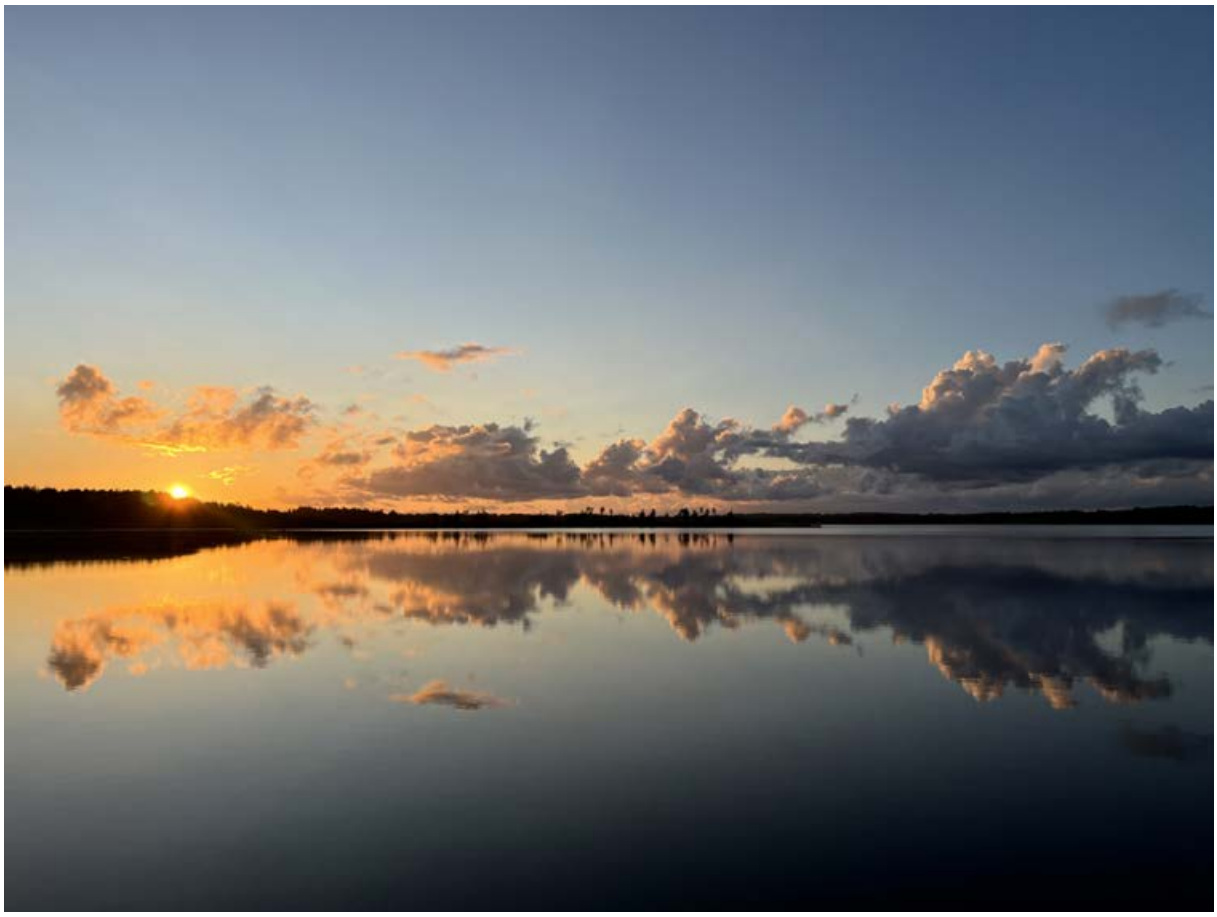
Eiropas Savienības LIFE programmas Integrētais projekts
"Latvijas upju baseinu apsaimniekošanas plānu ieviešana laba virszemes ūdens stāvokļa sasniegšanai"

COMPLETE MODELLING SYSTEM WITH TRANSBOUNDARY INTERFACES

DELIVERABLE R8 (C1D8)



Prepared within the LIFE GoodWater IP Action C1: "Development of the water quality and quantity system for the territory of Latvia"

Riga, 2025

Complete modelling system with transboundary interfaces

Uldis Bethers, Juris Seņņikovs, Pēteris Bethers, Andrejs Timuhins, SIA "Procesu analīzes un izpētes centrs"

Recommended citation: PAIC, 2025. Complete modelling system with transboundary interfaces. LIFE GoodWater IP, Rīga, 65p

Document prepared in the frame of the integrated project "Implementation of River Basin Management Plans of Latvia towards good surface water status" (LIFE GOODWATER IP, LIFE18 IPE/LV/000014), project has received funding from the LIFE Programme of the European Union and the Ministry of Smart Administration and Regional Development Republic of Latvia.

The information reflects only the LIFE GoodWater IP project beneficiaries' view and the European Climate, Infrastructure and Environment Executive Agency is not responsible for any use that may be made of the information contained therein.

Document versioning sheet	
Document version number	**v 1.0**
Document planned elaboration date	**12.2025**
Document elaboration date	**12.2025**
Document actual version elaboration date	**12.2025**
Project activity/sub-activity number	**C1**

SUMMARY

This document summarizes the development and deployment of the transboundary solutions, i.e. the interfaces with water quantity and quality models of neighbouring countries implemented in the water quantity and quality model of Latvia.

Document is written in English, it contains 65 pages, 9 tables and 10 references.

Table of Contents

Introduction

This document is a Technical report on the development and deployment of the interfaces with water quantity and quality models of neighbouring countries. It is Deliverable R8 (C1D8) within LIFE Goodwater project, Action C1.

The developed scripts for transboundary solutions are used with the water quantity and quality modelling system delivered within Deliverable R5 (C1D5) “Complete modelling system – data, models, results” PAIC (2023b). This modelling system comprises of the:

- data prepared within Deliverable R1 (C1D1) – Data base for model implementation, PAIC (2020);
- water quantity and quality modelling tools described in Deliverable R3 (C1D3) – Modelling results for Reference and Baseline scenarios, PAIC (2022b);
- model setups of the calibrated and validated modelling system for the water quantity and quality in the territory of Latvia delivered as Deliverable R2 (C1D2) – Calibrated and validated modelling system in PAIC (2022a);
- modelling results delivered as Deliverable R2 (C1D2) – Calibrated and validated modelling system in PAIC (2022a) and described in the Deliverable R4 (C1D4) – Documentation of development, calibration, validation and results of SWAT modelling system in PAIC (2023a).
- the Python scripts which realize source apportionment and water quality improvement measures in SWAT+ in the Deliverables R6 (C1D6) (Technical report on development and employment of source apportionment, PAIC (2023c)) and R7 (C1D7) (Technical report on development and employment of measure optimisation, PAIC (2023d)).

The transboundary solutions are summarized within the Task 8. Latvia is a transit country in a pathway of nutrients to the Baltic Sea, mainly receiving transboundary flows (from LT, BY, EE, RU); only small part of LV territory drains into RU. LV receives water via rivers flowing from neighbouring countries (Bartuva/Barta, Mūša/Mūsa, Nemunelis/Mēmele from LT, Zapadnaja Dvina/Daugava from BY). Some of LV rivers have transboundary catchments in territories of LT, BY, EE and RU. In such a situation best possible outputs from these countries should be used to ensure that Latvian measures for improvement of water quality account for transboundary pollution. Task 8 is focusing on building interfaces with neighbours, and it is based on the existing modelling system.

We expected that SWAT+ or similar water quality modelling systems will be employed in LT and EE in 2024/2025; however, this was case only for LT and not for EE. Further, contacts with authorities responsible for water quality in BY and RU are impossible since the unprovoked military attack of RU to UA in Feb-2022.

Above circumstances limit the options of development of transboundary interfaces with EE, BY and RU. Nevertheless, with this report we have reached the Milestone M8 “Transboundary interface with neighbouring countries”.

Chapter 1. Executive Summary

1.1 Problem Statement

Latvia is a transit country in the pathway of nutrients to the Baltic Sea, receiving significant transboundary water flows from Lithuania (LT), Belarus (BY), Estonia (EE), and Russia (RU). The SWAT+ water quality modeling system for Latvia must accurately represent inflows from neighbouring countries to ensure that Latvian measures for improving water quality properly account for transboundary pollution.

1.2 Solution Overview

A multi-level transboundary modeling approach has been developed and implemented within the SWAT+ modeling system for Latvia (LIFE IP GOODWATER project, 2020-2027). This approach adapts to varying levels of data availability across different transboundary sources, ranging from full model coupling with high-resolution external models to simple area-based scaling methods.

1.3 Five Transboundary Approaches

The system implements a flexible framework with five distinct approaches:

1. **Full WQ Model Coupling**: Direct integration with neighbouring SWAT+ model systems (Lithuania -> Latvia)
2. **Added Area Method**: Area-based parameter scaling for small transboundary catchments
3. **CORINE Land Cover Based**: Land cover-driven characterization for data-limited regions
4. **Observation Station Data**: Direct use of monitoring station measurements (Daugava from Belarus)
5. **Coarse Model Integration**: Framework for integrating regional-scale models

1.4 Current Implementation Status

Fully Implemented are 3 of 5 approaches.

Approach 1 - Lithuanian Model Coupling: **OPERATIONAL** - 31+ transboundary inlets configured in settings.py - Direct file-based coupling with Lithuanian SWAT+ system outputs - Rivers: Bartuva/Barta, M¯uša/M¯usa, Nemunelis/M¯emele, and tributaries. Implementation: updateTransboundaryInlets() in transboundary.py

Approach 2 - Added Area Method: **OPERATIONAL** - Database field: AddedArea in (catchments, transboundary) table - Automatic parameter scaling in watershed, retention, and HRU calculations - Implementation: Integrated throughout sqliteSetup.py, retention.py, lup.py - Applied to small transboundary catchments where external land use is unknown.

Approach 4 - Observation Data: **OPERATIONAL** - Inlet 1430: Daugava river from Belarus - Monitoring stations: Q=11 (discharge), WQ=70 (water quality) - Implementation: ExtractObservedLoads() function in transboundary.py - Automated time series extraction and SWAT+ format conversion.

Implementatation Framework is ready for 2 of 5 approaches:

Approach 3 - CORINE Land Cover: - CORINE data processing - Intended for Estonian (EE) inflows characterization – Infrastructure exists but requires development.

Approach 5 - Coarse Model Integration: **PARTIALLY READY** - File handling infrastructure exists in updateTransboundaryInlets() - Configuration dictionary extensible for additional model sources - Requires spatial/temporal resolution handling development.

1.5 System Architecture

1.5.1 Technical Components

Database Storage: - PostgreSQL database with versioned data components - Tables: inlets, transboundary catchments, outlets, transfers - Field names configured in settings.py.

Python Scripts: - Core module: transboundary.py (inlet data application) - Data structures: catchments.py (classes for inlets, catchments) - Configuration: settings.py (inlet dictionary, file paths) - Setup generation: sqliteSetup.py (SWAT+ database creation).

SWAT+ Integration: - Recall files: {subbasinNr}t.dat format - Variables: flow, sediment, nutrients (orgn, sedp, no3, solp, etc.) - Daily time step, complete simulation period coverage

1.5.2 Geographic Coverage

System Scale: - 3,780 subcatchments in Latvian system - 179 model watersheds (individual SWAT+ models) - ~195,000 HRUs (hydrological response units) - 32+ configured transboundary inlets.

Transboundary Sources:

Lithuania (LT): 31+ inlets with full SWAT+ model coupling.

Belarus (BY): 1 major inlet (Daugava) with observation data

Estonia (EE): Added area method, while frameworkready for CORINE use.

Russia (RU): Minor contributions via added area method.

1.6 Data Requirements by Approach

1.6.1 Approach 1 (Lithuanian Model)

• **Input**: Lithuanian SWAT+ output files (.out format)

• **Location**: External model directory specified in configuration

• **Update frequency**: Per simulation run

• **Variables**: Complete water quantity and quality time series

1.6.2 Approach 2 (Added Area)

• **Input**: Database table with catchment IDs and external areas (kmˆ2)

• **Format**: (catchments, transboundary) table, AddedArea field

• **Assumption**: Homogeneous land use across border

• **Scaling**: Proportional to area ratio

1.6.3 Approach 4 (Observations)

• **Input**: Monitoring station data files

  – Q_DATA: Discharge observations (mˆ3/s)

  – WQ_DATA: Water quality concentrations (mg/L)

• **Format**: Tab-separated text files with station IDs

• **Conversion**: Automated to kg/day loads for SWAT+ recall format

1.7 Key Findings

1.7.1 Implementation Quality

Strengths:

1. Modular, well-structured code with clear separation of concerns
2. Flexible configuration allowing easy addition of new inlets
3. Automatic integration of external areas in all relevant calculations
4. Multiple data source support (model outputs and observations)

Technical Characteristics:

1. Configuration-driven approach via Python dictionaries
2. Database-centric data management

3. Automated file format conversions

4. Integration with existing watershed hierarchy

1.7.2 Data Processing Pipeline

External Sources -> Configuration Check -> Data Extraction/Copy -> Format Conversion -> SWAT+ Recall Files -> Model Integration

Processing Steps:

1. Read inlet configurations from transBoundaryInlets dictionary

2.Check data source availability (model files vs. observations)

3. Extract or copy time series data

4. Convert to SWAT+ recall format with proper units

5. Write {subbasinNr}t.datfiles to setup directory

6. SWAT+ automatically reads files at runtime

1.8 Conclusions

The multi-level transboundary approach provides a pragmatic solution for representing water quality inflows under varying data availability conditions. The implemented system combines three operational approaches (Lithuanian model coupling, added area method, and observation-based data) to cover 32+ transboundary inlets affecting Latvian water bodies.

Operational Status: The system is fully functional for the primary transboundary sources (Lithuania and Belarus-Daugava), which represent the largest nutrient inputs to Latvia. The framework is extensible and prepared for additional sources as data availability improves.

Documentation Purpose: This document describes the premise, approaches, implementation status, and technical details of the transboundary modeling system to support:

- Model users and operators;

- Stakeholders and water management authorities;

- Partner organizations in neighbouring countries;

- Future system developers and maintainers.

Chapter 2. Introduction

2.1 Background

This document describes the transboundary inflow modeling approaches implemented in the SWAT+ water quality modeling system for Latvia. Latvia receives water via rivers flowing from neighbouring countries: Bartuva/Barta, Mūša/Mūsa, Nemunelis/Mēmele from Lithuania (LT), and Zapadnaja Dvina/Daugava from Belarus (BY). Additionally, some Latvian rivers have transboundary catchments with territories extending into Lithuania, Belarus, Estonia (EE), and Russia (RU).

The modeling system must account for these transboundary inflows to accurately represent water quantity and quality in Latvian water bodies. This requires interfaces with data from neighbouring countries, with approach depending on data availability.

2.2 Study Area

2.2.1 Model Coverage

The Latvian SWAT+ modeling system consists of: - 3,780 catchments - 179 model watersheds - Approximately 195,000 HRUs.

2.2.2 Transboundary Sources

From Lithuania: - Bartuva/Barta river - Mūša/Mūsa river - Nemunelis/Mēmele river - Additional smaller tributaries - Total: 31+ configured transboundary inlets.

From Belarus: - Daugava river (1 inlet) - Inlet ID: 1430.

From Estonia: - Small northern catchments - Framework prepared but not yet configured.

From Russia: - Minor eastern catchment extensions - Handled via added area approach.

2.3 Project Context

This work is part of the LIFE GoodWater IP project (2020-2027) "*Implementation of River Basin Management Plans of Latvia towards good surface water status*". Task 8 focuses on development and deployment of interfaces with water quantity and quality models of neighbouring countries.

The Lithuanian SWAT+ modeling system was developed in parallel (2020-2022 upgrade from SWAT to SWAT+, Lithuanian EPA project), enabling direct model coupling between the two systems.

2.4 Document Purpose and Scope

2.4.1 Purpose

This document describes:

- The premise for transboundary inflow modelling;
- List of approaches depending on available data;
- Implementation status of each approach;
- Technical details of implemented approaches;
- Data requirements and configuration.

2.4.2 Intended Audience

• Technical staff operating the modeling system

• Water management authorities using model results

• Partner organizations in neighboring countries

• Future system developers and maintainers

2.4.3 Scope

The document focuses on transboundary inflow modeling specifically. General SWAT+ model description, input data preparation, and calibration/validation results are documented in other project Deliverables (LIFE GoodWater IP reports R2-R7 (C1D2-C1D7)).

2.4.4 Document Structure

• **Chapter 3**: Multi-level approach framework

• **Chapter 4**: Technical implementation for each approach

• **Chapter 5**: Data requirements and availability

• **Chapter 6**: Current implementation status and validation

• **Appendices**: Code references, configuration templates, data source details

Chapter 3. Multi-Level Approach Framework

3.1 Overview

The transboundary modeling approach uses different methods depending on data availability. Five approaches have been identified, ranging from full model coupling (highest data requirement) to simple area scaling (lowest data requirement). The selection of approach for each transboundary inlet depends on:

1. Availability of external model data;
2. Availability of monitoring station data;
3. Catchment size;
4. Significance of contribution to Latvian water bodies.

3.2 Approach Selection Criteria

3.2.1 Data Availability Assessment

High Data Availability: - Full SWAT+ model available for upstream territory - Daily time series of all water quality parameters - Compatible model structure and output format - **-> Use Approach 1 (Full Model Coupling)**

Medium-High Data Availability: - Monitoring stations at border or near inlet point - Regular measurements of discharge and water quality - Sufficient temporal coverage - **-> Use Approach 4 (Observation Data)**

Medium Data Availability: - CORINE land cover data available - Catchment boundaries defined - No detailed model or monitoring data - **-> Use Approach 3 (CORINE-based)**

Low Data Availability: - Only catchment area known - Land use outside boundary unknown- Small catchment contribution - **-> Use Approach 2 (Added Area)**

Very Low Data Availability + Large Catchment: - Coarse-resolution regional model available - Limited spatial/temporal resolution - Significant contribution to loads - **-> Use Approach 5 (Coarse Model)**

3.2.2 Catchment Size Considerations

Large Catchments (>500 kmˆ2): - Significant contribution to loads - Justify effort for detailed characterization - Prefer Approach 1 or 4.

Medium Catchments (50-500 kmˆ2): - Moderate contribution - Use Approach 3 if available, otherwise Approach 2.

Small Catchments (<50 kmˆ2): - Minor contribution - Approach 2 usually sufficient

3.3 Approach Descriptions

3.3.1 Approach 1: Full WQ Model Coupling

Description: Direct coupling with external SWAT+ model system.

Data Source: Output files from Lithuanian SWAT+ model.

Method:

1. Lithuanian model runs independently
2. Output files (.out format) generated for border catchments
3. Files copied to Latvian model directory
4. Latvian model reads as external inflow (recall files)

Advantages:

- High spatial and temporal resolution
- Consistent modeling framework
- Full water quality parameter set
- Scenario compatibility (both models can run same scenarios)

Limitations:

- Requires coordination with external modeling team
- Dependent on external model updates
- File management overhead

Application: Lithuanian transboundary inflows (31+ inlets).

3.3.2 Approach 2: Added Area Method

Description: Scale catchment parameters proportionally to total area including external part.

Data Source: Catchment area outside modeling domain (from GIS).

Method:

1. Define catchment with part outside Latvian boundary
2. Calculate external area in database (AddedArea field, kmˆ2)

3. System automatically scales all calculations:

Total_Area = Internal_Area + External_Area Scaling_Factor = Total_Area / Internal_Area Scaled_Load = Internal_Load * Scaling_Factor

Assumptions:

- Land use in external part similar to internal part
- Hydrological processes similar
- No point sources in external part

Advantages:

- Simple implementation
- Low data requirement
- Automatic integration in all calculations

Limitations:

- Assumes homogeneous land use
- Less accurate for heterogeneous catchments
- Cannot represent point sources in external area

Application: Small transboundary catchments where external land use is not known.

3.3.3 Approach 3: CORINE Land Cover Based

Description: Characterize external catchment part using CORINE land cover data.

Data Source: CORINE land cover raster (pan-European dataset).

Method:

1. Extract CORINE classes for external catchment area
2. Calculate land use distribution
3. Translate CORINE classes to SWAT+ land use types
4. Generate parameters based on land use composition
5. Scale loads accordingly

Advantages:

- Better than simple area scaling
- Uses actual land cover information
- CORINE data freely available for EU countries

Limitations:

- CORINE resolution limited (100m)
- No information on management practices
- Still requires assumptions about parameters

Application: Intended for Estonian (EE) inflows where detailed model is not available.

3.3.4 Approach 4: Observation Station Data

Description: Use measured discharge and water quality data from monitoring stations.

Data Source: Observation databases (Q_DATA, WQ_DATA files).

Method:

1. Identify monitoring station at or near border
2. Extract discharge time series (mˆ3/s)
3. Extract water quality concentrations (mg/L)
4. Convert concentrations to loads (kg/day):

   Load = Concentration * Discharge * Conversion_Factor

5. Generate SWAT+ recall format file

Advantages:

- Direct measurements (no modeling uncertainty at inlet)
- Independent verification possible
- Suitable for large rivers with monitoring

Limitations:

- Requires monitoring station near inlet point
- Data gaps require interpolation
- Limited to monitored parameters
- No scenario analysis capability (historical data only)

Application: Daugava inflow from Belarus (Inlet 1430, Stations Q=11, WQ=70).

3.3.5 Approach 5: Coarse Model Integration

Description: Integration with regional or national-scale models with coarser resolution.

Data Source: Output from large-scale models (e.g., national water balance models).

Method:

1. Identify compatible regional model
2. Extract time series for relevant location
3. Handle spatial/temporal resolution differences:
    - Spatial disaggregation if needed
    - Temporal interpolation if needed
4. Convert to SWAT+ format

Advantages:

- Better than no information for large catchments
- Can provide scenario analysis
- May be the only option for non-EU countries

Limitations:

- Resolution mismatch challenges
- Uncertainty propagation
- Parameter compatibility issues

Application: Potential for large external catchments where detailed data unavailable.

3.4 Selection Decision Tree

Approach comparison matrix is given in Table 1. The Selection decision tree is as follows:

- Is detailed SWAT+ model available for upstream area?
    - |- YES -> Use Approach 1 (Full Model Coupling)
    - - NO -> Continue
- Is monitoring station available at border?
    - - YES -> Use Approach 4 (Observation Data)
    - - NO -> Continue
- Is CORINE data sufficient for characterization?
    - - YES -> Use Approach 3 (CORINE-based)
    - - NO -> Continue
- Is catchment large and regional model available?
    - - YES -> Use Approach 5 (Coarse Model) [if implemented]
    - - NO -> Use Approach 2 (Added Area)

Table 1: Approach Comparison Matrix

Aspect	Approach 1	Approach 2	Approach 3	Approach 4	Approach 5
Data Requirement	Very High	Very Low	Medium	Medium-High	Medium
Implementation Effort	Medium	Low	Medium	Medium	High
Accuracy	High	Low	Medium	High	Medium
Scenario Capability	Yes	Yes	Yes	No	Yes
Spatial Resolution	High	Lumped	Medium	Point	Low
Update Frequency	Per run	One-time	One-time	Per run	Per run

3.5 Current Application Summary

Current application summary is given in Table 2.

Table 2. Application Summary

Source Country	Rivers/Inlets	Approach Used	Inlets Count
Lithuania	Multiple	1	31+
Belarus	Daugava	4	1
Estonia	Small N catchments	3	N/A
Russia	Minor E catchments	2	N/A
Mixed	Small transboundary	2	few

Chapter 4. Technical Implementation

4.1 System Architecture

4.1.1 Overview

The transboundary system is implemented through:

- Python scripts in code/lib/py/directory
- Configuration in code/Scripts/settings.py
- PostgreSQL database tables
- SWAT+ recall files ({subbasinNr}t.dat format)

4.1.2 Key Software Components

Main Script: transboundary.py

- Function: updateTransboundaryInlets(self) – applies transboundary data to setup
- Function: ExtractObservedLoads() - extracts observation data to SWAT+ format

Data Structures: catchments.py

- Class: TExternalInlet - external inlet definition
- Class: TExternalCatchment - external catchment area
- Class: TCatchment – includes external area field

Configuration: settings.py

- Dictionary: transBoundaryInlets - inlet configurations
- Variables: Database table names, field names, file paths

Setup Generation: sqliteSetup.py

- Includes external areas in watershed calculations
- Scales parameters proportionally

Calculations: retention.py, lup.py, regionalization.py

- External area automatically included in all relevant calculations

4.1.3 Data Flow

External Data Sources

|

+ ---- + ---- +

| |

LT SWAT+ (.out) Observations

| |

transboundary.py

| - updateTransboundaryInlets()

| - ExtractObservedLoads()

|

{subbasin}t.dat

|

SWAT+ Model

4.2 Implementation Details per Approach

4.2.1 Approach 1: Lithuanian Model Coupling

Configuration

Location: settings.py, lines 562-608

Dictionary Structure:

```
transBoundaryInlets = {
        inletID: {
                'input_rec': 'filename.out', # LT model output filename
                'External_WATOUT': 'LT_filename.txt' # File in common directory
        },
        ...
}
```

Example Entry: 445: {'input_rec': '1000065.out', 'External_WATOUT': 'LT_1000065.txt'},

File Directory: transBoundaryFileDir = commonDataDir + 'TransboundaryInlets\\'

Implementation Function

Function: updateTransboundaryInlets(self)

File: transboundary.py, lines 13-41

Method of: TmdbSetup class

Logic:

```
def updateTransboundaryInlets(self):
        directory = self.SetupPath() + self.lastsubdir
        for cment in self.watershed.catchments:
            if len(cment.segment.externalinlets) > 0:
                for inl in cment.segment.externalinlets:
                    if inl.id in settings.transBoundaryInlets:
                        externalmodelfile=settings.transBoundaryInlets[inl.id]
                        ["External_WATOUT"]
                        outfile = directory + "/" + f"
                        {cment.localsubbasinnr}t.dat"
                        if os.path.exists(externalmodelfile):
                            # Copy from Lithuanian model
                            shutil.copyfile(externalmodelfile, outfile)
                        else:
                            # Extract from observations (fallback)
                            qstation=settings.transBoundaryInlets[inl.id]
                            ["QStation"]
                            wqstation=settings.transBoundaryInlets[inl.id]
                            ["WQStation"]
                            # Call ExtractObservedLoads()
```

File Format

Input: Lithuanian SWAT+ output file (.out format)

Output: SWAT+ recall file ({subbasinNr}t.dat)

Format: Space-separated values, columns: jday mo day_mo yr name type flo sed orgn sedp no3 solp chla nh3 no2 cbod dox san sil cla

Variables

• flo - flow (mˆ3/s)

• sed - sediment (metric tons)

• orgn - organic nitrogen (kg)

• sedp - particulate phosphorus (kg)

• no3 - nitrate (kg)

• solp - soluble phosphorus (kg)

• chla - chlorophyll-a (kg)

• nh3 - ammonia (kg)

• no2 - nitrite (kg)

• cbod - CBOD (kg)

• dox - dissolved oxygen (kg)

• Sediment fractions: san, sil, cla, sag, lag, grv

Configured Inlets (Sample)

445: {'input_rec':'1000065.out', 'External_WATOUT':'LT_1000065.txt'}, *# Bartuva*
1030: {'input_rec':'1000064.out', 'External_WATOUT':'LT_1000064.txt'},
1041: {'input_rec':'18131.out', 'External_WATOUT':'LT_18131.txt'},
1049: {'input_rec':'1000062.out', 'External_WATOUT':'LT_1000062.txt'},
2202: {"input_rec":"1000014.out", "External_WATOUT":"Musa.txt"}, *# Mūša*
 # ... (31+ total)

4.2.2 Approach 2: Added Area Method

Data Structure

Class: TExternalCatchment (catchments.py, lines 444-448)

```
class TExternalCatchment:
        def __init__(self):
                self.id = -1 # catchment ID
                self.catchment = None # TCatchment object reference
                self.addedarea = 0 # additional area in m^2
```

Class: TCatchment field (catchments.py, line 249)
self.externalarea = 0 *# area in m^2 outside modelling domain*

Database Configuration

Table: (catchments, transboundary)

Fields:

- catchmentID - integer, catchment identifier

- AddedArea - real, area in km^2 outside modeling domain

Settings Variable (settings.py, line 321):

"FIELDNAME_ADDEDAREA" : "AddedArea"

Table Location (settings.py):

TransboundaryCatchmentFile = ("catchments", "transboundary")

Data Loading

Function: readExternalCatchments()

File: catchments.py, lines 960-980

Method of: TProject class

SQL Query:

```
SELECT catchmentID, AddedArea
FROM catchments.transboundary
WHERE AddedArea IS NOT NULL;
```

Processing:

```
for r in rs:
        cID = r[0]
        if cID in self.catchments:
                ec = TExternalCatchment()
                self.externalcatchments[cID] = ec
                ec.id = cID
                ec.addedarea = r[1] # km^2 from database
                ec.catchment = self.catchments[cID]
# Sum all external areas per catchment
for cment in self.catchments.values():
        cment.externalarea = sum([v.addedarea for v in self.externalcatchments.values()
                if v.catchment == cment])
```

Area Scaling in Calculations

1. Watershed Area Calculations

File: sqliteSetup.py, lines 2362-2366

```
watershed_area = np.sum([cment.area + cment.externalarea
        for cment in self.watershed.catchments])
for cment in self.watershed.catchments:
        cmentArea = cment.area + cment.externalarea
        cmnetAreaExtFr = 1 + cment.externalarea / cment.area # Scaling multiplier
```

2. Retention Calculations

File: retention.py, lines 112-113

```
extmult = cment.externalarea / (cment.externalarea + cment.area)
cmentarea = (cment.area + cment.externalarea) / 1e4 # Convert to hectares
```

3. HRU Area Distribution

File: retention.py, line 421

area = hru.area / cment.totHRUArea * (cment.area + cment.externalarea)

4. Land Use Parameters

File: lup.py, lines 274-277

```
watershed_area = np.sum([cment.area + cment.externalarea
        for cment in self.watershed.catchments])
cmnetAreaExtFr = 1 + cment.externalarea / cment.area
```

5. Regionalization

File: regionalization.py, lines 401-404

```
a = cment.area + cment.externalarea
        for inl in cment.segment.externalinlets:
a += inl.externalarea
```

Calculation Example

Given:

- Internal catchment area: 100 kmˆ2
- External area: 20 kmˆ2
- Internal annual N load: 100 kg/ha/year

Calculation:

Total area = 100 + 20 = 120 km^2
Scaling factor = 120 / 100 = 1.2
Scaled N load = 100 * 1.2 = 120 kg/ha/year
Total catchment load = 120 kg/ha/year * 120 km^2 * 100 ha/km^2 = 1,440,000 kg/year

4.2.3 Approach 4: Observation Station Data

Configuration

Location: settings.py, line 567

1430: {"QStation": 11, "WQStation": 70, "External_WATOUT": "Daugava_6.txt"}

Parameters:

- inletID: 1430 (Daugava from Belarus)
- QStation: 11 (discharge monitoring station ID)
- WQStation: 70 (water quality monitoring station ID)

- External_WATOUT: Filename (used if file exists, otherwise extract from stations)

Implementation Function

Function: ExtractObservedLoads()

File: transboundary.py, lines 43-96

Parameters:

def ExtractObservedLoads(QobsFileName, WqObsFileName, QstatNr, WQStatNr, StartDate, EndDate, outFileName):

Process Steps:

1. Read Station Data (lines 44-63):

```
def readStationData(fileName, stationID, parse_dates="DATE"):
        st_data = pnd.read_csv(file_str + r".txt",
                na_values=["-"],
                sep='\t',
                skiprows=[1],
                header=0,
                parse_dates=[parse_dates])
        st_data = st_data[st_data[st_data.columns[0]] == stationID]
        st_data.index = st_data[parse_dates]
        st_data = st_data.sort_index()
        return st_data.interpolate()
```

2. Extract and Interpolate (lines 64-70):

```
Q = readStationData(QobsFileName, QstatNr, "DATE")
WQ = readStationData(WqObsFileName, WQStatNr, "Date")
dtrange = pnd.date_range(start=StartDate, end=EndDate)
Q = np.array(Q.reindex(dtrange.values).interpolate()).flatten()
WQ = np.array(WQ.reindex(dtrange.values).interpolate()).transpose()
```

3. Convert Concentrations to Loads (lines 73-75):

```
varcoeff = np.array([86400.0, # Flow: mˆ3/s -> mˆ3/day
                86.4*1e-3, # SS: mg/L -> kg/day (with mˆ3/s)
                86.4, # Nutrients: mg/L -> kg/day
                ... ],
dtype=np.float)
WQ = WQ * varcoeff[:WQ.shape[0], np.newaxis] * Q[np.newaxis, :]
WQ[0,:] = Q * varcoeff[0]
```

4. Format Output (lines 77-96):

```
restbl = pnd.DataFrame(index=dtrange)
restbl["jday"] = dtrange.dayofyear
restbl["mo"] = dtrange.month
restbl["day_mo"] = dtrange.day
restbl["yr"] = dtrange.year
restbl["name"] = "ch1"
restbl["type"] = "ds"
# Map observed variables to SWAT+ variables
restbl["flo"] = WQ[qwd["Flow"]]
restbl["sed"] = WQ[qwd["SS"]]
restbl["dox"] = WQ[qwd["DO"]]
restbl["cbod"] = WQ[qwd["BOD7"]]
restbl["nh3"] = WQ[qwd["NH4-N"]]
restbl["no2"] = WQ[qwd["NO2-N"]]
restbl["no3"] = WQ[qwd["NO3-N"]]
restbl["solp"] = WQ[qwd["PO4-P"]]
restbl["orgn"] = WQ[qwd["N-Org"]]
restbl["sedp"] = WQ[qwd["P-Org"]]
restbl.to_csv(outFileName, sep=" ", index=False, header=True, float_format='%.7g')
```

Input Data Files

Discharge File (Q_DATA):

STAT	DATE	Flow
11	2010.01.01	120.5
11	2010.01.02 1	18.3

...

Water Quality File (WQ_DATA):

Station	Date	SS	DO	BOD7	NH4-N	NO2-N	NO3-N	N-Org	PO4-P	P-Org
70	2010.01.01	15.2	11.5	3.2	0.15	0.02	1.8	0.3	0.08	0.12
70	2010.01.02	14.8	11.3	3.1	0.14	0.02	1.7	0.28	0.07	0.11

...

Output File Format

File: {subbasinNr}t.dat

```
jday mo day_mo yr name type flo sed orgn sedp no3 solp chla nh3 no2 cbod dox san sil cla
1 1 1 2010 ch1 ds 10368000 1312.03 25.92 10.368 155.52 6.9120 0 12.96 1.7280 27.648
99.360
```

2 1 2 2010 ch1 ds 10195200 1275.69 24.12 9.4608 146.66 6.0364 0 12.073 1.6588 26.752 97.526

...

Table 3. Unit Conversions, Q is in mˆ3/s

Variable	**Input Unit**	**Output Unit**	**Conversion Factor**
Flow	mˆ3/s	mˆ3/day	86400
SS	mg/L	metric tons/day	86.4 * 10-3 * Q
Nutrients	mg/L	kg/day	86.4 * Q
DO, BOD	mg/L	kg/day	86.4 * Q

4.2.4 Approaches 3 and 5

Approach 3 (CORINE): Would require:

- CORINE raster reading functions
- Land cover class translation tables
- Parameter generation based on land cover distribution

Approach 5 (Coarse Models): Infrastructure partially ready:

- updateTransboundaryInlets() can handle external files
- Configuration dictionary extensible - Needs spatial/temporal resolution handling

4.3 Configuration and Setup

4.3.1 Configuration File Structure

File: code/Scripts/settings.py

Transboundary Block (lines 562-608):

```
# Directory for transboundary inlet files
transBoundaryFileDir = commonDataDir + 'TransboundaryInlets\\'
# Inlet configuration dictionary
transBoundaryInlets = {
        inletID: {configuration_dict},
        ...
}
```

Configuration Dictionary Keys:

- input_rec: Lithuanian model output filename (Approach 1)

- External_WATOUT: File in common transboundary directory (Approach 1)

- QStation: Discharge station ID (Approach 4)

- WQStation: Water quality station ID (Approach 4)

4.3.2 Database Table Configuration

Field Name Variables (settings.py, line 321):

```
WatershedDefinitionVars = {
        "FIELDNAME_ADDEDAREA": "AddedArea",
        # ... other field names
}
```

Table Location Variables:

```
TransboundaryInletFile = ("catchments", "inlets")
TransboundaryCatchmentFile = ("catchments", "transboundary")
```

4.3.3 File Naming Conventions

Lithuanian Model Outputs:

- Pattern: LT_{catchmentID}.txt

- Example:LT_1000065.txt

- Location: transBoundaryFileDir

SWAT+ Recall Files:

- Pattern: {localsubbasinnr}t.dat

- Example: 5t.dat (for subbasin 5)

- Location: SWAT+ setup directory

Observation Data Files:

- Discharge: Configured in Q_DATA variable

- Water Quality: Configured in WQ_DATA variable

- Format: Tab-separated text files

4.3.4 Directory Organization

```
Project Root
|       - Common Data
|               `- TransboundaryInlets/
|                       |- LT_1000065.txt
|                       |- LT_1000064.txt
|                       `- ...
|       - Projects
|               `- Setup/
|                       `- Watersheds/
|                               `- BasinName/
|                                       `- WatershedName/
|                                               `- Case/
|                                                       |- 1t.dat
|                                                       |- 5t.dat
|                                                       `- ...
`- Scripts/
        |- settings.py
        `- lib/py/
                |- transboundary.py
                |- catchments.py
                `- ...
```

4.4 Code Structure and Key Functions

4.4.1 Function Summary

Table 4. Function summary

Function	File	Lines	Purpose
updateTransboundaryInlets()	transboundary.py	13-41	Apply transboundary data to model setup
ExtractObservedLoads()	transboundary.py	43-96	Extract observation data to SWAT+ format
readExternalInlets()	catchments.py	938-958	Load inlet definitions from database
readExternalCatchments()	catchments.py	960-980	Load external catchment areas
PrepareInlets()	catchments.py	106-150	Prepare inter-watershed inlet files

4.4.2 Class Structure

TExternalInlet (catchments.py:437-442):

```
class TExternalInlet:
        self.id = -1
        self.name = ""
        self.segmentTo = None # TRiverSegment object
        self.externalarea = 0.0 # mˆ2
```

TExternalCatchment (catchments.py:444-448):

```
class TExternalCatchment:
        self.id = -1
        self.catchment = None # TCatchment object
        self.addedarea = 0 # mˆ2
```

TCatchment (partial, catchments.py:249):

```
class TCatchment:
```

 # ... other fields
self.externalarea = 0 *# mˆ2 outside domain*

4.4.3 Integration Points

External area is used in:

1. **sqliteSetup.py**: Lines 588, 2362-2366, 2763, 2828
    - Watershed area calculations
    - HRU area scaling
    - Parameter setup
2. **retention.py**: Lines 112-113, 286, 351, 415, 421, 433
    - Load calculations
    - Retention coefficients
    - HRU-level area distribution
3. **lup.py**: Lines 274, 277
    - Land use parameter scaling
    - Watershed fractions
4. **regionalization.py**: Lines 401, 404
    - Parameter regionalization
    - Area-weighted averaging
5. **initPAICSWAT.py**: Line 144
    - Output file generation
    - Reporting external areas

Chapter 5. Data Requirements and Availability

5.1 Data Requirements Matrix

Data requirements matrix by approach is given in Table 5, whilst by the geographic area in Table 6.

Table 5. Data Requirements Matrix By Approach.

Approach	**Data Source**	**Format**	**Resolution**	**Update Frequency**
1 – Model Coupling	Lithuanian SWAT+ outputs	.out text files	Daily, catchment-level	Per simulation run
2 – Added Area	GIS catchment boundaries	Database table	One-time	Setup phase only
3 – CORINE	CORINE land cover	Raster (100m)	One-time	When updated (~6 years)
4 – Observations	Monitoring stations	Tabseparated text	Daily (or as available)	Per simulation run
5 – Coarse Model	Regional model outputs	Variable	Variable	Per simulation run

Table 6. Data Requirements Matrix By Geographic Area

Source Region	**Primary Approach**	**Data Available**
Lithuania	Approach 1	Full SWAT+ model
Belarus (Daugava)	Approach 4	Monitoring stations
Estonia	Approach 3	CORINE data
Russia	Approach 2	Catchment boundaries
Small catchments	Approach 2	Catchment boundaries

5.2 Current Data Sources

5.2.1 Lithuanian SWAT+ Model Outputs

Model System: Lithuanian SWAT+ developed 2020-2022 (Lithuanian EPA project)

Coverage: Entire Lithuanian territory, 1,237 catchments, 106 model watersheds

Output Files:

- Format: SWAT+ standard output (.out files)
- Location: Lithuanian model project directory
- Variables: Complete water quantity and quality suite
- Temporal resolution: Daily
- Spatial resolution: Catchment outlet level

Transfer Method:

- Files copied to common directory: transBoundaryFileDir
- Naming convention: LT_{catchmentID}.txt
- Update: Per simulation run, coordinated between teams

Configuration: 31+ border catchments configured in transBoundaryInlets dictionary

Example Output Variables:

flo - Flow (m^3/s)
sed - Sediment (metric tons/day)
orgn - Organic nitrogen (kg/day)
sedp - Particulate phosphorus (kg/day)
no3 - Nitrate nitrogen (kg/day)
solp - Soluble phosphorus (kg/day)
nh3 - Ammonia nitrogen (kg/day)
no2 - Nitrite nitrogen (kg/day)
cbod - Carbonaceous BOD (kg/day)
dox - Dissolved oxygen (kg/day)

5.2.2 Latvian Observation Network

Discharge Observations (Q_DATA):

- Source: Latvian Environment Agency monitoring network
- Format: Tab-separated text file
- Columns: STAT (station ID), DATE, Flow (mˆ3/s)

- Temporal coverage: Historical period (typically 2000-present)

- Update frequency: Regular monitoring program

Water Quality Observations (WQ_DATA):

- Source: Latvian Environment Agency WQ monitoring

- Format: Tab-separated text file - Columns: Station, Date, SS, DO, BOD7, NH4-N, NO2-N, NO3-N, N-total, PO4-P, P-total, N-Org, P-Org

- Sampling frequency: Typically monthly

- Coverage: Selected monitoring stations

Daugava Stations:

- Discharge station: ID = 11

- Water quality station: ID = 70 - Location: Near Belarusian border

- Purpose: Transboundary inlet 1430

5.2.3 Belarusian Monitoring Data

Availability: Limited direct access to Belarusian monitoring data

Current Approach: Use Latvian monitoring stations near border: - Daugava discharge and WQ at first Latvian station - Represents conditions at border - Data from Latvian monitoring network

Data Exchange: No formal bilateral data exchange agreement currently in place.

5.2.4 External Catchment Areas

Source: GIS analysis of catchment boundaries

Method:

1. Delineate catchments from DEM

2. Overlay with country boundaries

3. Calculate area outside Latvian territory

4. Store in database table

Database Table: (catchments, transboundary)

- Field: catchmentID (integer)

- Field: AddedArea (real, kmˆ2)

Quality: Depends on DEM and boundary accuracy

5.2.5 CORINE Land Cover Database

Source: Copernicus Land Monitoring Service

Coverage: All EU countries (including Estonia, Latvia, Lithuania)

Resolution: 100m raster

Classes: 44 land cover classes (reduced from detailed nomenclature)

Versions:

- CORINE 2018 (most recent)
- Updates approximately every 6 years

Availability: Freely available from Copernicus

5.3 Data Formats and Standards

5.3.1 SWAT+ Output Format

File Extension: .out or .txt

Structure: Space-separated values

Header Line: Column names

Daily Records: One line per day

Example:

```
jday mo day yr name gis_id flo sed orgn sedp no3 solp chla nh3 no2 cbod dox
1    1   1  2010 ch1  1000   120 15  2.5  1.2  18  0.8  0.05 1.5 0.2 3.2  11.5
2    1   2  2010 ch1  1000   118 14  2.4  1.1  17  0.7  0.04 1.4 0.2 3.1  11.3
```

5.3.2 SWAT+ Recall Format

File Name: {subbasinNr}t.dat

Purpose: External inflow to SWAT+ model

Format: Space-separated, specific column order

Required Columns:

jday mo day_mo yr name type flo sed orgn sedp no3 solp chla nh3 no2 cbod dox san sil cla

Name Field: Arbitrary name (e.g., “ch1”)

Type Field: “ds” (daily series)

Temporal Coverage: Must cover entire simulation period

5.3.3 Observation Data Format

Discharge File (Q_DATA):

STAT<tab>DATE<tab>Flow
11<tab>2010.01.01<tab>120.5
11<tab>2010.01.02<tab>118.3

Water Quality File (WQ_DATA):

Station<tab>Date<tab>SS<tab>DO<tab>BOD7<tab>NH4-N<tab>NO2-N<tab>NO3-N<tab>...
70<tab>2010.01.01<tab>15.2<tab>11.5<tab>3.2<tab>0.15<tab>0.02<tab>1.8<tab>...

Units: - Flow: mˆ3/s - Concentrations: mg/L - Date format: YYYY.MM.DD

5.3.4 Database Tables

Inlets Table (catchments.inlets):

```
CREATE TABLE catchments.inlets (
        inletID INTEGER PRIMARY KEY,
        inletName TEXT,
        inletTo INTEGER, -- River segment ID
        externalArea REAL -- Optional, mˆ2
);
```

Transboundary Catchments Table (catchments.transboundary):

```
CREATE TABLE catchments.transboundary (
        catchmentID INTEGER PRIMARY KEY,
        AddedArea REAL -- kmˆ2
);
```

5.4 Data Quality and Uncertainty

5.4.1 Quality Control Procedures

Lithuanian Model Outputs:

- Validated through Lithuanian model calibration
- Mass balance checks in Lithuanian system
- Comparison with observations at Lithuanian stations

Observation Data:

- Latvian Environment, Geology and Meteorology Centre QA/QC procedures

- Outlier detection and removal

- Gap filling via interpolation (linear)

- Missing value handling: - or NA in input files

External Areas:

- GIS validation against topographic maps

- Cross-check with official statistics where available

- Manual verification of large areas

5.4.2 Gap Filling Methods

Time Series Gaps:

- Method: Linear interpolation (pandas.interpolate())

- Applied to: Both discharge and water quality

- Limitation: Simple method, may not capture events

Missing Parameters:

- If parameter not monitored: Set to zero or background value

- Document assumptions in setup notes

5.4.3 Uncertainty Estimation

Approach 1 (Model Coupling):

- Uncertainty from Lithuanian model calibration. Typically NSE > 0.5 for discharge, NSE > 0.3 for nutrients

- Propagates Lithuanian model uncertainty to Latvian system

Approach 2 (Added Area):

- Assumes homogeneous land use -> can be significant error source

- Uncertainty increases with heterogeneity

- Better for small, uniform catchments

Approach 4 (Observations):

- Measurement uncertainty: Typically 5-10% for discharge, 10-20% for concentrations

- Sampling frequency: Monthly sampling may miss events

- Interpolation uncertainty: Higher for irregular sampling

5.5 Transboundary Inlet Catalog

5.5.1 Lithuanian and Belarussian Inlets (31+ configured)

The list of Lithuanian (31+) and Belarussian (Daugava) Inlets is provided in Tables 7-8.

Table 7. List of Lithuanian Inlets.

Inlet ID	River/Location	Input File	External File
445	Bārtuva	1000065.out	LT_1000065.txt
1030		1000064.out	LT_1000064.txt
1041		18131.out	LT_18131.txt
1049		1000062.out	LT_1000062.txt
2181		1000032.out	LT_1000032.txt
2194		7193.out	LT_7193.txt
2202	Mūša	1000014.out	Musa.txt
2259		1000031.out	LT_1000031.txt
2261		1000026.out	LT_1000026.txt
2263		1000028.out	LT_1000028.txt
2264		1000029.out	LT_1000029.txt
2300		1000024.out	LT_1000024.txt
2310		1000018.out	LT_1000018.txt
2312		1000003.out	LT_1000003.txt
2316		1000002.out	LT_1000002.txt
2317		1000006.out	LT_1000006.txt
2321		1000005.out	LT_1000005.txt
2324		1000009.out	LT_1000009.txt
2326		1000010.out	LT_1000010.txt

Inlet ID	River/Location	Input File	External File
2327		1000012.out	LT_1000012.txt
2350		1000015.out	LT_1000015.txt
2468		1000025.out	LT_1000025.txt
2471		1000022.out	LT_1000022.txt
2472		1000021.out	LT_1000021.txt
2475		1000020.out	LT_1000020.txt
2483		1000007.out	LT_1000007.txt
2500		1000016.out	LT_1000016.txt
2505		21113.out	LT_21113.txt
2640		19612.out	LT_19612.txt
2642		20381.out	LT_20381.txt
2957		1000035.out	LT_1000035.txt

(Note: Full table continues with additional inlets 2974, 2975, 3226)

Table 8. List of Belorussian Inlets.

Inlet ID	River	Q Station	WQ Station	External File
1430	Daugava	11	70	Daugava_6.txt

5.5.3 Other Transboundary Catchments

External catchments with added area approach (Approach 2) are defined in database table (catchments, transboundary). Specific IDs and areas depend on GIS analysis results.

5.6 Data Update Procedures

5.6.1 Regular Simulation Runs

Frequency: As needed for scenario analysis

Lithuanian Model Outputs:

1. Coordinate with Lithuanian team on scenario definition
2. Lithuanian team runs model, generates output files
3. Files transferred to transBoundaryFileDir
4. Latvian model setup updated
5. Latvian model run executed

Observation Data:

1. Extract latest data from monitoring database
2. Update Q_DATA and WQ_DATA files
3. Run ExtractObservedLoads() via model setup script
4. Verify recall files generated correctly

5.6.2 Database Updates

External Areas:

- Update: When catchment delineation changes
- Process: Re-run GIS analysis, update database table
- Frequency: Rare (only with major setup revisions)

Inlet Definitions:

- Update: When new inlets added or configuration changed
- Process: Edit settings.py configuration dictionary
- Frequency: As needed during development

5.6.3 Version Control

Code: SVN version control system

Data:

- Database: PostgreSQL with backup procedures
- External files: Versioned in shared directory
- Configuration: Tracked in SVN with code

Documentation: Changes documented in project notes and reports

Chapter 6. Current Implementation Status

6.1 Implementation Status Summary

6.1.1 Fully Implemented Features

Approach 1: Lithuanian Model Coupling [OPERATIONAL]

- Status: Operational
- Inlets configured: 31+
- Rivers covered - all: Bārtuva/Bārta, Mūša/Mūsa, Nemunelis/Mēmele, tributaries
- Function: updateTransboundaryInlets() in transboundary.py
- Tested: Yes, in operational use

Approach 2: Added Area Method [OPERATIONAL]

- Status: Operational
- Database table: (catchments, transboundary) with AddedArea field
- Automatic scaling: Integrated in all calculation modules
- Files: sqliteSetup.py, retention.py, lup.py, regionalization.py
- Tested: Yes, catchments with external areas processed correctly

Approach 4: Observation Station Data [OPERATIONAL]

- Status: Operational
- Inlet: 1430 (Daugava from Belarus)
- Stations: Q=11, WQ=70
- Function: ExtractObservedLoads() in transboundary.py
- Tested: Yes, generates correct recall files

6.1.2 Partially Implemented Features

Approach 5: Coarse Model Integration [PARTIALLY READY, USED in LT]

- Status: Infrastructure ready, implemented in LT model for BY and RU
- Available: File handling in updateTransboundaryInlets(), extensible configuration dictionary
- Missing: Spatial/temporal resolution handling, Specific coarse model interfaces

6.1.3 Not Implemented Features

Approach 3: CORINE Land Cover Based [NOT IMPLEMENTED]

- Status: Not implemented
- Reason: Priority given to approaches with immediate data availability
- Requirements: CORINE data processing, land cover translation tables
- Potential use: Estonian inflows, other data-limited regions

6.2 Validation Results

6.2.1 Approach 1 Validation (Lithuanian Model Coupling)

Method: Comparison of Lithuanian model outputs at border with Latvian model inflows

Rivers Validated: Bārtuva, Mūša

Check Performed:

- Mass balance: Loads entering Latvian system match Lithuanian outputs
- Time series continuity: No gaps or format errors
- Unit consistency: Verified mˆ3/s flow, kg/day loads

Result: Files correctly transferred and read by Latvian SWAT+ system

Calibration Status:

- Lithuanian model: Calibrated and validated (2020-2022 project)
- Latvian model: Calibrated independently, transboundary inputs treated as boundary conditions
- No specific cross-border calibration performed (different calibration periods, data availability)

6.2.2 Approach 2 Validation (Added Area)

Method: Check that external areas correctly included in calculations

Validation Steps:

1. Identify catchments with AddedArea > 0 in database
2. Verify externalarea field populated in catchment objects
3. Check area calculations in output:
    - Watershed total area includes external areas

- HRU areas scaled proportionally

4. Mass balance: Loads scale with total area

Result: External areas correctly integrated

Example Check:

 # Catchment with added area
Internal area: 100 km^2
External area: 20 km^2
Expected scaling: 1.2
 # Verification in code
watershed_area = sum(cment.area + cment.externalarea) *# Includes external*
cmnetAreaExtFr = 1 + cment.externalarea / cment.area *# = 1.2 [OK]*

6.2.3 Approach 4 Validation (Daugava Observations)

Method: Comparison of extracted loads with manual calculations

Validation Steps:

1. Extract sample period from observation files
2. Calculate loads manually: Load = Concentration * Flow * 86.4
3. Compare with ExtractObservedLoads() output
4. Verify SWAT+ recall file format correct

Result: Unit conversions correct, file format accepted by SWAT+

Data Quality Checks:

- Missing data: Linear interpolation applied
- Outliers: Checked insource data (Latvian EPA QC)
- Temporal coverage: Complete for simulation period

Comparison with Upstream Observations:

- Limited data available from Belarusian side
- Latvian station assumed representative of border conditions

6.3 Known Limitations

6.3.1 Technical Constraints

Approach 1 (Lithuanian Coupling):

- Dependency: Requires Lithuanian model run before Latvian run
- Coordination: File transfer must be managed manually or via script
- Scenario compatibility: Both models must use compatible scenarios
- Format dependency: Sensitive to changes in SWAT+ output format

Approach 2 (Added Area):

- Assumption: Homogeneous land use may not hold for all catchments
- No spatial detail: Cannot represent point sources in external area
- Parameter uncertainty: External area characteristics estimated, not measured

Approach 4 (Observations):

- Temporal resolution: Monthly WQ sampling may miss events
- Parameter coverage: Only monitored parameters available
- Scenario limitation: Historical data only, cannot simulate future scenarios
- Station location: May not be exactly at border

6.3.2 Data Availability Issues

Belarusian Data:

- Limited access to upstream monitoring data
- No formal data exchange agreement
- Reliance on Latvian border stations

Estonian Data:

- Small catchment contributions
- CORINE approach not yet implemented
- Currently may use added area as fallback

Russian Data:

- Very limited transboundary area
- Minor contribution to total loads

- Added area approach sufficient for current needs

6.3.3 Model Assumptions

Homogeneous Land Use (Approach 2):

- Valid for: Small, uniform catchments
- Questionable for: Large or heterogeneous catchments
- Impact: May over/underestimate loads if land use differs significantly

Monitoring Station Location (Approach 4):

- Assumption: Station represents border conditions
- Reality: May be some distance from actual border
- Impact: Includes/excludes small catchment area

Temporal Interpolation:

- Method: Linear interpolation for missing data
- Limitation: Does not capture event dynamics
- Impact: May smooth peak events

6.4 Computational Performance

6.4.1 Processing Times

Transboundary Data Application:

- Lithuanian files (31 inlets): ~1-2 seconds (file copy operations)
- Observation extraction (1 inlet): ~5-10 seconds (data processing)
- External area loading: <1 second (database query)

Total Overhead: Negligible compared to SWAT+ model run time

6.4.2 Resource Requirements

Disk Space:

- Lithuanian output files: ~1-5 MB per inlet per year
- Observation data files: ~1-10 MB total
- Recall files generated: ~1-5 MB per inlet per year

Memory: No significant memory requirements for transboundary processing

Network: File transfer from Lithuanian system if not on shared storage

6.4.3 Scalability

Current Configuration: 32+ inlets handled efficiently

Potential Expansion:

- Adding inlets: Simple, modify configuration dictionary
- Performance: Linear scaling, no performance issues expected up to 100+ inlets

6.5 Operational Experience

6.5.1 Setup and Configuration

Ease of Use:

- Configuration via Python dictionary in settings.py
- Clear structure, easy to add new inlets
- Database tables straightforward

Common Issues:

- File path errors: Ensure transBoundaryFileDir correct
- Missing files: Lithuanian outputs must be in place before Latvian run
- Station ID errors: Verify station IDs match data files

Solutions:

- Use absolute paths in configuration
- Implement file existence checks (already in code)
- Document station IDs in settings comments

6.5.2 Maintenance

Regular Tasks:

- Update Lithuanian output files when new scenarios run
- Update observation data files with latest monitoring data
- Verify recall files generated correctly before SWAT+ run

Occasional Tasks:

- Add new inlets when needed (modify settings.py)

- Update external areas if catchment delineation changes
- Review and update documentation

Effort: Low maintenance once configured

6.5.3 User Feedback

Model Operators: Configuration clear, system works reliably

Issues Reported: Minimal, mainly related to file path configuration

Improvements Implemented: File existence checking, clear error messages

6.6 Integration with Modeling Workflow

6.6.1 Typical Workflow

1. **Scenario Definition**: Define modeling scenario (baseline, measures, climate, etc.)
2. **Lithuanian Model**:
    - Run Lithuanian SWAT+ for defined scenario
    - Generate output files for border catchments
    - Transfer files to transBoundaryFileDir
3. **Observation Data**:
    - Extract latest monitoring data
    - Update Q_DATA and WQ_DATA files (if needed)
4. **Latvian Model Setup**:
    - Run model setup scripts
    - Scripts automatically:
        - Load external areas from database
        - Copy Lithuanian files or extract observations
        - Generate recall files
        - Create SWAT+ input database
5. **Model Execution**:
    - Run SWAT+ for all watersheds
    - SWAT+ automatically reads recall files
6. **Post-Processing**:

• Extract results

• Visualize in PAICSWAT and QGIS

• Generate reports

6.6.2 Scenario Analysis

Baseline Scenario:

- Lithuanian model: Current conditions
- Observations: Historical period data
- External areas: Current delineation

Measure Scenarios:

- Lithuanian model: Run with measures in Lithuanian territory
- Latvian model: Run with measures in Latvian territory
- Combination: Both models with measures

Scenario Compatibility:

- Approach 1: Full compatibility, both models can simulate same scenarios
- Approach 2: Scales with internal parameters, inherits scenario assumptions
- Approach 4: Limited to historical data, cannot simulate future scenarios

6.6.3 Quality Assurance

Checks Performed:

1. File format validation: Recall files match SWAT+ requirements
2. Temporal coverage: Data covers simulation period
3. Mass balance: Loads consistent across boundaries
4. Unit verification: Conversions correct (mˆ3/s, kg/day, etc.)
5. Missing data: Interpolation log reviewed

Error Handling:

- File not found: Warning message, halt setup
- Format errors: Detailed error message with line number
- Data gaps: Log warning, interpolation applied

6.7 Documentation Status

6.7.1 Code Documentation

Inline Comments:

- transboundary.py: Moderate (main logic commented)
- catchments.py: Moderate (class definitions documented)
- settings.py: Good (configuration options explained)

Function Docstrings:

- Present for main functions
- Could be expanded with parameter descriptions

6.7.2 User Documentation

Configuration Guide:

- Documented in GOODWATER reports (R2-R7)
- This document provides comprehensive reference

Operational Procedures:

- Workflow documented in project notes
- Could benefit from step-by-step user manual

6.7.3 Technical Documentation

This Document:

- Comprehensive technical reference
- Code structure and line references provided
- Data formats documented

Related Documents:

- GOODWATER R2 (C1D2): System structure
- GOODWATER R3-R6 (C1D2-C1D6): Calibration, validation, applications
- Lithuanian reports: Upstream model documentation, e.g. PAIC (2022c)

6.8 Comparison with Original Plan

6.8.1 Planned vs. Implemented

Implementation status of transboundary solutions is shown in Table 9.

Table 9. Implementation status of transboundary solutions.

Component	Planned	Implemented	Status
LT model coupling	Yes	Yes	Complete
BY observation data	Yes	Yes	Complete
Added area method	Yes	Yes	Complete
CORINE approach	Yes	No	Framework ready
Coarse model framework	Yes	Partial	Infrastructure ready
EE inflows	Yes	Yes (added area)	Corine approach to add
RU inflows	Yes	Yes (added area)	Complete

6.8.2 Deviations from Plan

CORINE Approach Not Implemented:

- Reason: Estonian inflows small, added area used instead and is sufficient
- Priority: Resources focused on primary sources (LT, BY)
- Impact: Minimal, can be added if needed

Coarse Model Framework Partial:

- Reason: No specific coarse model identified for integration
- Available: Infrastructure can accommodate future additions
- Impact: None, currently, ready for future needs

6.8.3 Additional Features Implemented

Automatic Area Scaling:

- Integrated throughout calculation modules
- More comprehensive than originally planned
- Ensures consistency across all load calculations

Flexible Configuration:

- Dictionary-based approach allows easy extension
- Can handle mixed approaches per inlet
- Fallback mechanism (observations if model file not found)

6.9 System Maturity Assessment

6.9.1 Development Stage

Status: Production-ready for implemented approaches

Maturity Level:

- Approach 1: Production, tested, operational
- Approach 2: Production, tested, operational
- Approach 4: Production, tested, operational
- Approach 3: Design stage (planned)
- Approach 5: Framework stage (infrastructure ready)

6.9.2 Reliability

Code Stability: High for implemented approaches

Error Rate: Low, issues mainly configuration-related

Recovery: Good error messages, clear failure points

6.9.3 Maintainability

Code Quality: Good structure, moderate documentation

Configuration: Clear and extensible

Dependencies: Standard libraries (pandas, numpy), SWAT+ executable

Future Development: Framework supports additions with minimal changes to existing code

Chapter 7. Appendices

7.1 Appendix A: Code Reference Guide

7.1.1 A.1 Key File Locations

Python Scripts (relative to project root):

- code/lib/py/transboundary.py - Core transboundary functionality
- code/lib/py/catchments.py - Data structures and database loading
- code/Scripts/settings.py - Configuration
- code/lib/py/sqliteSetup.py - SWAT+ setup generation
- code/lib/py/retention.py - Load and retention calculations
- code/lib/py/lup.py - Land use parameters
- code/lib/py/regionalization.py – Parameter regionalization

Data Directories:

- {commonDataDir}/TransboundaryInlets/ - Lithuanian model output files
- Projects/Setup/Watersheds/{Basin}/{Watershed}/{Case}/ - SWAT+ setups with recall files

7.1.2 A.2 Function Documentation

updateTransboundaryInlets(self)

File: transboundary.py, lines 13-41

Class Method of: TmdbSetup

Purpose: Apply transboundary inflow data to SWAT+ model setup

Parameters: self (TmdbSetup object)

Returns: None (modifies setup files in place)

Called by: Model setup scripts

Logic:

1. Iterate through catchments with external inlets
2. Check configuration in settings.transBoundaryInlets
3. If Lithuanian file exists: copy to setup directory
4. If not: extract from observations (call ExtractObservedLoads)

ExtractObservedLoads()

File: transboundary.py, lines 43-96

Purpose: Extract monitoring data and convert to SWAT+ recall format

Parameters:

- QobsFileName: Path to discharge observation file
- WqObsFileName: Path to water quality observation file
- QstatNr: Discharge station ID
- WQStatNr: Water quality station ID
- StartDate: Start date for extraction
- EndDate: End date for extraction
- outFileName: Output file path

Returns: None (writes file)

Process:

1. Read station data from files
2. Interpolate to daily time series
3. Convert concentrations to loads
4. Format as SWAT+ recall file

readExternalCatchments()

File: catchments.py, lines 960-980

Class Method of: TProject

Purpose: Load external catchment areas from database

Parameters: None (uses self.catchments)

Returns: None (populates self.externalcatchments)

SQL: SELECT catchmentID, AddedArea FROM catchments.transboundary WHERE AddedArea IS NOT NULL

7.1.3 A.3 Class Definitions

TExternalInlet

File: catchments.py, lines 437-442

Purpose: Store external inlet definition

Fields:

- id (int): Inlet identifier

- name (str): Inlet name

- segmentTo (TRiverSegment): Target river segment

- externalarea (float): External area in mˆ2

TExternalCatchment

File: catchments.py, lines 444-448

Purpose: Store external catchment area information

Fields:

- id (int): Catchment identifier

- catchment (TCatchment): Reference to catchment object

- addedarea (float): Additional area in mˆ2

TCatchment (relevant field)

File: catchments.py, line 249

Field: externalarea (float): Area in mˆ2 outside modelling domain

Usage: Sum of all added areas for this catchment

7.2 Appendix B: Database Schema

7.2.1 B.1 Inlets Table

Table Name: catchments.inlets

```
CREATE TABLE catchments.inlets (
        inletID INTEGER PRIMARY KEY,
        inletName TEXT,
        inletTo INTEGER, -- River segment ID
        externalArea REAL -- Optional, in mˆ2
);
```

Example:

```
INSERT INTO catchments.inlets VALUES (1430, 'Daugava_from_BY', 1250, NULL);
```

7.2.2 B.2 Transboundary Catchments Table

Table Name: catchments.transboundary

```
CREATE TABLE catchments.transboundary (
        catchmentID INTEGER PRIMARY KEY,
        AddedArea REAL -- In kmˆ2
);
```

Example:

```
INSERT INTO catchments.transboundary VALUES (1523, 12.5);
```

7.3 Appendix C: Configuration Templates

7.3.1 C.1 Lithuanian Model Coupling Configuration

```
# In settings.py
transBoundaryFileDir = r'I:\Common\Data\TransboundaryInlets\\'
transBoundaryInlets = {
        # Inlet ID: Configuration dictionary
        445: {
                'input_rec': '1000065.out', # LT model output filename
                'External_WATOUT': 'LT_1000065.txt' # File in transBoundaryFileDir
        },
        1030: {
                'input_rec': '1000064.out',
                'External_WATOUT': 'LT_1000064.txt'
        },
        # Add more inlets as needed
}
```

7.3.2 C.2 Observation-Based Configuration

```
# In settings.py
Q_DATA = r'I:\Common\Data\Observations\discharge.txt'
WQ_DATA = r'I:\Common\Data\Observations\water_quality.txt'
transBoundaryInlets = {
        1430: {
                "QStation": 11, # Discharge station ID
                "WQStation": 70, # WQ station ID
                "External_WATOUT": "Daugava_6.txt" # Fallback filename
        },
}
```

7.3.3 C.3 Mixed Configuration Example

```
# In settings.py
transBoundaryInlets = {
        # Lithuanian inlets - model coupling
        445: {'input_rec': '1000065.out', 'External_WATOUT': 'LT_1000065.txt'},
        1030: {'input_rec': '1000064.out', 'External_WATOUT': 'LT_1000064.txt'},
        # Daugava - observations
        1430: {"QStation": 11, "WQStation": 70, "External_WATOUT": "Daugava_6.txt"},
}
```

7.3.4 C.4 Database Configuration

```
# In settings.py
WatershedDefinitionVars = {
        "FIELDNAME_ADDEDAREA": "AddedArea", # km^2
        # ... other field names
}
TransboundaryInletFile = ("catchments", "inlets")
TransboundaryCatchmentFile = ("catchments", "transboundary")
```

7.4 Appendix D: File Format Specifications

7.4.1 D.1 SWAT+ Recall File Format

Filename: {subbasinNr}t.dat

Format: Space-separated values

Header: Column names on first line

Columns (in order):

jday mo day_mo yr name type flo sed orgn sedp no3 solp chla nh3 no2 cbod dox san sil cla

Example:

jday mo day_mo yr name type flo sed orgn sedp no3 solp chla nh3 no2 cbod dox san sil cla
1 1 1 2010 ch1 ds 120.5 15.2 2.5 1.2 18.0 0.8 0.05 1.5 0.2 3.2 11.5 0 0 0 0 0 0 0
2 1 2 2010 ch1 ds 118.3 14.8 2.4 1.1 17.5 0.7 0.04 1.4 0.2 3.1 11.3 0 0 0 0 0 0 0

Field Descriptions:

- jday: Julian day (1-366)
- mo: Month (1-12)
- day_mo: Day of month (1-31)
- yr: Year (4 digits)
- name: Channel name (arbitrary, e.g., “ch1”)
- type: Data type (“ds” = daily series)
- flo: Flow (mˆ3/s)
- sed: Sediment (metric tons/day)
- orgn: Organic nitrogen (kg/day)
- sedp: Particulate phosphorus (kg/day)
- no3: Nitrate nitrogen (kg/day)
- solp: Soluble phosphorus (kg/day)
- chla: Chlorophyll-a (kg/day)
- nh3: Ammonia nitrogen (kg/day)
- no2: Nitrite nitrogen (kg/day)
- cbod: Carbonaceous BOD (kg/day)
- dox: Dissolved oxygen (kg/day)

- san, sil, cla, sag, lag, grv: Sediment fractions (metric tons/day)

- null: Temperature placeholder (not used)

7.4.2 D.2 Observation Data File Formats

Discharge File (Q_DATA):

STAT<tab>DATE<tab>Flow
11<tab>2010.01.01<tab>120.5
11<tab>2010.01.02<tab>118.3

Water Quality File (WQ_DATA):

Station <tab> Date <tab> SS <tab> DO <tab> BOD7 <tab> NH4-N <tab> NO2-N <tab> NO3-N <tab> N mineral <tab> 70 <tab> 2010.01.01 <tab> 15.2 <tab> 11.5 <tab> 3.2 <tab> 0.15 <tab> 0.02 <tab> 1.8 <tab> 1.97 <tab> 2.5 <tab>

Units:

- Flow: mˆ3/s

- All concentrations: mg/L

- Date format: YYYY.MM.DD or YYYYMM-DD

7.5 Appendix E: Glossary

Added Area: Catchment area outside modeling domain, stored in database, used for proportional parameter scaling

Catchment: Drainage area contributing to a river segment outlet (syn: subbasin)

External Inlet: Inflow point from outside modeling domain (transboundary source)

External Area: See Added Area

HRU: Hydrological Response Unit - homogeneous area within catchment with unique land use, soil, and slope combination

Recall File: SWAT+ input file format for time series data (discharge, loads) from external sources

Transboundary Catchment: Catchment with portion of drainage area outside country boundary

Transboundary Inlet: Point where water enters modeling domain from upstream country

Watershed: Collection of connected catchments processed as single SWAT+ model

7.6 Appendix F: Acronyms

• **BY**: Belarus

• **CORINE**: Coordination of Information on the Environment (EU land cover database)

• **DEM**: Digital Elevation Model

• **EE**: Estonia

• **EPA**: Environmental Protection Agency

• **EU**: European Union

• **GIS**: Geographic Information System

• **GOODWATER**: LIFE IP project “Implementation of River Basin Management Plans of Latvia”

• **HRU**: Hydrological Response Unit

• **LT**: Lithuania

• **LV**: Latvia

• **NSE**: Nash-Sutcliffe Efficiency

• **PAIC**: “Procesu analīzes un izpētes centrs” (Process Analysis and Research Centre)

• **PBIAS**: Percent Bias

• **QA/QC**: Quality Assurance / Quality Control

• **RBMP**: River Basin Management Plan

• **RU**: Russia

• **SWAT**: Soil and Water Assessment Tool

• **SWAT+**: Soil and Water Assessment Tool Plus (enhanced version)

• **WFD**: Water Framework Directive

• **WQ**: Water Quality

References

PAIC, 2020. DATA BASE FOR MODEL IMPLEMENTATION. LIFE GoodWater IP deliverable R1, Rīga, 31p.

PAIC, 2022a. CALIBRATED AND VALIDATED MODELLING SYSTEM. LIFE GoodWater IP deliverable R2, Rīga, 47p.

PAIC, 2022b. MODELLING RESULTS FOR REFERENCE AND BASELINE SCENARIOS. LIFE GoodWater IP deliverable R3, Rīga, 30p.

PAIC, 2022c. Water modelling system of Lithuania upgrade and software support services. 2nd Final report. Riga, September-2022.

PAIC, 2023a. Documentation of development, calibration, validation and results of SWAT modelling system. LIFE GoodWater IP deliverable R4, Rīga.

PAIC, 2023b. Complete modelling system – data, models, results. LIFE GoodWater IP deliverable R5, Rīga.

PAIC, 2023c. Technical report on development and employment of source apportionment. LIFE GoodWater IP deliverable R6, Rīga, 14p

PAIC, 2023d. Technical report on development and employment of measure optimisation. LIFE GoodWater IP deliverable R7, Rīga, 48p

SWAT+ Theoretical Documentation: Available at https://swat.tamu.edu/software/plus/

SWAT+ Input/Output Documentation: Available at https://swat.tamu.edu/software/plus/