AI for engineering

MDPI

Technical Note

Agentic SWMM: Auditable and Reproducible Stormwater Modelling Workflow with Agent Skills and Model Context Protocol

Zhonghao Zhang and Caterina Valeo *

Department of Mechanical Engineering, University of Victoria,
P.O. Box 1700, STN CSC, Victoria, BC V8W 2Y2, Canada; zhonghaoz@uvic.ca
* Correspondence: valeo@uvic.ca

Abstract

Configuring urban hydrological models, such as the Storm Water Management Model (SWMM), for operational use remains onerous for many modellers. We propose aiswmm, a SWMM-specialized agentic runtime, together with an Agentic SWMM workflow that embeds (Agent) Skills and Model Context Protocol (MCP) tools to automate QGIS preprocessing, SWMM configuration, execution, and postprocessing. We demonstrate this natural-language triggered workflow on the Tod Creek watershed (located on the Saanich Peninsula, British Columbia). We also validate the proposed Agentic SWMM workflow at three levels: (i) a QGIS-based watershed-pour-point detection that agrees with the commercial PCSWMM® method to within 0.88% of the watershed perimeter (approximately 7.5 pixels in the digital elevation model); (ii) byte-identical SWMM output files (Secure Hash Algorithm 256-bit identical) between the command-line execution and the MCP paths across 60 paired simulations, and (iii) peak inflow at the watershed outlet matching to three significant digits between the manual SWMM interface and Agentic SWMM workflows. The results confirm that Agentic SWMM workflow can produce the same outputs with the manual SWMM interface, as they are designed to use the same computational engine. We also propose a verification-first contract and byte-level audit chain that record the inputs, parameters, and outputs of each run, thereby supporting the auditability and reproducibility. The aiswmm runtime, Skills, MCP servers, and byte-level audit chain are released as open source and remain compatible with mainstream agentic runtimes (Codex, Claude Code, Hermes, and OpenClaw) to support reproducible SWMM modelling driven by natural language.

Keywords: hydrological modelling; Agentic AI; SWMM; Model Context Protocol (MCP); Agent Skills; agentic runtime; reproducible modelling workflow

Check for updates

Academic Editors: Moutaman M. Abbas, Radu Muntean and Mengqian Lu

Received: 18 March 2026
Revised: 3 June 2026
Accepted: 3 June 2026
Published: 9 June 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

1.1. Hydrological Modelling Bottleneck—Physically Based Modelling

Hydrological modelling of urban stormwater systems remains operationally onerous [1–3]. Currently, a major pain point in hydrological modelling is that physics-based modelling software, such as Environmental Protection Agency (EPA) Storm Water Management Model (SWMM), PCSWMM® and MIKE+®, require extensive interaction between the user and the interface to constantly adjust various parameters during the modelling process [4,5]. While the visualization capabilities are excellent and the built-in physically based formulas make the process and results interpretable, the need for

multiple scenarios and simulations can result in a tedious modelling configuration process, especially during model calibration [3,6,7]. As well, users often need a significant amount of time to understand and familiarize themselves with the interface and to debug software errors [8,9].

Many useful Python packages have been developed and have made important contributions to SWMM-based modelling research. For example, 'SWMManywhere' supports the generation of synthetic urban drainage network models and SWMM input files from publicly available geospatial data. This makes it possible to conduct preliminary urban drainage network forecasts in scarce data areas [10]. 'PySWMM' provides Python-based access to the SWMM engine, enabling runtime interaction, parameter modification, optimization, control, and postprocessing [11]. Recent studies have also coupled PySWMM with optimization libraries such as 'pymoo' [12] for automatic SWMM calibration. As well, 'swmm_api' (Python toolkit) can support SWMM setup, editing, analysis, visualization, sensitivity analysis, and calibration [2]. These open-source Python packages have significantly reduced the complexity of user interface operations, but they are primarily geared towards programmers. For normal users, end-to-end configuration, operation, calibration, result extraction, and quality assurance still face efficiency challenges.

1.2. Machine Learning Alternatives

Conversely, with the development of machine learning, hydrology is currently focusing on various alternative models such as long short-term memory and physics-informed neural networks, increasing their time series recognition capabilities or adding physical constraints based on partial differential equation residuals [13–15]. This allows for the training of data-driven models that can match physics-based models in most contexts, thereby increasing operational efficiency. However, the defining weakness of these models is their poor generalization ability [16]. Their near-black-box training characteristics make accurate representation of interpretability and physical consistency very difficult, often leading to overfitting issues in data scarce situations [17–19]. Furthermore, these surrogate models may be hard to interpret in terms of the physical processes occurring in real engineering projects, flowrate estimation and locating low-impact developments [20–22]. The auditability and traceability are primary requirements in operational hydrology. Physically based modelling cannot be, and should not be, fully replaced by other surrogate models, especially for physics informed decisions [23,24]. Machine learning model training essentially learns data mapping patterns rather than truly understanding the physics [21,25]. These limitations force future researchers to seek out methods that may compromise integrity of the physics being modelled. Therefore, since one single solution is not feasible, a complementary strategy should be sought that retains the physics-based concepts, is an easy-to-operate, interpretable, and auditable hydrological model. The modelling workflow can potentially couple the Python tools described in Section 1.1, as well as eliminating learning barriers.

1.3. Agentic AI, Agentic Runtime, and Model Context Protocol

Currently, various Agentic AI technologies are flourishing. Unlike conventional chat-oriented systems (e.g., ChatGPT or Gemini) that primarily optimize for natural-language responses, the Agentic AI differs from a chat assistant in that it can plan multi-step actions, call external tools (e.g., web retrieval, scripts, application programming interfaces, file operations), observe their outputs, and iteratively refine planning toward a user-specified goal under explicit permissions [26]. More specifically, the agentic runtime is an emerging core execution layer within Agentic AI systems, and this core determines where and

how the capabilities of Agentic AI are executed, constrained, and recorded. For example, OpenClaw is a typical agentic runtime designed for complex goals [27]. It was released by Peter Steinberger on 29 January 2026, in a GitHub repository [28,29]. As well, Hermes (v0.8.0), a type of memory-augmented agentic runtime, was released by Nous Research on 8 April 2026, in another GitHub repository [30]. The emergence of these agentic runtimes has significantly improved the generality of large language models (LLMs). With the development of Agentic AI, some open-source software such as SWMM, can expose communication methods to Agentic AI through Model Context Protocol (MCP), enabling software invocation [31,32]. For example, Claude Code's Agent Skills (hereafter, Skills) can package workflows together that an agent can recognize; thus, achieving simple migration of automated workflows [32]. The idea of automated workflows is not new [33]. Platforms such as nodemation (n8n) provide a practical example of workflow automation, which aims to create simple chatbots or trigger specific workflows at specific times [34–36]; however, the configuration process is often highly complex.

To address the current pain points in hydrological modelling, we propose an Agentic SWMM workflow based on aiswmm runtime, Skills, MCP, and the SWMM engine. This enables the configuration of SWMM and the definition of user-defined plots using natural language. In addition, we verified the consistency of the entire workflow using the Tod Creek watershed in Saanich, BC, Canada. The results show that SWMM configured through natural language was consistent with the results configured manually through the SWMM graphical user interface (GUI). This verified that the core hydrological derivation process does not depend on the Agentic AI, but on the SWMM engine that it invokes. This paper proposes a practical architecture that integrates the following elements:

(i) aiswmm as a workflow execution runtime;
(ii) Skills as reusable workflow modules;
(iii) Model Context Protocol as a standard interface layer;
(iv) a verification-first traceability contract for each model run;
(v) equivalence verification between manual SWMM execution and Agentic SWMM execution.

Together, this architecture ensures that the Agentic SWMM workflow can operate the SWMM engine and the QGIS (Version 4.0.2) engine as well as the downstream Python scripts in a controlled and auditable manner via natural language.

2. Materials and Methods

The methodology is depicted in Figure 1 to indicate how the Agentic SWMM orchestrated tasks, starting with input Geo-layers data for the case study watershed. As shown in Figure 1, we present a complete workflow that packages the multi-source data (digital elevation model (DEM), land use, soil, rainfall) required for modelling into an input package for Geographic Information System (GIS) MCP. End users can interact with SWMM by issuing commands via natural language as shown in the Supplementary File, Section S1 and Figure S1. In the middle layer, we packaged SWMM's commands as callable tools through MCP. Since SWMM lacks the lookup table capability to transform raw data layers into model parameters, we generated a built-in transformation lookup table through dialogue in the GIS MCP as well.

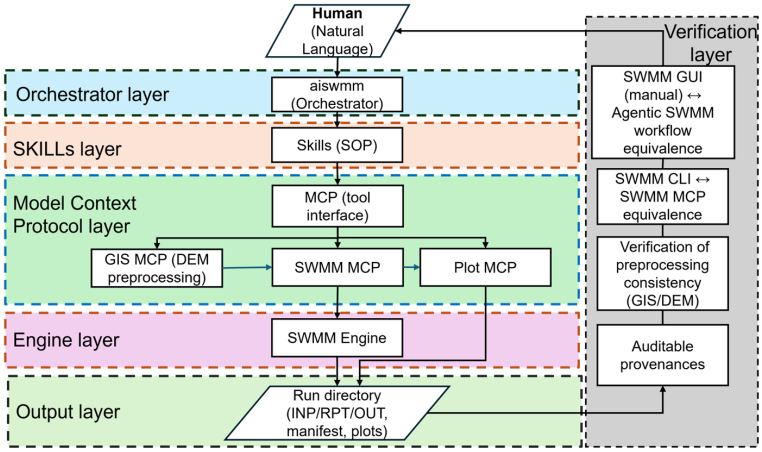


Figure 1. Methodology flowchart of the proposed Agentic SWMM workflow.

2.1. Study Site

The Tod Creek watershed, which spans about 17 km^2, is situated on Vancouver Island's Saanich Peninsula in British Columbia, Canada. The covered region is mostly rural with a few small residential villages scattered throughout the watershed's undulating terrain (see Figure 2a in the digital elevation model). The 1000–1200 mm of annual precipitation is mostly concentrated between October and March of the following year. The soil types range from poorly draining clays to well-drained soils that are rich in organic matter (see Figure 2b). The resolution of the digital elevation model is 27 m.

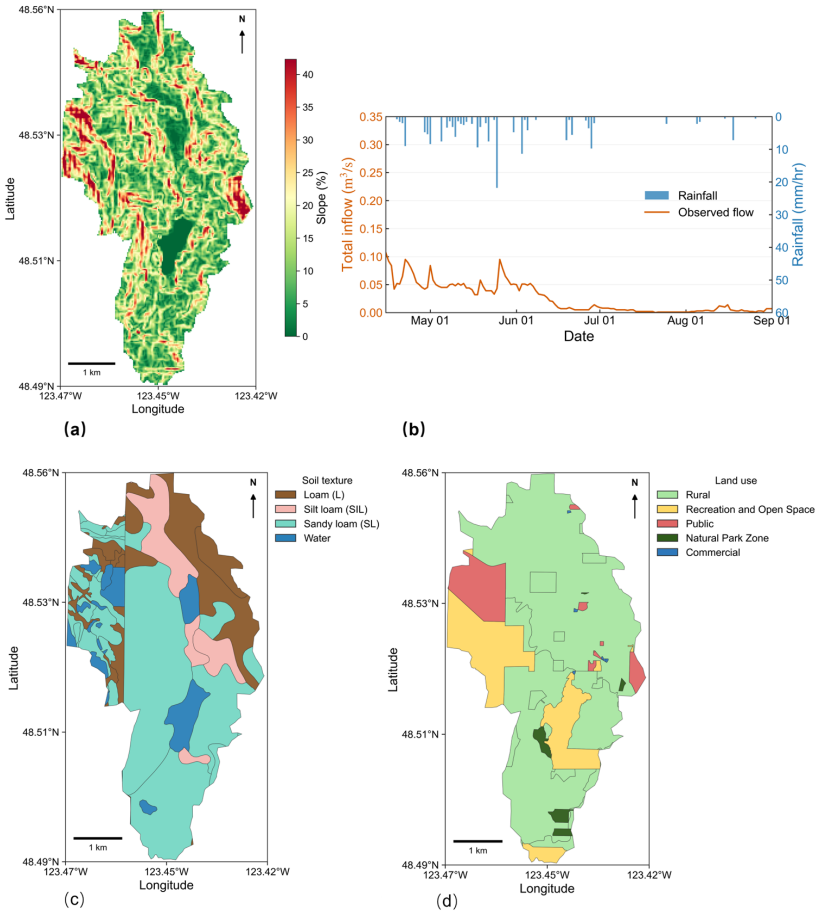


Figure 2. Tod Creek watershed case study model input datasets for Agentic SWMM workflow (**a**) DEM-derived slope (%); (**b**) 1984 daily rainfall (mm) and observed streamflow (m^3/s); (**c**) soil polygons; and (**d**) land-use polygons.

The Tod Creek database contained (a) DEM; (b) soil type polygons with texture attributes; (c) land-use polygons (zoning-style categories used as a land-use proxy); and (d) 1984 rainfall records (daily totals in units of mm). For demonstration purposes, a lumped model representation of Saanich was used to validate the end-to-end workflow. The outlet location was selected algorithmically on the DEM boundary (minimum elevation candidate) to obtain a consistent reference outlet or pour point for the stormwater simulation.

2.2. Methodology of Agentic SWMM System

This research compiles the SWMM command-line programme (v5.2.4) from the EPA SWMM source code on macOS and executes simulations using the standard command "swmm5 <inp> <rpt> <out>". Our SWMM Skill workflow automatically creates a separate run directory for each run, uniformly saving the input file (INP), report file (RPT), and binary output (OUT), and additionally, the swmm-runner Skill, thereby generating a manifest file as shown in the Supplementary File, Section S2 and Table S1. The biggest difference from the traditional user interface approach is that while users can see the modifications made in a GUI, this manifest, based on aiswmm (Version 0.6.4), is mandatory and written into the swmm-runner Skill. The manifest records the software version, key user configurations and parameter mappings, rainfall event information, and a summary of key results (such as continuity error, peak flow, etc.); thus, achieving transparency, reverse tracing, and auditability for each numerical experiment. There are two continuity errors that are separately tracked in the SWMM report file: the percentage mass-balance errors for runoff, and for flow routing. The magnitude of these two errors reflects the volumetric balance of surface runoff generation and the numerical stability of the hydraulic routing [37]. In addition, the same SWMM input file is run synchronously in the SWMM GUI, and the consistency of key indicators and resulting statistics is used as the basis for equivalence verification. This verification ensures that the automated command-line process and the GUI run are consistent in terms of numerical behaviour. After establishing this equivalence, we invoke the SWMM MCP via natural-language instructions that specify the original locations of the required datasets, trigger automated execution and call the plot Skill to generate the runoff comparison results shown in Section 3 of this paper.

2.2.1. aiswmm

We proposed a new agentic runtime, called aiswmm, for reproducible SWMM modelling. The aiswmm is used as an orchestrator that executes commands, reads and writes artefacts, and invokes external programmes in a controlled environment. As shown in Figure 3a, we detail the architecture of the aiswmm runtime and its six core components loop. The runtime loop is inspired by ReAct [38] but emits structured tool calls (OpenAI function calling) instead of parsed text actions [39,40]. This architecture fundamentally and strongly constrains the tool-calling capabilities of LLMs. In this runtime, we force a five-layer execution-control harness or rule system (Figure 3b) that determines when the agent may auto-execute and when it must pause and defer to the human modeller's selection [41,42]. Hence, the aiswmm runtime cannot execute arbitrary commands, and is specifically designed for auditable, reproducible, and traceable SWMM modelling environment.

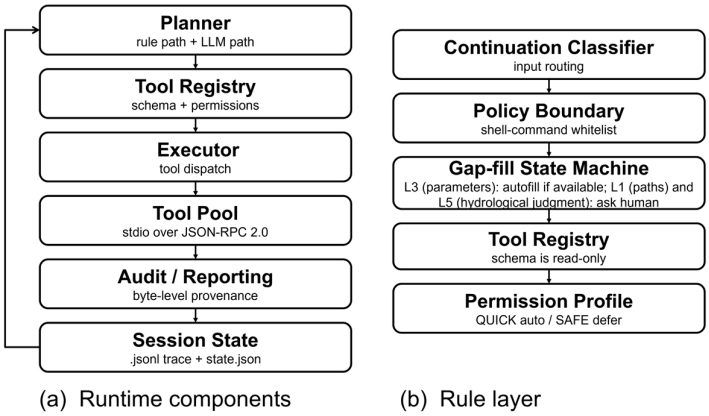


Figure 3. Conceptual design diagram for aiswmm runtime from structure perspective and safety control perspective for Agentic SWMM workflow. (**a**) The aiswmm runtime structure and (**b**) a five-layer rule system illustrating how the system controls tool calls and prevents uncontrolled execution.

Within this work, aiswmm does not replace SWMM, but instead standardizes the execution and bookkeeping around it. It is currently compatible with most LLMs through application programming interfaces keys; thus, enabling the analysis of complex tasks. In our case study, we use OpenAI GPT-5.5 Thinking [43], with the model identifier gpt-5.5-2026-04-23 as the primary model of aiswmm. In practice, aiswmm workflow is triggered by a natural language prompt. This will invoke the available Skills and orchestrate the MCP service to run the SWMM engine automatically and generate model outputs. The natural language prompt input format we recommend is shown in the Supplemental File Figure S2.

Note that, swmm-gis, swmm-runner, and swmm-plot in the prompts are Skill modules (tool bundles). The Skills formalize domain knowledge and modelling procedures as standard operating procedures; therefore, the workflow does not require high quality user prompts. The dependence on the overall prompting process is significantly reduced because the capability boundaries, parameter structure, and output formats are all "well designed" by the Skill/MCP interfaces. As a result, the LLM does not need to guess the user's meaning, which maximizes portability and reproducibility, as well as minimizes LLM hallucination.

2.2.2. Model Context Protocol

MCP is a protocol that allows AI models to call external tools, read files, run models, access databases, or execute certain local functions through a unified interface [44,45]. Traditional wrappers are typically designed for developers, and they mainly aim to simplify the invocation of specific functions, commands, or software interfaces. Compared with wrappers, MCP goes beyond simply encapsulating existing software functionality one-to-one. It provides a standardized protocol layer for Agentic AI; thus, supporting not only tool execution but also tool discovery [46,47]. The agentic runtime is able to identify available tools, understand their functional descriptions, read the input schema, and invoke different software capabilities through a unified interface. Under this design, SWMM, QGIS, Python scripts, and other external utilities are wrapped as tools. Each tool is discoverable, invokable, and orchestrable [47]. This enables cross-software communication and workflow coordination. Note that the current MCP layer runs on JSON-RPC 2.0 over stdio [32].

In this case study, mainly three MCP servers are used: (a) swmm-gis-mcp for DEM data preprocessing; (b) swmm-runner-mcp to generate inputs, run swmm5, extract quantities of interest and continuity metrics; and (c) swmm-plot-mcp to read INP/OUT artefacts and produce figures that conform to a fixed user's specification. Moreover, aiswmm can orchestrate MCP services after configuring the application programming interface keys

of any large language model. Therefore, users can use LLM to configure the required MCP services for different analysis scenarios, such as pollutant diffusion or water quality modules. In this study, we only encapsulated the peak flows we were interested in to demonstrate the overall effect of Agentic SWMM. The MCP services can be customized, recreated, and adjusted according to the needs of end users (such as adjusting quantity of interests like water quality or any function available in any software if they have an application programming interface). However, the current validation in Section 3 mainly supports peakflow outputs and should not be interpreted as a complete validation of all SWMM outputs. Table 1 shows an interface catalogue for a SWMM-focused MCP toolchain. Table 1 lists six MCP tools designed for the entire Agentic AI workflow.

Table 1. MCP tool interface specification for the Agentic SWMM workflow.

MCP Server	Tool Name	Required Inputs	Outputs	Units/Assumptions	Failure Modes
swmm-gis-mcp	qgis_package_final_layers	dem (path); outGeojson (path); outPng (path)	subcatchments.shp; flow.shp; slope_percent.tif; outfall.shp; overview.png;	Layers in EPSG:XXXX; outfall manually selected in QGIS/GRASS	Required output layer missing in QGIS dir; CRS mismatch; output path not writable
swmm-runner-mcp	swmm_run	inp (path); runDir (path)	JSON manifest; writes rpt/out/log/manifest	Flow units follow INP FLOW_UNITS (CMS → m^3/s)	swmm5 missing; INP errors; runDir not writable; nonzero return code
swmm-runner-mcp	swmm_peak	rpt (path)	JSON: node, peak, timeHHMM, source	peak in SWMM flow units; time in report timebase	RPT parse mismatch; node not found and return null; encoding issues
swmm-runner-mcp	swmm_continuity	rpt (path)	JSON: runoff table; routing table; continuityErrorPct	SWMM report units; continuity in percent	Continuity blocks missing; parser mismatch across SWMM versions
swmm-runner-mcp	swmm_compare	rpt (path); rpt2 (path)	JSON: aErr, bErr (headline continuity)	percent (%)	Same as continuity; different INPs/steps and return expected differences
swmm-plot-mcp	plot_rain_runoff_si	inp (path); out (path); outPng (path)	writes PNG; JSON: ok,outPng	Rain as mm/Δt (default 5 min); flow m^3/s if CMS	OUT missing/corrupt; TS_RAIN missing; swmmtoolbox missing; font fallback

2.2.3. Skills

Beyond MCP, Skills package scripts, configuration, and conventions for specific tasks or workflow (e.g., SWMM input/report/binary output files Via MCP, running SWMM, extracting outputs, and producing figures). This packaging ensures the workflow can be repeated and shared with minimal manual setup. From the perspective of the Agentic SWMM workflow, the Skills are an important complement to MCP, and they work as a standard operating procedure to tells the agent what procedure should be followed. This is because MCP primarily addresses tool discovery and invocation, rather than how the tools are called in a specific order; for example, from SWMM execution to result extraction, and then to plotting. Without Skills as a soft constraint, the quality of runtime results can easily lead to a high dependence on the quality of the prompt. We adopt Skills to render commands of "what to call, in what order, using what specifications; how to implement quality gates; and how to produce reproducible workpieces" into a reusable methodology. This is shown in Table 2.

Table 2. Agentic SWMM capability Skills catalogue (selected representative Skills, and full 14 Skills catalogue provided in the Supplementary File, Table S2).

Skill Name	Purpose	Key Scripts	RelatedMCP Server	Main Tools	Outputs
swmm-gis	GIS/DEM preprocessing (pour point/outlet selection) for reproducible SWMM workflows	scripts/find_pour_point.py	mcp/swmm-gis/server.js	gis_find_pour_point	GeoJSON pour point + DEM preview PNG
swmm-runner	Run swmm5 reproducibly and extract peak/continuity + write manifest.json	scripts/swmm_runner.py	mcp/swmm-runner/server.js	swmm_run, swmm_peak, swmm_continuity, swmm_compare	run directory: rpt/out/stdout/stderr/ manifest.json; metrics JSON
swmm-plot	Publication-grade plotting	scripts/plot_rain_runoff_si.py	mcp/swmm-plot/server.js	plot_rain_runoff_si	figure PNG/PDF
Swmm-experiment-audit	Auditing the Agentic SWMM runs	scripts/audit_run.py	mcp/swmm-experiment-audit/server.js	audit_run	experiment_provenance.json

In the Agentic SWMM scenario, the Skill layer:

- orchestrates the corresponding MCP to autogenerate the SWMM inputs (INPs) file;
- runs the swmm engine, and extracts indicators such as peaks and continuity;
- performs conservation/continuity checks and GUI with command-line interface (CLI) equivalence comparisons;
- includes built-in plotting specifications (units, axes, fonts, and window clipping, etc.) and automatic repair rules for common pitfalls;
- enforces standardized run directories and manifests (version, hash, parameter mapping, key results, plots, etc.).

Therefore, simply relying on MCP can easily degenerate into a set of "bare tools", which can be called but are difficult to manage and reproduce stably over a long period of time. This also makes it difficult to quickly port the entire workflow. Skill, however, organizes interface capabilities into reproducible experimental packages that are auditable, shareable, and maintainable; thus, enabling workflows to run repeatedly and be reliably reused with minimal manual setup. As well, in the data preprocessing part, because DEM-based preprocessing is often required for hydrologic model setup (e.g., identifying a pour point and delineating subcatchments) and EPA SWMM does not provide these GIS functions, we implemented a dedicated swmm-gis Skill to automate this step. We use two methods from the DEM to find candidate outlets: one is the lowest point on the boundary, and the other is the point with the largest flow accumulation. This Skill saves the pour point within the coordinate system as a reproducible geographic file, which can be directly used for SWMM configuration.

2.3. Verification, Reproducibility and Repeatability

2.3.1. Verification-First Provenance Generation

The aiswmm runtime automatically generates two complementary provenance files via Skill during each SWMM run, as shown in Table 3: (a) a runner-level manifest.json (written directly by the swmm-runner Skill upon SWMM execution) and (b) an audit-layer provenance file experiment_provenance.json (written by the swmm-experiment-audit Skill in Table 1). Together, the two files record the Secure Hash Algorithm 256-bit (SHA-256) hashes of every input INP, binary OUT, and report RPT artefact, together with the outcome of a deterministic quality assurance gate suite (including continuity errors, peak metric integrity, SWMM return code, and eight SWMM-specific screening rules). This ensures that every workflow step can be traced back to a byte-identifiable run (more details are provided in the Supplementary File Sections S2 and S4 shows an example).

Table 3. Two provenance files written automatically on every Agentic SWMM run. Note that this is a simplified table, and full lists are in the Supplementary File Section S2.

File Type	Group	What Is Included
manifest.json	files	INP path + SHA-256 hash, SWMM5 version + return code, absolute paths of rpt, out, stdout, stderr
manifest.json	results	Peak inflow (node, value, time, report-section source) and two continuity errors with mass-balance sub-tables
experiment_provenance.json	run context	Schema version, workflow mode, natural-language objective
experiment_provenance.json	environment	Git branch + HEAD commit + working-tree status, Python and SWMM versions
experiment_provenance.json	artefact index	Every run file by id, role, paths, SHA-256, producer, downstream use
experiment_provenance.json	verification	Quality Assurance checks (pass/fail), eight SWMM screening rules, human-decision slot

Because the Skill-level Python scripts in every Agentic SWMM run generate both files automatically and mandatorily, they are produced independently of the agent runtime (aiswmm, Codex, OpenClaw, Claude Code, or any other orchestrators). This independent mechanism makes sure that the provenance files are documented regardless of which agent triggers the workflow.

2.3.2. Verification of DEM-Based Preprocessing

Because EPA SWMM does not provide necessary GIS preprocessing utilities, we implemented an automated pour-point detection routine as an independent SWMM–GIS MCP and Skill (as shown in Tables 1 and 2) to reproduce key preprocessing steps in a traceable, script-driven workflow, and to verify consistency against a conventional GUI-based approach (PCSWMM Version 7.7). Its pseudocode implementation logic has been included in Section S3 of the Supplementary File.

2.3.3. Repeatability: Equivalence Validation Between CLI and MCP

In this section, we tested whether the results produced by the packaged MCP service are numerically consistent with a direct SWMM CLI execution under identical model inputs. Our baseline model is a lumped model with a timestep of 5 min, representing the Chicago design storm (which is the same SWMM input structure and parameters for hydrology/hydraulics). Note that Chicago design storm is a synthetic hyetograph with a single peak placed at a prescribed fraction r of the storm duration [48,49]. Two different tests were performed. In Test 1, a total of 30 parameter configurations were conducted where the Chicago storm-shape parameter r (time to peak $r \in [0.30,\ 0.50]$) is increased from 0.30 by an increment of (0.50–0.30)/30 in each run while holding all other model settings constant [50]. Similarly, test 2 involves 30 parameter configurations where the percentage of imperviousness (I) is varied by a factor ranging from $[0.9,\ 1.1]$ while holding all other model settings constant. Note that these parameter configurations are aimed at verifying the equivalence between the CLI and MCP paths, not a hydrological result. The parameter ranges and sample size only need to cover diverse, realistic SWMM inputs. Because the equivalence is verified at the byte level, the conclusion does not depend on their exact values. Since there are 30 configurations in each test, and we run the test in two different ways (CLI and MCP), there is a total of 120 SWMM runs. We then compare the degree of agreement of the computed peak flow at the outlet for each run using R^2 and the Nash and Sutcliffe Efficiency (NSE). Theoretically, if the execution between CLI and MCP are

equivalent, then R^2 and NSE should both be exactly 1, as the SHA-256 values of the SWMM input and output files (Section 2.3.1) should be byte-identical (excluding timestamp fields).

2.3.4. Reproducibility: SWMM GUI vs. Agentic SWMM

To verify cross-system consistency, we manually recreated the baseline model in the SWMM GUI and note that the baseline model is generated by the Agentic SWMM from the user prompt. After the manual configuration, we ran the SWMM GUI and compared the results with the Agentic AI generated baseline model to verify whether its ability to automate configuration using natural language was consistent with manual configuration.

3. Results

3.1. Verification of Data Preprocessing Tools

Because SWMM itself does not integrate DEM-related GIS preprocessing functions, we implemented "automatic pour point identification" as an independent SWMM-GIS Skill to extract key spatial features before modelling. Specifically, we used QGIS Processing (grass:r.watershed invoked through Agentic SWMM) to compute D8 flow direction and accumulation across the DEM and select the stream cell with the maximum upstream accumulation as a candidate pour point, which is automatically output as a point.

Figure 4a shows the DEM visualization generated by the Agentic SWMM–GIS Skill. The automatically identified candidate pour point is shown as a red mark. The right panel visualization of the pour point location in the same area was generated in PCSWMM using manually identified pour points from [49]. The results show that the two methods exhibit high consistency (with 0.88% error and within approximately 7.5 DEM pixels) in spatial location and topographical consistency, indicating that Skill can stably produce pour point results similar to those of the traditional GUI workflow.

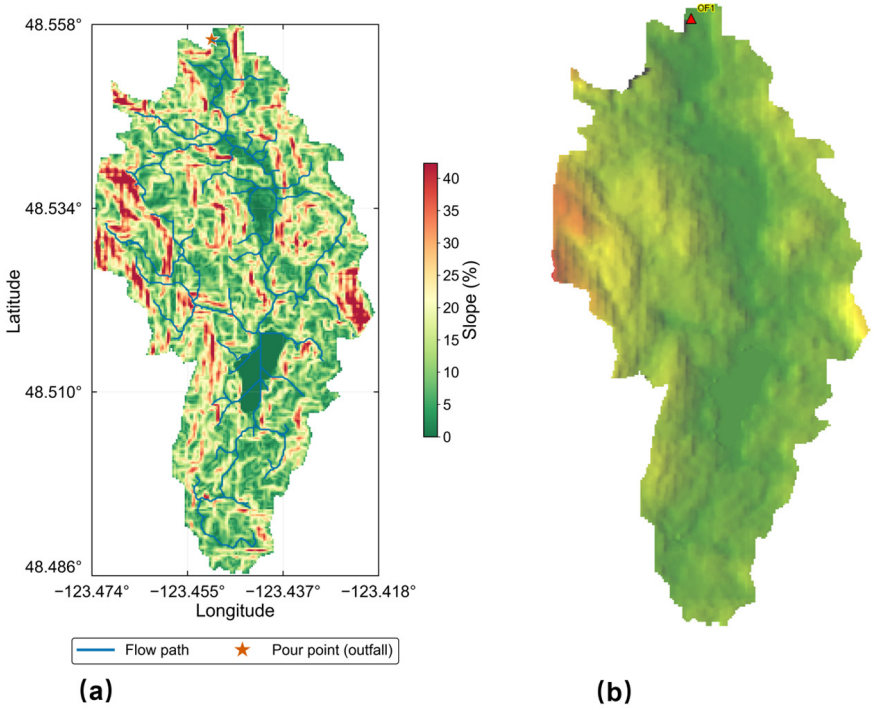


Figure 4. The comparison of the boundary outlet detection and plot (**a**) pour-point generation via Agentic SWMM via swmm-gis Skill; and (**b**) the pour-point (red triangle) generated with the PCSWMM graphical user interface.

3.2. Repeatability: SWMM CLI and SWMM MCP

The CLI execution results are shown as black solid lines and the MCP results are orange dashed lines overlaid on top. The complete set of 60 pairs simulation plots is provided in the Supplementary File, Figures S3 and S4, and all the models .out files passed the audit test and SHA-256 byte-identical. Figure 5a,b, and Figures S3 and S4 in the Supplementary File show that the evaluation metrics, R^2 and NSE, are both identically 1.0 (SHA-256 byte-identical, and tolerance = 0), demonstrating consistent outputs across all 60 simulation pairs. Therefore, we can reasonably conclude that the model called by the MCP server and the model used via the command line through the CLI port are consistent, and that the core hydrological computation based on aiswmm is entirely based on the SWMM engine rather than the large language model itself. Furthermore, the SHA-256 hashes of the manifest.json files for both tests are all identical; the return code is all 0, and max | ΔQpeak | = 0.0, proving that the CLI and MCP have completely identical INPs (same hash). Further indications of the robustness of the MCP service are shown in Table S2 in the Supplementary File.

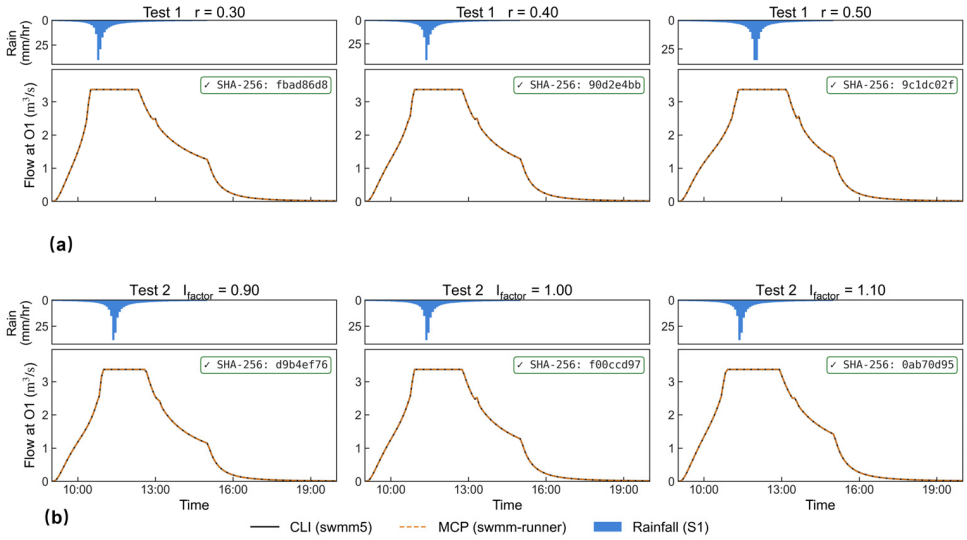


Figure 5. Computed hydrographs at the outlet from six parameter configurations, demonstrating the byte-identical results (excluding the timestamps) between the Model Context Protocol (orange dashed line) and command-line interface (black line) execution modes: (**a**) the three parameter configurations result pairs for Test 1 and (**b**) the three parameter configurations result pairs for Test 2 using the methodology described in Section 2.3.2. The complete set of results of 60 pairs (120 runs) is provided in the Supplementary File, Figures S3 and S4.

3.3. Reproducibility: Agentic SWMM and SWMM GUI

Using identical INPs, we conducted equivalence verification on two execution paths. Figure 6a shows the GUI display obtained through manual parameter configuration and execution using the traditional SWMM graphical user interface. Figure 6b shows the workflow from Figure 1, entirely driven by aiswmm through a natural language dialogue, utilizing the SWMM Skill and MCP toolchain. Agentic AI automatically generates/verifies inputs from the original dataset, calls swmm5 to execute simulations, and extracts key outputs. As the comparison in Figure 6 shows, the peak flow results obtained from both paths are completely identical, indicating that the aiswmm-driven SWMM modelling approach is consistent with SWMM GUI execution. Not surprisingly, the two approaches yield identical peaks to three decimal places, but they are not bit identical. This is because of the subtle differences in node naming, junction name, and modelling procedures during manual modelling that can lead to different SHA-256 values. This indicates that in the

current AI age, the Agentic SWMM workflow can produce peak values that are identical with SWMM GUI execution (to within three decimal places) thereby, greatly expanding the model's usability. To test the scalability of the Agentic SWMM workflow's generic modelling process beyond the Tod Creek case study, we additionally validated its core modules on three external SWMM benchmarks:

(i) the Tecnopolo January-1994 prepared-input case from [51,52];
(ii) the TUFLOW–SWMM Module 03 coupled benchmark [53]; and
(iii) the Generate_SWMM_inp INP-derived adapter case [54].

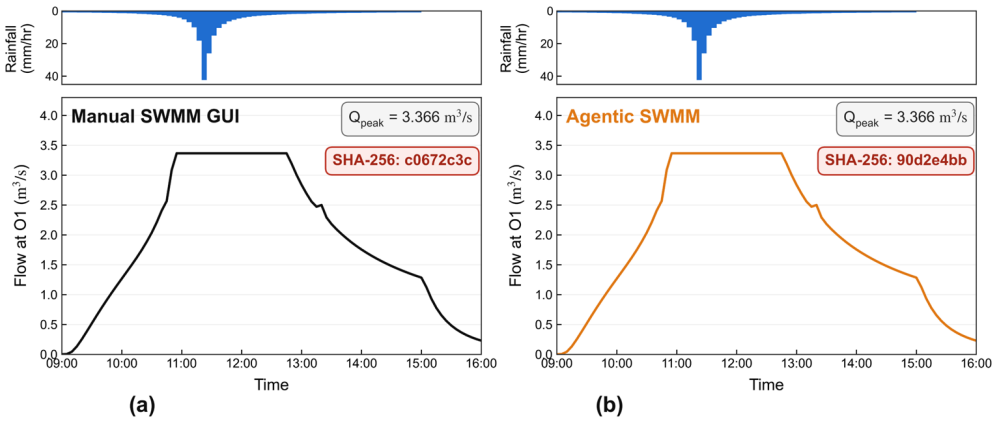


Figure 6. (**a**) Equivalence between manual SWMM graphical user interface execution and (**b**) natural language-driven Agentic SWMM execution at the Tod Creek watershed outlet (node O1, 25 May 1984 Chicago design storm with peak-time r = 0.40). Peak inflow matches to three significant figures.

In each benchmark, we audited the artefacts: manifest.json and experiment_provenance.json against the published reference outputs. The validation scope and conclusions are reported in the Supplementary File, Section S5.

3.4. Verification-First Provenance

Figure 7 is an additional audit example showing the SHA-256 comparison between Figure 7a,b. The inconsistencies observed are caused by LLM scripts at the generation level. Both panels were labelled as r = 0.40 and appear visually consistent, while the SHA-256 prefixes are different ('90d2e4bb. . .' vs. '44667e8f. . .').

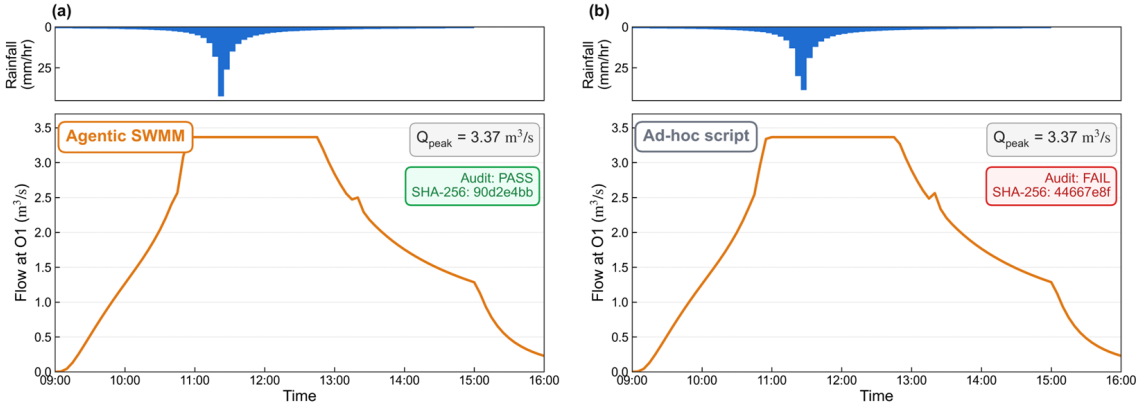


Figure 7. The auditing process detected a Secure Hash Algorithm 256-bit byte-level inconsistency caused by LLM hallucination. (**a**) Byte-level consistency check between the original Model Context Protocol execution mode and command-line interface; and (**b**) inconsistency between the LLM-generated Python script way and command-line interface execution results.

After reviewing the manifests, we found that two independently generated Python scripts selected different nearest parameter values at the Tod Creek watershed outlet (node O1, 25 May 1984 Chicago design storm with peak-time $r = 0.40$). Figure 7a selected $r = 0.39655$, and Figure 7b used $r = 0.40345$ (details are provided in the Supplementary File, Section S6). Both values were rounded to $r = 0.40$ in the figure labels, which masked the underlying difference. This finding shows that the two manifests (provenance) are not only a passive record of model execution but can also detect inconsistencies before they are entered into the final scientific record, hence, we call this a verification-first contract. This result also proves our MCP is more stable than an ad hoc Python script. There are further auditing/testing case reports that can be found in the Supplementary File, Table S3 (unit inconsistencies) and Table S4 (different observation times).

4. Discussion

4.1. The Explainable and Auditable Results

The interpretability of our results does not arise from Agentic AI itself or from the underlying LLM. Rather, interpretability is grounded in the physical model and its explicit, inspectable input/output files. In our framework, SWMM constitutes the computational core: all hydrologic calculations are executed by SWMM, while the Agentic layer is responsible for orchestration, tracking, and standardized output generation. To mitigate LLM hallucinations, we first established the full end-to-end workflow prototype and then performed manual checks to confirm consistency between automated and reference runs.

Not surprisingly, all LLM-driven workflows, including aiswmm and other agentic runtimes such as Codex, Claude Code, and OpenClaw, inherently translate natural language prompts into tool calls in non-deterministic ways [55,56], which may lead to subtle but real inconsistencies in the executed workflow [57]. This is a well-documented characteristic of LLM-driven systems. Therefore, we designed a verification-first provenance mechanism (as described in the two audit files included in Section 2.3.1 and Supplementary File Section S2) that records the identifying source of each execution run using SHA-256. The two JSON files ensure that the key information of each run is documented and can be traced, facilitating reproducibility.

The aiswmm runtime mechanism also makes the entire workflow become auditable and reproducible. The aiswmm runtime loop shown in Figure 3a was built specifically for the SWMM engine, and Figure 3b indicates when the runtime should stop and when to ask users for more input, rather than executing arbitrarily. Beyond the two audit files, as shown in Figure 3a, the runtime also produces a session_state.json, capturing the agent loop state (goal, plan, tool calls, failures) for cross-session referencing.

4.2. Minimized LLM Model Hallucinations in Hydrology Domain

LLMs must be applied to stormwater hydrology in a safe manner [58,59], and a purely black-box approach to using Generative AI may violate physical laws due to LLM hallucinations [20,21,40,60–62]. In our work, as shown in Figure 1, the LLM hallucinations are minimized through three layers of defence: (a) the runtime layer that places a hard constraint on the LLM's action space through a registered, schema-validated tool list; (b) the MCP interface layer that declares structured tool signatures that the LLM can reason about; and (c) the Skill layer that encodes SWMM-specific domain constraints; for example, 'the peak is always read from Node Inflow Summary, never from Node Depth Summary'. Hence, the runtime will execute the deterministic Python implementations rather than those temporarily generated by the LLM. The aiswmm runtime orchestrates which Skill to invoke, and then the Skill decides how the SWMM operation is performed. In this work, we also packaged the entire Agentic SWMM workflow as a standardized Skill with mandatory

artefacts and quality gates (e.g., continuity diagnostics and consistent quantity-of-interest definitions), improving portability and substantially reducing the errors caused by LLM hallucination. In practice, while poor prompting can lead to variability in how non-expert users specify tasks [63], the Skill constrains execution to validated procedures and specifications, thereby minimizing the workflow failures caused by ambiguous or low-quality prompts. Also, Skills and MCP are portable [32]; they can run with any agentic runtimes (Codex, Claude code, Hermes, OpenClaw, and, etc.). However, these constraints are still soft, and they may still operate at the prompt-injection level for the AI agent or LLM [63,64].

4.3. The Transferability and Irreplaceability of Expert Experience

The goal is not to replace SWMM or the modellers, but to make SWMM-based modelling easier to reproduce, audit, remember, and trust. In the current AI age, we found that the hydrological modelling process and part of the human-specific modelling experience can be standardized through Skills (Table 2), and the MCP (Table 1) can provide available tools for the agentic runtime. However, expert involvement remains significant for verifying auditing reports and assessing their physical meaning. Agentic SWMM helps automate tedious tasks, but it does not replace expert judgement. Hydrologists' experience and judgement can and must be extended to the existing aiswmm runtime as needed (as shown in Figure 3b), which means that we not only need to verify the report but also need to track the computational behaviour of the agentic runtime [65]. We can add new deterministic SWMM diagnostic rules to make the agentic runtime traceable, transparent, and trustworthy. However, a trade-off between flexibility and stability needs to be considered and properly designed [42]. For example, should shell permissions be granted to the runtime [66,67]? Should we relax the permissions to allow the LLM to automatically complete extremely important urban drainage network information, such as invert elevation? Should we let the model run first and then review the record afterward [68]? We need to balance this tradeoff within an acceptable level of LLM hallucination [69] for hydrological field.

4.4. Limitation

First, the Agentic SWMM workflow is still under active development and testing. This work focuses on the operational layer of SWMM-based modelling rather than its theoretical formulation. We built the Agentic SWMM workflow around the SWMM engine, and none of the SWMM governing equations or numerical schemes have been modified; all hydraulic and hydrologic computations remain those of EPA SWMM 5.2.4. We do not seek to replace SWMM or software assembly processes but rather attempt to propose a new pipeline for hydrological modelling in the current Agentic AI age in a safe manner.

Second, in this latest research, we found that building the urban pipeline network system presents some challenges. This is not only due to the scarcity of real urban pipeline network data, but also because expert experience is needed to estimate parameters, such as those in the parameter mapping tables. Therefore, modellers may need to participate in re-customizing the existing Skills and MCP. As the hallucination of LLM comes from its generation mechanism [70] and training method, it cannot be eliminated just by prompt injection or changing the hyperparameters (e.g., temperature) [40]. The runtime, MCP tool whitelist, parameter schema, audit logs, and continuity checks essentially restrict the free-flowing behaviour of LLM within an inspectable, traceable environment, thereby minimizing and tracking errors to satisfy the needs of operational hydrology or computational hydrology.

Third, aiswmm is a memory-informed runtime (we developed a memory layer for it). It can not only perform general text-based lesson memory, similar to Hermes [30], but also implements a SWMM domain-specific quantitative memory layer, as well as bidirectional

connectivity with the audit layer. Therefore, logically, it forms a closed loop: aiswmm memory layer → aiswmm decision layer → SWMM execution → audit → aiswmm memory layer. The data from the memory layer are stored as plain text files (JSONL, YAML, and Markdown), with SQLite used only as an optional index for large stores rather than as the primary store. The audit layer can now produce Markdown format documents, thereby completing subsequent analysis using Obsidian [71]. However, the verification of the effectiveness for this memory informed structure needs time and real SWMM project data support. Therefore, although we have built this memory-informed framework in aiswmm v0.7.1, it has not yet been sufficiently validated. Therefore, it was not completely demonstrated in the paper.

Finally, uncertainty quantification was not demonstrated in the main results. The open-source repository already provides preliminary support; however, an uncertainty quantification (such as a watershed-entropy-based metric [72]) module needs to be incorporated into the current workflow. Additionally, Skills and MCP need to be established and extended for subcatchment partitioning.

5. Conclusions

The overall Agentic SWMM workflow described in this paper suggests that Agentic AI offers great operational potential for hydrological modelling in place of traditional GUI modes of parameter modification. Agentic AI runtime can orchestrate tool-based workflows (via Skills and MCP) and potentially across multiple software systems to execute, verify, and reproduce hydrological modelling tasks beyond conversational responses. This paper details how the aiswmm architecture, leveraging mainstream MCP services and Skills, can potentially replicate the previously manually driven GUI operation process. This application-level implementation greatly facilitates subsequent researchers or users unfamiliar with model deployment and operation to configure and audit their own hydrological models in a very short time.

Our novelty lies in proposing an automated, auditable, and reproducible Agentic SWMM framework for stormwater modelling. Our work is not only in designing the MCP tools and Skills for SWMM, but also in verifying the equivalence between Agentic SWMM and manual SWMM GUI workflow. Also, the verification-first mechanism can better help researchers understand the performance of Agentic AI and use AI in a safe manner for hydrological modelling in the current AI age.

The contributions of this work are not only in making the hydrological modelling configuration process simpler and more trustworthy, but also in providing a suitable and appropriate framework for leveraging the capabilities of Agentic AI in the hydrological modelling domain. Potentially, transferring the orchestration capabilities of Agentic AI into our field can better support the understanding of hydrological modelling processes. This is a practical attempt to integrate agentic AI with hydrological modelling, that is both automated and auditable, rather than treating automation and reproducibility as competing goals. Future work will focus on letting the Agentic SWMM workflow learn from its own audited history (memory-informed), turning past modelling experience into guidance for new watersheds.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/aieng1010005/s1, Figure S1: aiswmm (v0.6.4 alpha) architecture and CLI as well as model informed memory layer for future research, and flowchart (by GPT image 2.0) on GitHub; Table S1: Run Manifest Schema for Reproducible SWMM Execution; Figure S2: The natural language prompt input format; Table S2: Complete OpenClaw SWMM capability catalog: 14 Skills, 11 MCP servers; Figure S3: The NSE, R^2 values for Scenario A, compared with SWMM CLI and SWMM MCP via aiswmm; Figure S4: Computed hydrographs at the outlet for the upper and lower

limit for 30 runs with hyetograph shown in blue on the left axis (inverted) for (a) Test 1 top panel and (b) Test 2 downside panel. CLI results are shown as black solid lines and MCP results are orange dashed lines overlaid on top; Table S3: More Audit_Tamper_Comparison; Table S4: The issue of unit conversion was captured from the auditing note.

Author Contributions: Conceptualization, Z.Z. and C.V.; methodology, Z.Z. and C.V.; software, Z.Z.; validation, Z.Z. and C.V.; formal analysis, Z.Z.; investigation, Z.Z. and C.V.; resources, C.V.; data curation, Z.Z.; writing—original draft preparation, Z.Z.; writing—review and editing, Z.Z. and C.V.; visualization, Z.Z. and C.V.; supervision, C.V.; project administration, C.V.; funding acquisition, C.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Natural Sciences and Engineering Research Council of Canada (RGPIN-2022-04352) and the University of Victoria.

Data Availability Statement: Agentic SWMM is a pre-1.0 project under active development. As of v0.6.4, Agentic SWMM provides 14 Skills, 11 MCP servers, and 45 MCP tools supporting raw GIS preprocessing, parameter mapping, climate/rainfall preprocessing, pipe-network assembly (Alpha), model building (INP), SWMM execution, plotting, model calibration (Alpha), model sensitivity and uncertainty analysis, experiment audit, and cross-run memory informed modelling. Four of these Skills and six MCP tools were exercised in the equivalence-verification and validation experiments reported in this paper. All source code is released under the MIT licence. Source codes: https://github.com/Zhonghao1995/agentic-swmm-workflow (accessed on 2 June 2026). Archived release (Zenodo): https://doi.org/10.5281/zenodo.20337281 (accessed on 2 June 2026). PyPI version: aiswmm (0.6.4). Docker image (GHCR): ghcr.io/zhonghao1995/agentic-swmm-workflow:v0.6.4 (accessed on 2 June 2026). Git tag: v0.6.4 (commit a0650e9). LLM Model: gpt-5.5-2026-04-23. Live replay demo: https://aiswmm.com/demo/ (accessed on 2 June 2026).

Acknowledgments: The Authors are grateful to the Reviewers for their insightful suggestions that greatly improved the manuscript. The Authors would also like to thank NSERC for funding this work and to CHI Inc in Guelph, Ontario, Canada for providing the PCSWMM software used in this work.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Abbreviations

The following abbreviations are used in this manuscript:

EPA	Environmental Protection Agency
SWMM®	Storm Water Management Model
MCP	Model Context Protocol
CLI	command-line interface
GUI	graphical user interface
DEM	digital elevation model
GIS	Geographic Information System
INP/RPT/OUT	SWMM input/report/binary output files
LLM	large language model
SHA-256	Secure Hash Algorithm 256-bit

References

1. Dobson, B.; Jovanovic, T.; Alonso-Álvarez, D.; Chegini, T. SWMManywhere: A Workflow for Generation and Sensitivity Analysis of Synthetic Urban Drainage Models, Anywhere. *Environ. Model. Softw.* **2025**, *186*, 106358. [CrossRef]
2. Pichler, M. Swmm_api: A Python Package for Automation, Customization, and Visualization in SWMM-Based Urban Drainage Modeling. *Water* **2025**, *17*, 1373. [CrossRef]
3. Niazi, M.; Nietch, C.; Maghrebi, M.; Jackson, N.; Bennett, B.R.; Tryby, M.; Massoudieh, A. Storm Water Management Model: Performance Review and Gap Analysis. *J. Sustain. Water Built Environ.* **2017**, *3*, 04017002. [CrossRef] [PubMed]
4. Gan, T.; Tarboton, D.G.; Dash, P.; Gichamo, T.Z.; Horsburgh, J.S. Integrating Hydrologic Modeling Web Services with Online Data Sharing to Prepare, Store, and Execute Hydrologic Models. *Environ. Model. Softw.* **2020**, *130*, 104731. [CrossRef]

5. Chawanda, C.J.; George, C.; Thiery, W.; van Griensven, A.; Tech, J.; Arnold, J.; Srinivasan, R. User-Friendly Workflows for Catchment Modelling: Towards Reproducible SWAT+ Model Studies. *Environ. Model. Softw.* **2020**, *134*, 104813. [CrossRef]
6. Ahmadi, H.; Scott, D.; Sample, D.J.; Shahed Behrouz, M. Automated Calibration of SWMM for Improved Stormwater Model Development and Application. *Hydrology* **2025**, *12*, 129. [CrossRef]
7. Sun, R.; Hernández, F.; Liang, X.; Yuan, H. A Calibration Framework for High-Resolution Hydrological Models Using a Multiresolution and Heterogeneous Strategy. *Water Resour. Res.* **2020**, *56*, e2019WR026541. [CrossRef]
8. López-Ballesteros, A.; Srinivasan, R.; Senent-Aparicio, J. Introducing MapSWAT: An Open Source QGIS Plugin Integrated with Google Earth Engine for Efficiently Generating Ready-to-Use SWAT+ Input Maps. *Environ. Model. Softw.* **2024**, *179*, 106108. [CrossRef]
9. Schaffhauser, T.; Garijo, D.; Osorio, M.; Bittner, D.; Pierce, S.; Vargas, H.; Disse, M.; Gil, Y. A Framework for the Broad Dissemination of Hydrological Models for Non-Expert Users. *Environ. Model. Softw.* **2023**, *164*, 105695. [CrossRef]
10. Dobson, B.; Alonso-Álvarez, D.; Chegini, T. SWMManywhere: Synthesise Urban Drainage Network Models Anywhere in the World. *J. Open Source Softw.* **2025**, *10*, 7729. [CrossRef]
11. McDonnell, B.; Ratliff, K.; Tryby, M.; Wu, J.; Mullapudi, A. PySWMM: The Python Interface to Stormwater Management Model (SWMM). *J. Open Source Softw.* **2020**, *5*, 1. [CrossRef]
12. Blank, J.; Deb, K. Pymoo: Multi-Objective Optimization in Python. *IEEE Access* **2020**, *8*, 89497–89509. [CrossRef]
13. Sakar, C.; Tsukanov, K.; Schwartz, N.; Moreno, Z. A Physics-Informed Neural Network Workflow for Forward and Inverse Modeling of Unsaturated Flow and Root Water Uptake from Hydrogeophysical Data. *J. Hydrol.* **2026**, *665*, 134675. [CrossRef]
14. Qi, X.; de Almeida, G.A.M.; Maldonado, S. Physics-Informed Neural Networks for Solving Flow Problems Modeled by the 2D Shallow Water Equations without Labeled Data. *J. Hydrol.* **2024**, *636*, 131263. [CrossRef]
15. Feng, Z.; Zhang, J.; Niu, W. A State-of-the-Art Review of Long Short-Term Memory Models with Applications in Hydrology and Water Resources. *Appl. Soft Comput.* **2024**, *167*, 112352. [CrossRef]
16. Panda, C.; Panda, K.C.; Singh, R.M.; Singh, R.; Singh, V.P. A Generalised Hydrological Model for Streamflow Prediction Using Wavelet Ensembling. *J. Hydrol.* **2025**, *655*, 132883. [CrossRef]
17. Baniya, S.; Maity, D. A Comprehensive Review of Theoretical Concepts and Advancements in Physics-Informed Neural Networks with Applications in Structural Engineering. *Artif. Intell. Rev.* **2026**, *59*, 101. [CrossRef]
18. Li, J.; Wu, G.; Zhang, Y.; Shi, W. Optimizing Flood Predictions by Integrating LSTM and Physical-Based Models with Mixed Historical and Simulated Data. *Heliyon* **2024**, *10*, e33669. [CrossRef] [PubMed]
19. De La Fuente, L.A.; Ehsani, M.R.; Gupta, H.V.; Condon, L.E. Toward Interpretable LSTM-Based Modeling of Hydrological Systems. *Hydrol. Earth Syst. Sci.* **2024**, *28*, 945–971. [CrossRef]
20. Garzón, A.; Kapelan, Z.; Langeveld, J.; Taormina, R. Machine Learning-Based Surrogate Modeling for Urban Water Networks: Review and Future Research Directions. *Water Resour. Res.* **2022**, *58*, e2021WR031808. [CrossRef]
21. Reichstein, M.; Camps-Valls, G.; Stevens, B.; Jung, M.; Denzler, J.; Carvalhais, N. Prabhat Deep Learning and Process Understanding for Data-Driven Earth System Science. *Nature* **2019**, *566*, 195–204. [CrossRef]
22. Zhang, Z.; Valeo, C. Low Impact Development Technologies for Mitigating Climate Change: Summary and Prospects. *Natl. Sci. Open* **2024**, *3*, 20230025. [CrossRef]
23. Stagge, J.H.; Rosenberg, D.E.; Abdallah, A.M.; Akbar, H.; Attallah, N.A.; James, R. Assessing Data Availability and Research Reproducibility in Hydrology and Water Resources. *Sci. Data* **2019**, *6*, 190030. [CrossRef]
24. Hutton, C.; Wagener, T.; Freer, J.; Han, D.; Duffy, C.; Arheimer, B. Most Computational Hydrology Is Not Reproducible, so Is It Really Science? *Water Resour. Res.* **2016**, *52*, 7548–7555. [CrossRef]
25. Karniadakis, G.E.; Kevrekidis, I.G.; Lu, L.; Perdikaris, P.; Wang, S.; Yang, L. Physics-Informed Machine Learning. *Nat. Rev. Phys.* **2021**, *3*, 422–440. [CrossRef]
26. Navaie, K. From Rights to Runtime: Privacy Engineering for Agentic AI. *AI Mag.* **2025**, *46*, e70036. [CrossRef]
27. Acharya, D.B.; Kuppan, K.; Divya, B. Agentic AI: Autonomous Intelligence for Complex Goals-A Comprehensive Survey. *IEEE Access* **2025**, *13*, 18912–18936. [CrossRef]
28. Steinberger, P. Introducing OpenClaw. Available online: https://openclaw.ai/blog/introducing-openclaw (accessed on 19 May 2026).
29. OpenClaw Contributors Openclaw/Openclaw [GitHub Repository]. Available online: https://github.com/openclaw/openclaw (accessed on 19 May 2026).
30. Nous Research Hermes Agent: The Agent That Grows with You. Available online: https://github.com/NousResearch/hermes-agent (accessed on 19 May 2026).
31. Xu, W.; Huang, C.; Gao, S.; Shang, S. LLM-Based Agents for Tool Learning: A Survey. *Data Sci. Eng.* **2025**, *10*, 533–563. [CrossRef]
32. Anthropic Introducing the Model Context Protocol. Available online: https://www.anthropic.com/news/model-context-protocol (accessed on 19 May 2026).

33. Aria, M.; Cuccurullo, C. Bibliometrix: An R-Tool for Comprehensive Science Mapping Analysis. *J. Informetr.* **2017**, *11*, 959–975. [CrossRef]
34. Pawar, M.S. A Practical Evaluation of Self-Hosted N8n for Secure and Scalable Workflow Automation. *Int. J. Sci. Res. Eng. Manag.* **2025**, *9*, 1–9. [CrossRef]
35. Ramadhani, A.; Yantoro, M.D.; Akmal, M.F.; Mahfud, M.; Fauzi, F. Chatbot Otomatis Dengan N8n Dan Ai Untuk Analisis Data Dan Pelaporan Hasil. *J. Ris. Tek. Komput.* **2025**, *2*, 18–23. [CrossRef]
36. Venkiteela, P. N8n- An Open-Source Workflow Automation for Enterprise Integration and AI Orchestration. *Int. J. Comput. Appl.* **2025**, *187*, 1–11. [CrossRef]
37. Rossman, L.A.; Simon, M. *Storm Water Management Model User's Manual Version 5.2. EPA/600/R-22/030*; U.S. Environmental Protection Agency: Washington, DC, USA, 2022.
38. Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; Cao, Y. React: Synergizing Reasoning And Acting In Language Models. In Proceedings of the 11th International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, 1–5 May 2023.
39. OpenAI Function Calling and Other API Updates. Available online: https://openai.com/index/function-calling-and-other-api-updates/ (accessed on 19 May 2026).
40. Zhang, Z.; Valeo, C. Fuzzy-Based Input Method for Uncertainty Quantification in a Deterministic Model Comparison with ChatGPT for Peak Flow Prediction. *J. Hydrol. X* **2025**, *28–29*, 100208. [CrossRef]
41. Dai, J.; Pan, X.; Sun, R.; Ji, J.; Xu, X.; Liu, M.; Wang, Y.; Yang, Y. Safe Rlhf: Safe Reinforcement Learning From Human Feedback. In Proceedings of the 12th International Conference on Learning Representations, ICLR 2024, Vienna, Austria, 7–11 May 2024.
42. Dahlgren Lindström, A.; Methnani, L.; Krause, L.; Ericson, P.; de Rituerto de Troya, Í.M.; Coelho Mollo, D.; Dobbe, R. Helpful, Harmless, Honest? Sociotechnical Limits of AI Alignment and Safety through Reinforcement Learning from Human Feedback. *Ethics Inf. Technol.* **2025**, *27*, 28. [CrossRef] [PubMed]
43. OpenAI GPT-5.5 Thinking (Model Snapshot Gpt-5.5-2026-04-23). Available online: https://platform.openai.com/docs/models/gpt-5.5 (accessed on 19 May 2026).
44. Ray, P.P. A Survey on Model Context Protocol: Architecture, State-of-the-Art, Challenges and Future Directions. *arXiv*. 2025. Available online: https://www.techrxiv.org/doi/full/10.36227/techrxiv.174495492.22752319/v1 (accessed on 2 June 2026).
45. Li, H.; Xu, Y.; Hong, T. EnergyPlus-MCP: A Model-Context-Protocol Server for Ai-Driven Building Energy Modeling. *SoftwareX* **2025**, *32*, 102367. [CrossRef]
46. Gullí, A. Model Context Protocol. In *Agentic Design Patterns*; Springer: Cham, Switzerland, 2025; pp. 147–162. [CrossRef]
47. Hou, X.; Zhao, Y.; Wang, S.; Wang, H. Model Context Protocol (MCP): Landscape, Security Threats, and Future Research Directions. *ACM Trans. Softw. Eng. Methodol.* **2026**. [CrossRef]
48. Yin, J.; Yu, D.; Yin, Z.; Liu, M.; He, Q. Evaluating the Impact and Risk of Pluvial Flash Flood on Intra-Urban Road Network: A Case Study in the City Center of Shanghai, China. *J. Hydrol.* **2016**, *537*, 138–145. [CrossRef]
49. Zhang, Z.; Valeo, C. Verification of PCSWMM's LID Processes and Their Scalability over Time and Space. *Front. Water* **2022**, *4*, 1058883. [CrossRef]
50. Derayeh, K.; Finlay, S.; Fox, S.; MacKinnon, S.; Matheson, B.; Millar, G. *Handbook of Steel Drainage & Highway Construction Products*, 2nd ed.; Corrugated Steel Pipe Institute: Cambridge, ON, Canada, 2007.
51. Zarlenga, A.; Fiori, A. A Continuous-Simulation Approach for the Design and Long-Term Performance Assessment of Infiltration Basins for Sustainable Urban Water Management. *Sustainability* **2026**, *18*, 4488. [CrossRef]
52. Zarlenga, A.; Fiori, A. SWMM Models for Urban Drainage Network and Infiltration Ponds Simulation. Available online: https://zenodo.org/records/19606749 (accessed on 1 June 2026).
53. TUFLOW TUFLOW SWMM Tutorial—Module 03 Complete Model. Available online: https://wiki.tuflow.com/TUFLOW_SWMM_Tutorial_M03 (accessed on 19 May 2026).
54. Schilling, J.; Tränckner, J. Generate_SWMM_inp: An Open-Source QGIS Plugin to Import and Export Model Input Files for SWMM. *Water* **2022**, *14*, 2262. [CrossRef]
55. Atil, B.; Aykent, S.; Chittams, A.; Fu, L.; Passonneau, R.J.; Radcliffe, E.; Rajagopal, G.R.; Sloan, A.; Tudrej, T.; Ture, F.; et al. LLM Stability: A Detailed Analysis with Some Surprises. *arXiv* **2025**, arXiv:2408.04667.
56. Patil, S.G.; Zhang, T.; Wang, X.; Gonzalez, J.E. Gorilla: Large Language Model Connected with Massive APIs. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 9–15 December 2024; Volume 37.
57. Ouyang, S.; Zhang, J.M.; Harman, M.; Wang, M. An Empirical Study of the Non-Determinism of ChatGPT in Code Generation. *ACM Trans. Softw. Eng. Methodol.* **2025**, *34*, 42. [CrossRef]
58. Eythorsson, D.; Clark, M. Toward Automated Scientific Discovery in Hydrology: The Opportunities and Dangers of AI Augmented Research Frameworks. *Hydrol. Process.* **2025**, *39*, e70065. [CrossRef]
59. Richards, C.E.; Tzachor, A.; Avin, S.; Fenner, R. Rewards, Risks and Responsible Deployment of Artificial Intelligence in Water Systems. *Nat. Water* **2023**, *1*, 422–432. [CrossRef]

60. Karpatne, A.; Deshwal, A.; Jia, X.; Ding, W.; Steinbach, M.; Zhang, A.; Kumar, V. AI-Enabled Scientific Revolution in the Age of Generative AI: Second NSF Workshop Report. *npj Artif. Intell.* **2025**, *1*, 18. [CrossRef]
61. Sela, L.; Sowby, R.B.; Salomons, E.; Housh, M. Making Waves: The Potential of Generative AI in Water Utility Operations. *Water Res.* **2025**, *272*, 122935. [CrossRef] [PubMed]
62. Eythorsson, D.; Keshavarz, K.; Thébault, C.; Ahmed, M.; Spiteri, R.; Pietroniro, A.; Clark, M. AI Enhanced Environmental Modelling Workflows: Towards Automated Scientific Exploration in Hydrology. In Proceedings of the EGU General Assembly 2025, Vienna, Austria, 27 April 2025; p. EGU25-4565.
63. Zhan, Q.; Liang, Z.; Ying, Z.; Kang, D. INJECAGENT: Benchmarking Indirect Prompt Injections in Tool-Integrated Large Language Model Agents. In Proceedings of the Annual Meeting of the Association for Computational Linguistics, Bangkok, Thailand, 16 August 2024.
64. Zverev, E.; Abdelnabi, S.; Tabesh, S.; Fritz, M.; Lampert, C.H. Can Llms Separate Instructions From Data? And What Do We Even Mean By That? In Proceedings of the 13th International Conference on Learning Representations, ICLR 2025, Singapore, 23–27 April 2025.
65. Sampson, A.K. *The Hydrologic Modeler's Evolving Role in the Age of AI*; Copernicus Publications: Vienna, Austria, 2026.
66. Bengio, Y.; Hinton, G.; Yao, A.; Song, D.; Abbeel, P.; Darrell, T.; Harari, Y.N.; Zhang, Y.Q.; Xue, L.; Shalev-Shwartz, S.; et al. Managing Extreme AI Risks amid Rapid Progress: Preparation Requires Technical Research and Development, as Well as Adaptive, Proactive Governance. *Science* **2024**, *384*, 842–845. [CrossRef]
67. Ruan, Y.; Dong, H.; Wang, A.; Pitis, S.; Zhou, Y.; Ba, J.; Dubois, Y.; Maddison, C.J.; Hashimoto, T. Identifying The Risks Of Lm Agents With An Lm-Emulated Sandbox. In Proceedings of the 12th International Conference on Learning Representations, ICLR 2024, Vienna, Austria, 7–11 May 2024.
68. Oreskes, N.; Shrader-Frechette, K.; Belitz, K. Verification, Validation, and Confirmation of Numerical Models in the Earth Sciences. *Science* **1994**, *263*, 641–646. [CrossRef]
69. Huang, L.; Yu, W.; Ma, W.; Zhong, W.; Feng, Z.; Wang, H.; Chen, Q.; Peng, W.; Feng, X.; Qin, B.; et al. A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions. *ACM Trans. Inf. Syst.* **2025**, *43*, 42. [CrossRef]
70. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention Is All You Need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Volume 2017-December, pp. 6000–6010.
71. Obsidian-Sharpen Your Thinking. Available online: https://obsidian.md/ (accessed on 20 May 2026).
72. Zhang, Z.; Valeo, C. Quantifying Uncertainty in Flowrate Modelling Using Spatially Defined Fuzzy Entropy Based on Hydrological Processes in a Catchment. *J. Hydrol.* **2026**, *664*, 134447. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.