 1: Establishing a Federated and Innovative Data Space**. This phase focuses on creating a large-scale data space controlled by public data holders, ensuring alignment with the Smart Middleware Platform and broader data space ecosystems. It also aims to offer middleware service solutions facilitating data sharing and management while refining the blueprint based on gained experience.
- **Objective 2: Pilots and Refinement of the Data Space Blueprint**. Through 10-12 cross-sectoral data pilots in various EU communities, this objective aims to validate and refine the data space blueprint. Using common high-value datasets, it seeks to create added value by combining data from various domains, contributing to the sustainability plan, and minimising infrastructure investment.
- **Objective 3: Fostering Innovation and Ecosystem Alignment**. Engaging stakeholders, compliance with sector-specific legislation, establishing links with Horizon Europe missions, creating services available via trusted application catalogues, and enhancing ethical considerations and AI-enabled local solutions constitute the focus of this objective.

DS4SSCC-DEP, building upon established European networks, aims to significantly contribute to the sustainability goals of European citizens. It emphasises co-creation, standardisation, business models, strategies for data spaces, and envisions a federated platform.

**DS4SSCC-DEP consortium** includes the following partners: