netidee

PROJEKTE

LEO Trek

Endbericht | Call 19 | Projekt ID 7442

Content

1 Introduction

All the objectives planned for the project have been successfully achieved. Table 1 provides an overview of the key project output and the goals. The target number of academic publications for excellent success has already been reached. Overall, work has been carried out according to plan.

Table 1: Project KPIs Overview

KPI	Current State	Goal
Academic publications	8	Minimum: 3 Ideally: 5
Open-source software projects	6	3
Blog posts	8	8

2 Project Description

LEO Trek explores how modern distributed applications can be executed efficiently across the emerging Edge–Cloud–Space 3D Continuum. This continuum extends today's Edge–Cloud computing model by adding Low Earth Orbit (LEO) satellites as an additional execution layer. LEO constellations introduce new opportunities for low-latency connectivity and in-orbit processing, but also add strong dynamics: nodes move, links appear and disappear, available bandwidth fluctuates, and resource constraints are more pronounced than in conventional cloud environments. These characteristics make it difficult to apply existing orchestration, scheduling, and cost optimization techniques, which typically assume stable networks and static infrastructure.

The main objective of the project is to provide open, reusable building blocks that enable the design, evaluation, and optimization of serverless and AI-enabled workflows in this highly dynamic environment. In particular, LEO Trek aims to (i) make the 3D Continuum experimentally accessible through scalable simulation, (ii) develop scheduling and optimization mechanisms that respect platform, workload, and network Service Level Objectives (SLOs), and (iii) provide a platform that enables state management and hardware acceleration that are required by real applications deployed across heterogeneous layers.

The primary target groups of LEO Trek are researchers and developers working on distributed systems, edge-cloud-native platforms, and serverless computing, as well as teams working on Earth observation analytics. The outputs are particularly relevant for users who need to evaluate orchestration strategies under mobility and network uncertainty, such as academic groups, platform engineers, and applied R&D teams. Since all results are released as open-source software, the project also explicitly targets third parties who want to reuse and extend the artifacts for their own research prototypes or industrial proofs of concept.

The project delivers a set of complementary results that collectively span the lifecycle, from experimentation to real-world execution. First, Stardust provides a scalable and extensible simulation environment for the 3D Continuum. It allows realistic evaluation of routing and orchestration algorithms under dynamic topologies, supports large mega-constellations, and offers a plugin mechanism to integrate custom logic, making it possible to explore ideas that would be difficult or costly to test in real satellite systems. Second, the project introduces scheduling and optimization mechanisms for serverless workflows. HyperDrive is an SLO-aware scheduler for the 3D Continuum that jointly considers resource capacity, workload requirements, and network conditions. By explicitly accounting for satellite mobility and operational constraints it enables placement decisions that remain effective despite changing connectivity. Third, workflow-level resource optimization is addressed by ChunkFunc. ChunkFunc assigns resource profiles to individual functions based on their input size, improving the cost–performance trade-off of serverless workflows. This enables more reliable SLO compliance while reducing unnecessary overprovisioning. Fourth, the project tackles state management in distributed serverless workflows through Databelt. Databelt continuously places and propagates state along the expected execution path, anticipating movement in the 3D Continuum. Fusing state access for co-located functions significantly reduces remote storage operations and workflow latency.

Finally, Gaia provides a practical approach to hardware acceleration for serverless AI. Gaia automates CPU and GPU selection, moving it from manual developer decisions to the platform by combining static code analysis with runtime telemetry. This ensures that performance SLOs are met while avoiding the inefficient use of accelerators. Overall, LEO Trek delivers an open-source toolkit for understanding and optimizing serverless execution in the Edge–Cloud–Space continuum. The results support reproducible experimentation, enable new orchestration and optimization techniques, and provide reusable system components for future research and practical deployments.

3 Status of Work Packages

The work packages are proceeding according to plan. Table 2 lists the academic publications made over the first half of the project, and Table 3 lists the open-source software projects.

Table 2: LEO Trek Academic Publications

No.	AP	Publication
1	AP2	T. Pusztai, J. Hisberger, C. Marcelino, and S. Nastic, “Stardust: A Scalable and Extensible Simulator for the 3D Continuum,” in 2025 IEEE International Conference on Edge Computing and Communications (EDGE), 2025.
2	AP3	T. Pusztai, C. Marcelino, and S. Nastic, “HyperDrive: Scheduling Serverless Functions in the Edge-Cloud-Space 3D Continuum,” in 2024 IEEE/ACM Symposium on Edge Computing (SEC), 2024.

3	AP3	T. Pusztai and S. Nastic, "ChunkFunc: Dynamic SLO-aware Configuration of Serverless Functions," IEEE Transactions on Parallel and Distributed Systems, 2025.
4	AP4	M. Helcig and S. Nastic, "FedCCL: Federated Clustered Continual Learning Framework for Privacy-focused Energy Forecasting," in The 9th IEEE International Conference on Fog and Edge Computing (ICFEC), 2025.
5	AP4	C. Marcelino, S. Gollhofer-Berger, T. Pusztai, and S. Nastic, "Cosmos: A Cost Model for Serverless Workflows in the 3D Compute Continuum," in 2025 IEEE International Conference on Smart Computing (SMARTCOMP), 2025.
6	AP4	C. Marcelino, L. Guelmino, T. Pusztai, and S. Nastic, "Databelt: A Continuous Data Path for Serverless Workflows in the 3D Compute Continuum". *Journal of Systems Architecture*, 2025.
7	AP4	C. Marcelino, T. Pusztai, and S. Nastic, "Roadrunner: Accelerating Data Delivery to WebAssembly-based Serverless Functions". In *Proceedings of the 24th International Middleware Conference (Middleware 25)*, 2025.
8	AP4	M. Reisecker, C. Marcelino, T. Pusztai, and S. Nastic, "Gaia: Hybrid Hardware Acceleration for Serverless AI in the 3D Compute Continuum". In *Proceedings of the IEEE/ACM 12th International Conference on Big Data Computing, Applications and Technologies (BDCAT 25)*, 2025.

Table 3: LEO Trek Published Open-source Software

No.	AP	Software	URL
1	AP2	Stardust 3D Continuum Simulator	https://github.com/polaris-slo-cloud/stardust
2	AP3	HyperDrive Serverless Scheduler	https://github.com/polaris-slo-cloud/hyper-drive
3	AP3	ChunkFunc Serverless Workflow Optimizer	https://github.com/polaris-slo-cloud/chunk-func
4	AP4	FedCCL Federated Learning Framework	https://github.com/polaris-slo-cloud/fedccl
5	AP4	Databelt Serverless Function State Management	https://github.com/polaris-slo-cloud/databelt
6	AP4	Gaia Hybrid Serverless Runtime for AI Workloads	https://github.com/polaris-slo-cloud/gaia

3.1 AP 1 – *Project Management*

This work package has been carried out and concluded according to plan. The team has shown high ambition and produced substantial results, as detailed in the subsequent AP results. There were no noteworthy issues or delays. The project costs adhered to the plan.

There has been a change to the team: Thomas Pusztai's contract at the TU Wien expired on August 31, 2025, and thus, he had to leave the project team. Since he had to use up his remaining vacation time, his last day of work was July 30. His duties have been taken over by Cynthia Marcelino and Stefan Nastic.

3.2 AP 2 – *Stardust Simulator*

This work package has been carried out and concluded according to plan. It resulted in one academic publication and one open-source software artifact.

3.2.1 Stardust: A Scalable and Extensible Simulator for the 3D Continuum

Low Earth Orbit (LEO) mega constellations provide low-latency communication between LEO and terrestrial nodes and among terrestrial nodes, extending the Edge-Cloud Continuum into an Edge-Cloud-Space 3D Continuum. Developing orchestration services and applications for the 3D Continuum, such as RapidREC, requires realistic simulations of the highly dynamic network conditions and node locations inherent to this environment. Unfortunately, existing simulators only allow for relatively small constellations to be simulated without scaling to a large number of host machines. Stardust is a scalable and extensible open-source simulator for the 3D Continuum. Our main contributions are:

- Stardust, a scalable and extensible next-generation simulator for the 3D Continuum with support for simulating LEO-, Cloud-, and Edge nodes in a scalable manner. Stardust enables experiments for evaluating networking and orchestration algorithms for the 3D Continuum. It supports simulating mega constellations three times the size of the currently largest constellation, with almost 7k satellites on a single machine.
- A dynamic routing mechanism that enables experimentation with different routing mechanisms by making the ISL routing protocol and the network path computation changeable. This allows, e.g., changing the default +Grid ISL routing to a different protocol or to introduce caching or hypergraph algorithms as a replacement for Dijkstra's algorithm to calculate node-to-node network paths.
- SimPlugin, a plugin mechanism that serves as the integration point for custom logic that Stardust should execute at every step of the simulation. A SimPlugin has access to the complete infrastructure state and, thus, allows integrating, e.g., orchestration algorithms/software that should be evaluated using Stardust.

3.3 AP 3 - *LEO Trek Scheduler*

This work package has been carried out and concluded according to plan. It resulted in the publication of two academic papers and two software artifacts.

3.3.1 HyperDrive – Serverless Workflow Scheduler for the 3D Continuum

HyperDrive is a platform and network Service Level Objective-aware scheduler for serverless functions in the 3D continuum. The 3D continuum expands the Edge-Cloud continuum to include low earth orbit (LEO) satellites. These satellites have enormously grown in number in the recent years and are projected to provide valuable compute resources, especially for Earth Observation (EO) data from satellites by avoiding unnecessary downlinking of the massive amounts of data. Satellite EO data can be used to survey the region around an accident to assess the state of the road network, predict congestion, and devise a plan for faster recovery. The contributions of HyperDrive include:

- A novel Serverless Platform that introduces novel components and mechanisms tailored to the unique characteristics of the 3D Continuum. HyperDrive enables functions to be seamlessly executed anywhere in the 3D Continuum, optimizing performance and reliability.

- A Serverless scheduler for the 3D Continuum that considers constraints such as resource capacity, application SLO requirements, and network load to minimize the end-to-end Serverless workflow latency. The HyperDrive scheduler, also considers satellite position and thermal conditions to enable function scheduling in the 3D Continuum. By considering edge, cloud, and space conditions, HyperDrive executes functions that meet every SLO requirement in the 3D Computing Continuum. HyperDrive achieves 71% lower end-to-end (E2E) network latency than the next best baseline approach.

The HyperDrive scheduler is designed to address the challenges that arise in the placement of serverless functions in the 3D Continuum using an optimization problem (see academic paper for this) and using a Multi Criteria Decision Making (MCDM) approach. The MCDM approach considers the vicinity of candidate nodes to the node of the previous function and source data, the resources of the candidate node, the network SLOs, and the maximum allowed operating temperature of the candidate node, if it is a satellite.

3.3.2 ChunkFunc – SLO- and Input-aware Resource Optimizer for Serverless Workflows

ChunkFunc is a resource optimizer for serverless workflows. It assigns resource profiles to a serverless workflow's functions to ensure that the response time Service Level Objective (SLO) of the workflow is met, while minimizing costs. Unlike much of the state-of-the-art, ChunkFunc considers the size of the input data of a function when assigning resources. This ensures SLO compliance when the input is larger than average and saves costs when the input is smaller than average. This approach benefits applications with highly diverse input sizes, such as traffic analysis systems. During rush hour, the input to a periodically executed accident detection workflow is larger than average and during night, the input is smaller than average. ChunkFunc's contributions include:

- An SLO- and input data size-aware function performance model for determining optimized configurations in serverless workflows, depending on the input data size.
- ChunkFunc Profiler, which automatically builds performance models for serverless functions and workflows based on typical input data sizes. Profiling is automatic, users only deploy a function and specify typical input data. A novel, auto-tuned Bayesian Optimization approach reduces the profiling costs by up to 90% compared to exhaustive profiling and ensures high accuracy of the results.
- ChunkFunc Workflow Optimizer, which leverages various heuristics to dynamically adapt the resource configuration of functions in a workflow to meet a performance-based SLO (e.g., response time), while minimizing cost. Depending on the workflow it increases SLO adherence by a factor of 1.12 to 2.0 and reduces costs by up to 53% The Workflow Optimizer is extensible with arbitrary performance-based SLOs.

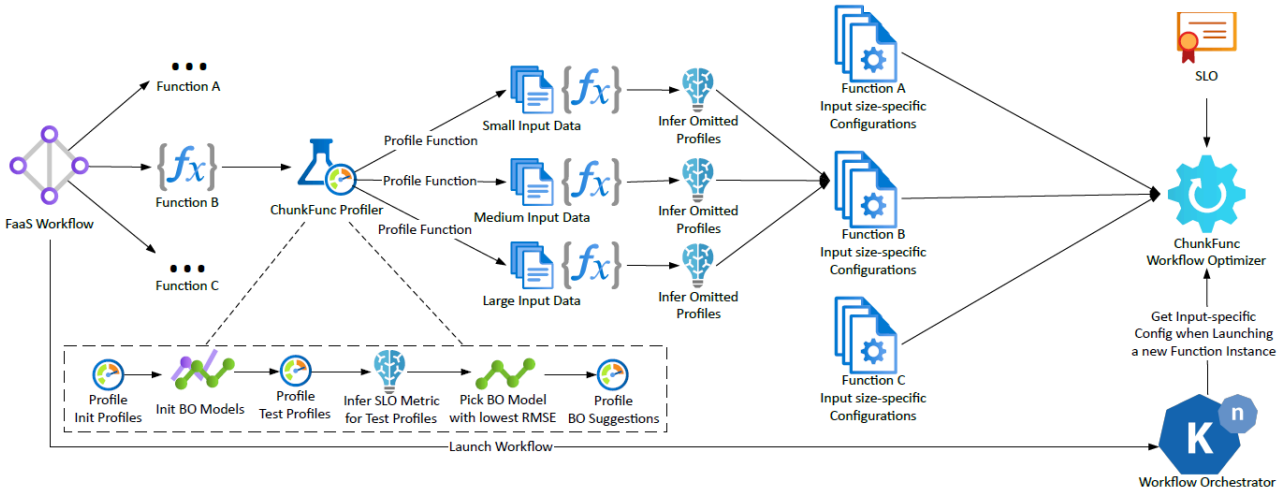

Figure 1: Overview of the ChunkFunc System and Lifecycle of a Serverless Workflow.

The ChunkFunc framework consists of two major components: The Profiler and the Workflow Optimizer. Figure 1 presents an overview of ChunkFunc and the lifecycle of a serverless workflow within the system. Upon their deployment, serverless functions are automatically picked up by the ChunkFunc Profiler. It deploys function instances using various resource configurations to execute profiling runs with their typical input data sizes, without any user interaction. To reduce the number of profiling runs, while maintaining a high accuracy of the results, the choice of resource configurations is guided by Bayesian Optimization. Our BO Dynamic Hyperparameter Selection picks the hyperparameter that yields the most accurate results for a particular function type and input size combination. Finally, the input-specific performance profiles are leveraged by the ChunkFunc Workflow Optimizer, which provides a suitable resource profile, to meet the workflow's SLO and minimize cost, to the serverless orchestrator prior to invoking a function.

3.4 AP 4 - *LEO Trek Platform*

This work package has been carried out and concluded according to plan. It resulted in the publication of five academic papers and two software artifacts.

3.4.1 FedCCL: Federated Clustered Continual Learning Framework for Privacy-focused Energy

Privacy-preserving distributed model training is crucial for modern machine learning applications, yet existing Federated Learning approaches struggle with heterogeneous data distributions and varying computational capabilities. Traditional solutions either treat all participants uniformly or require costly dynamic clustering during training, leading to reduced efficiency and delayed model specialization. FedCCL (Federated Clustered Continual Learning) is a framework that addresses these challenges through a combination of pre-training clustering and asynchronous Federated Learning. Unlike most of the existing approaches that perform clustering during or after training [1] [2], FedCCL employs DBSCAN clustering

based on static characteristics before training begins. This approach enables immediate model specialization while reducing coordination overhead. Furthermore, participants can belong to multiple

clusters simultaneously, facilitating more nuanced knowledge sharing than strict partitioning approaches [3]. FedCCL's main contributions include:

- FedCCL Framework: A Federated Learning framework that integrates clustered pre-training with an enhanced asynchronous FedAvg algorithm. The framework operates through a two-phase approach, initially clustering clients based on their inherent system properties before training, followed by client-driven updates with model locking during aggregation. Mitigating the performance degradation typically seen in asynchronous Federated Learning with heterogeneous data while maintaining reduced overhead.
- FedCCL Predict & Evolve: Through our system property-based clustering approach, FedCCL creates a framework that provides a specialized model for newly joining clients without requiring prior exposure to their specific data distributions. In the Predict phase, new clients can immediately benefit from these highly specialized models to generate predictions. As clients begin contributing their own data, they enter the Evolve phase, where they participate in training and refining cluster-specific models. Our evaluation demonstrates this capability through robust generalization metrics, where models achieve nearly identical performance levels for both training and independent populations, with mean error rates showing minimal degradation of only 0.14 percentage points for new installations.

3.4.2 Cosmos: A Cost Model for Serverless Workflows in the 3D Compute Continuum

To allow evaluating costs for serverless deployments in the 3D (Edge-Cloud-Space) Continuum efficiently, Cosmos constitutes a novel cost- and performance-cost-tradeoff model for serverless workflows that identifies key factors that affect cost changes across different workloads and cloud providers. Common approaches for serverless cost estimation include: (a) Predictions [4, 5, 6] use models, such as ML and math models to estimate costs based on historical execution data. This enables the estimation and analysis of costs without executing or even deploying a workflow. However, these high-level predictions often fail to provide detailed cost breakdowns or to identify the main drivers of higher expenses. (b) Simulations [7, 8, 9] enable users to explore how costs behave under different parameter configurations. They offer valuable insights into performance and expenses across various workload patterns, highlighting important trade-offs. However, existing simulation tools often lack fine-grained parameters to identify which aspects contribute to higher costs.

Since current cost models are not detailed enough for precise performance-cost tradeoff decisions, users often err on the side of caution and incur higher costs to ensure performance. The Cosmos cost model enables the building of intelligent frameworks to optimize serverless costs and maximize performance. Our main contributions include:

- Cosmos: A cost and a performance-cost tradeoff model for serverless workflows that incorporates the heterogeneity and dynamic characteristics of the 3D Continuum. Cosmos isolates the main

cost drivers while accounting for their interdependencies, providing an understanding of how different

factors impact execution and cost, e.g., resource constraints, workload characteristics, communication overhead, and dynamic pricing.

- A cost taxonomy that classifies the main cost drivers, enabling their identification among invocation, compute, data transfer, state management, and BaaS. This provides insights into specific cost drivers for serverless workflows across the different layers of the 3D Continuum.

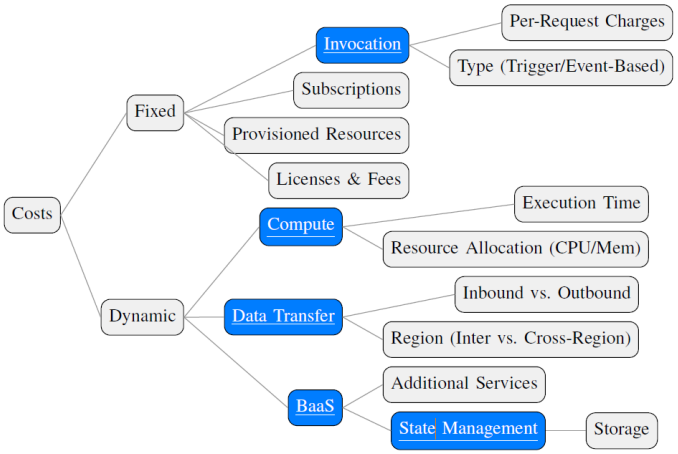

Figure 2: Serverless workflow costs drivers, highlighting key cost drivers: Invocation, Compute, Data Transfer, and State Management (partial view).

3.4.3 Databelt: A Continuous Data Path for Serverless Workflows in the 3D Continuum

LEO constellations extend the Edge–Cloud Continuum into an Edge–Cloud–Space 3D Continuum, where nodes move, links appear and vanish, and latency/bandwidth fluctuate constantly. Traditional serverless workflows assume stable networks and centralized storage, which quickly become a bottleneck when functions span satellites, the edge, and the cloud. Databelt is our state-aware serverless model and system that continuously places, propagates, and fuses function state along the workflow's path, anticipating orbital movement and meeting application SLOs. Our key contributions are:

- Databelt: A novel state-aware serverless model and architecture that enables state placement in the dynamic and heterogeneous environments of the 3D Continuum. Databelt allows serverless functions to move the data in orbit closer to the target function, and place functions within a workflow on nearby nodes, reducing workflow latency while adhering to specific environmental conditions of the 3D Compute Continuum;
- A function state propagation mechanism that leverages node position, including edge, cloud, and satellite, to identify the neighbor nodes and propagate the state to specific nodes within the execution range while complying with SLO requirements.
- A function state fusion mechanism to avoid multiple state retrievals for functions that share the same serverless runtime, thus minimizing storage operations to a constant request amount instead of a linear increase and consequently reducing latency.

Fig 3 shows Databelt's execution model that proceeds in three lightweight phases. When a workflow stage starts, Databelt first identifies the live topology (i.e., nodes, links, and their latency/bandwidth windows),

excluding nodes that are about to leave the communication range. It then computes a shortest propagation path, reverses it to prioritize nodes closer to the eventual destination, and selects the first candidate that satisfies a maximum migration time bound derived from data size and available bandwidth.

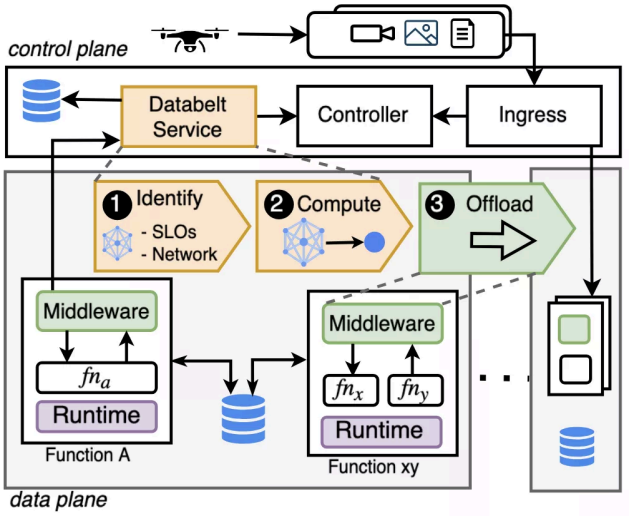

Figure 3: Databelt Architecture

Finally, it offloads state to that target; if the node is temporarily unreachable, Databelt safely falls back to the current executor and re-evaluates on the next step. In parallel, when several functions are co-located in one sandbox, a small middleware bundles their state reads and writes and serves them from local storage whenever possible. Access control remains strict via immutable, per-invocation Databelt State Keys, which allow each function to retrieve only the state it is authorized to access.

3.4.4 Gaia: Hybrid Hardware Acceleration for Serverless AI in the 3D Compute Continuum

Gaia addresses a central limitation of current serverless platforms for AI: hardware acceleration is still largely treated as a manual configuration choice instead of a first-class platform concern. As AI workloads move across heterogeneous environments in the 3D Compute Continuum, from edge devices and cloud datacenters to Low Earth Orbit satellites, they increasingly rely on GPUs to satisfy strict latency and throughput SLOs. Contemporary approaches either require developers to statically decide whether a function should run on CPU or GPU, or they perform a single device selection based on a snapshot of conditions. In dynamic settings with workload drift, fluctuating resource availability, and mobility effects such as satellite handovers, these one-time decisions break down and lead either to SLO violations or to inefficient, cost-intensive GPU usage.
Gaia introduces a GPU-as-a-service model and architecture that shifts device selection from developers to the platform and couples static analysis with continuous runtime adaptation. Conceptually, Gaia provides two complementary mechanisms. The Execution Mode Identifier performs a static inspection of the function code at deployment time. It parses the source into an abstract syntax tree, detects deep learning framework imports such as PyTorch and TensorFlow, identifies explicit GPU calls, and characterizes tensor operations to estimate computational intensity. Based on this analysis, it assigns one of four execution

modes: cpu, cpu_preferred, gpu_preferred, or gpu, and embeds this classification, along with its rationale, into the function manifest. This allows the platform to schedule functions onto CPU or GPU-capable nodes without requiring code changes or explicit hardware annotations from developers.

Gaia architecture, shown in Fig. 4, is organized into three planes: user, control, and data. In the user plane, the Static Code Analyzer classifies functions and annotates the manifest with execution hints. In the control plane, a coordinator performs dynamic runtime management, closing the loop with telemetry to enable safe and seamless mode switches. In the data plane, the function code and runtime execute, interfacing with the underlying hosting stack to utilize proper hardware acceleration, such as CPU or GPU backends, as required.

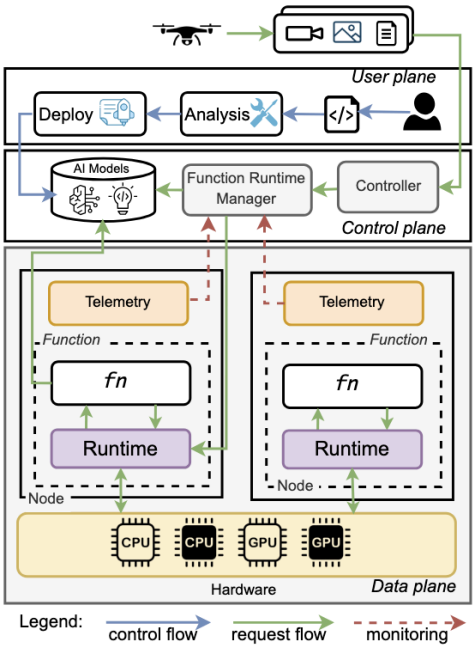

Figure 4: Gaia Architecture

3.5 AP 5 - *Documentation & Dissemination*

This work package has been carried out and concluded according to plan. Every software artifact includes documentation in its open-source repository. Additionally, each repository is accompanied by an academic publication.

4 Implementation of Funding Conditions

No special conditions were defined for this project.

5 Project End Results

1	Projektzwischenbericht	CC BY 4.0	netidee.at/leo-trek
2	Projektendbericht	CC BY 4.0	netidee.at/leo-trek
3	Paper – Stardust: A Scalable and Extensible Simulator for the 3D Continuum	CC BY 4.0	netidee.at/leo-trek/stardust-scalable-and-extensible-simulator-3d-continuum & https://arxiv.org/abs/2506.01513
4	Paper – HyperDrive: Scheduling Serverless Functions in the Edge-Cloud-Space 3D Continuum	CC BY 4.0	netidee.at/leo-trek & https://doi.org/10.1109/SEC62691.2024.00028
5	Paper – ChunkFunc: Dynamic SLO-Aware Configuration of Serverless Functions	CC BY 4.0	netidee.at/leo-trek/chunkfunc-serverless-workflow-resource-optimizer & https://ieeexplore.ieee.org/iel8/71/4359390/10959103.pdf
6	Paper – FedCCL: Federated Clustered Continual Learning Framework for Privacy-focused Energy Forecasting	CC BY 4.0	netidee.at/leo-trek & https://doi.org/10.1109/ICFEC65699.2025.00012
7	Paper – Cosmos: A Cost Model for Serverless Workflows in the 3D Compute Continuum	CC BY 4.0	netidee.at/leo-trek/cosmos-cost-model-serverless-workflows-3d-compute-continuum & https://arxiv.org/pdf/2504.20189
8	Paper – Databelt: A Continuous Data Path for Serverless Workflows in the 3D Compute Continuum	CC BY 4.0	netidee.at/leo-trek/databelt-continuous-data-path-serverless-workflows-3d-continuum & https://doi.org/10.1016/j.sysarc.2025.103577
9	Paper – Roadrunner: Accelerating Data Delivery to WebAssembly-based Serverless Functions	CC BY 4.0	netidee.at/leo-trek & https://dl.acm.org/doi/10.1145/3721462.3770777
10	Paper – Gaia: Hybrid Hardware Acceleration for Serverless AI in the 3D Compute Continuum	CC BY 4.0	netidee.at/leo-trek/gaia-hybrid-hardware-acceleration-serverless-ai-3d-compute-continuum &

			https://doi.org/10.1145/3773276.3774299
11	SW – Stardust 3D Continuum Simulator	Apache 2.0	netidee.at/leo-trek/stardust-scalable-and-extensible-simulator-3d-continuum & https://github.com/polaris-slo-cloud/stardust-go
12	SW – HyperDrive Serverless Scheduler	Apache 2.0	netidee.at/leo-trek & https://github.com/polaris-slo-cloud/hyper-drive
13	SW – ChunkFunc Serverless Workflow Optimizer	Apache 2.0	netidee.at/leo-trek/chunkfunc-serverless-workflow-resource-optimizer & https://github.com/polaris-slo-cloud/chunk-func/
14	SW – FedCCL Federated Learning Framework	Apache 2.0	netidee.at/leo-trek & https://github.com/polaris-slo-cloud/fedccl
15	SW – Databelt Serverless Function State Management	Apache 2.0	netidee.at/leo-trek/databelt-continuous-data-path-serverless-workflows-3d-continuum & https://github.com/polaris-slo-cloud/databelt/
16	SW – Gaia Hybrid Serverless Runtime for AI Workloads	Apache 2.0	netidee.at/leo-trek/gaia-hybrid-hardware-acceleration-serverless-ai-3d-compute-continuum & https://github.com/polaris-slo-cloud/gaia
17	Entwickler_innen-DOKUMENTATION	CC BY Apache 2.0	netidee.at/leo-trek/leo-trek & https://github.com/polaris-slo-cloud/stardust https://github.com/polaris-slo-cloud/hyper-drive https://github.com/polaris-slo-cloud/chunk-func https://github.com/polaris-slo-cloud/fedccl

			https://github.com/polaris-slo-cloud/databelt https://github.com/polaris-slo-cloud/gaia
18	Anwender_innen-DOKUMENTATION	CC BY Apache 2.0	netidee.at/leo-trek/leo-trek & https://github.com/polaris-slo-cloud/stardust https://github.com/polaris-slo-cloud/hyper-drive https://github.com/polaris-slo-cloud/chunk-func https://github.com/polaris-slo-cloud/fedccl https://github.com/polaris-slo-cloud/databelt https://github.com/polaris-slo-cloud/gaia
19	Dokumentation Externkommunikation zur Erreichung Sichtbarkeit/Nachhaltigkeit (als Teil des Endberichtes)	CC BY	netidee.at/leo-trek/leo-trek
20	Veröffentlichungsfähiger Einseiter / Zusammenfassung	CC BY	netidee.at/leo-trek/

6 Exploitation of the Project Results in Practice

The project results are directly applicable in research and applied development. All software artifacts are released as open source and can be used to evaluate, design, and optimize serverless workflows in dynamic Edge–Cloud–Space environments. In particular, Stardust enables realistic experimentation without access to real satellite infrastructure, while HyperDrive and ChunkFunc support improved latency, cost efficiency, and SLO compliance on heterogeneous platforms.

In practice, Databelt and Gaia address key deployment challenges related to state management and hardware acceleration in distributed serverless systems. Beyond academic use, the results already serve as a technical foundation for applied research and innovation activities, including the Starbase startup initiative, where selected components are being adapted for real-world Earth observation and distributed AI use cases.

7 Dedicated Project Website

There is no dedicated project website.

8 Dissemination and Networking

All of LEO Trek's innovations were published in academic papers. Most of them (4 out of 5) were conference papers and were presented to the audience at the respective conferences. This has sparked conversations with the attendees and plans for a Horizon Europe research project in the near future.

Furthermore, the project has resulted in a business idea for a startup company, Starbase, which has been accepted into the TU Wien i2c incubator program.

9 Planned Activities After the End of the Netidee Project

After the conclusion of the netidee project, several follow-up activities are planned to ensure sustainability and further impact of the results. All developed software artifacts will continue to be maintained as open-source projects, including bug fixes, documentation improvements, and compatibility updates with evolving serverless platforms.

The project results form the foundation for ongoing and future scientific work. The developed models, simulators, and systems will be extended and evaluated in larger-scale and longer-term experiments, including additional workload classes, hardware accelerators, and emerging Edge–Cloud–Space use case scenarios such as Serverless Compound AI. Several follow-up publications are planned, building on the existing results and addressing open research challenges identified during the project.

10 Suggestions for Further Development by Third Parties

The project results are designed to be modular, extensible, and reusable, offering multiple entry points for further development by third parties. Researchers can build upon the Stardust simulator to evaluate new routing, orchestration, or scheduling algorithms for dynamic and heterogeneous environments, or to extend the simulator with additional node types, mobility models, or network characteristics.

Developers and platform providers can reuse and extend open-source components, such as HyperDrive, ChunkFunc, Databelt, and Gaia, to integrate advanced scheduling, cost optimization, state management, or hardware acceleration mechanisms into existing serverless platforms. The provided documentation and system abstractions enable straightforward adaptation to alternative infrastructures, cloud providers, or edge environments.

Finally, practitioners and industry partners can leverage the project results as a reference architecture for designing serverless and AI-enabled systems in distributed and resource-constrained settings. Potential further developments include tighter integration with commercial cloud services, enhanced security and trust mechanisms, and domain-specific optimizations for application areas such as mobility, energy systems, smart cities, or satellite-based data analytics.

11 References

[1] F. Sattler, K.-R. Müller und W. Samek, „Clustered Federated Learning: Model-Agnostic Distributed Multitask Optimization Under Privacy Constraints," *IEEE Transactions on Neural Networks and Learning Systems,* Bd. 32, Nr. 8, p. 3710–3722, 2021.

[2] A. Ghosh, J. Chung, D. Yin und K. Ramchandran, „An Efficient Framework for Clustered Federated Learning," *IEEE Transactions on Information Theory,* Bd. 68, Nr. 12, p. 8076–8091, 2022.

[3] E. Yoo, H. Ko und S. Pack, „Fuzzy Clustered Federated Learning Algorithm for Solar Power Generation Forecasting," *IEEE Transactions on Emerging Topics in Computing,* Bd. 10, Nr. 4, p. 2092–2098, 2022.

[4] S. Eismann, J. Grohmann, E. van Eyk, N. Herbst und S. Kounev, „Predicting the Costs of Serverless Workflows," New York, NY, USA, Association for Computing Machinery, 2020, p. 265–276.

[5] R. Cordingly, W. Shu und W. J. Lloyd, „Predicting Performance and Cost of Serverless Computing Functions with SAAF," 2020, p. 640–649.

[6] C. Lin und H. Khazaei, „Modeling and Optimization of Performance and Cost of Serverless Applications," *IEEE Transactions on Parallel and Distributed Systems,* Bd. 32, Nr. 3, p. 615–632, 2021.

[7] L. C. Da Silva, R. Medeiros und N. Rosa, „COSTA: A cost-driven solution for migrating applications in multi-cloud environments," 2023, p. 57–63.

[8] F. Liu und Y. Niu, „Demystifying the Cost of Serverless Computing: Towards a Win-Win Deal," *IEEE Transactions on Parallel and Distributed Systems,* Bd. 35, Nr. 1, p. 59–72, 2024.

[9] P. Garcia Lopez, A. Slominski, B. Metzler, M. Berhendt und S. Shillaker, „Serverless End Game: Disaggregation enabling Transparency," 2024, p. 9–14.