MAST

Managing Sustainability Tradeoffs

Grant Agreement No.: 22035

D2.4 – MAST reference architecture

Document Number	D2.4
Document Title	MAST reference architecture
Version	1.1
Status	Final
Work Package	WP2
Deliverable Type	Report
Contractual Date of Delivery	M18 – June 2026
Actual Date of Delivery	M23 – November 2026 (public version)
Responsible Unit	Almende BV (task leader)
Contributors	Aalborg Universitet, Canon Production Printing Netherlands B.V., Cleanwatts Digital, S.A., FBBasic, ISEP, University of Groningen, Wirtek A/S
Keyword List	Architecture, Sustainability, Toolkit
Dissemination level	Public

Amendment History

Version	Date	Author (Unit)	Description
0.1	15.05.2026	ALM	Initial Draft with ToC
0.2	16.06.2026	ALM	Added input CLW
0.3	22.06.2026	ALM	Added input WRT, CPP
0.4	23.06.2026	ALM	Added high-level architecture + explanation and descriptions
0.5	24.06.2026	ALM	ToC revision, intro, conclusions, finalization for internal review
0.6	29.06.2026	RUG, all	Internal review
1.0	30.06.2026	ALM	Finalization of 1.0 (confidential)
1.1	15.11.2026	ALM	Upgrade to public version

\

1 Executive Summary

This deliverable presents the first version of the MAST Reference Architecture, a technology-agnostic architectural framework for sustainability-aware software-intensive and cyber-physical systems. The architecture is derived through a bottom-up analysis of the three MAST use cases from Wirtek, Cleanwatts, and Canon Production Printing, resulting in a common set of architectural building blocks, data flows, and architectural principles for sustainability management. The architecture is centered around a sustainability feedback loop consisting of data collection, interpretation, analysis, decision support, optimization, and control, and is applicable to both environmental sustainability concerns, such as energy consumption and carbon footprint, and technical sustainability concerns, such as maintainability, technical debt, and system evolution.

The proposed architecture is validated by mapping the detailed architectures of the three use cases onto the reference architecture. The resulting mappings demonstrate that the architecture is capable of capturing sustainability-management capabilities across diverse domains while remaining independent of specific technologies and implementation choices. Although the use cases primarily focus on runtime sustainability management, the architecture is equally applicable at design time, enabling existing and future sustainability-related methods, tools, and techniques throughout the system lifecycle to be positioned within a common architectural framework. The MAST Reference Architecture therefore provides both a consolidated view of sustainability-aware system management within the project and a foundation for future developments in the subsequent work packages.

2 Table of Contents

1 Executive Summary 2
2 Table of Contents 3
3 Introduction 4
3.1 Background and motivation 5
3.2 Methodology 5
3.3 Terminology and definitions 7
4 MAST sustainability concept 8
4.1 Sustainability as a Multidimensional Concept 9
4.2 Sustainability as Trade-off Management 9
4.3 Sustainability as a Managed Property 10
4.4 From Sustainability Concept to Sustainability Feedback Loop 10
5 MAST reference architecture 10
5.1 Objectives 11
5.2 Architectural principles 11
5.3 High-level architecture overview 14
5.4 Functional building blocks 15
5.5 Data flows 17
5.6 Extensibility 19
6 Example use cases 19
6.1 The WRT use case 20
6.2 The CLW use case 23
6.3 The CPP use case 26
7 Conclusion 29

3 Introduction

The MAST project addresses the challenge of managing sustainability trade-offs in software-intensive and cyber-physical systems. While technical sustainability concerns aspects such as maintainability, evolvability, and technical debt, environmental sustainability focuses on energy consumption, resource efficiency, and carbon emissions. In practice, decisions that improve one sustainability dimension may negatively affect another, requiring systematic methods to analyse, evaluate, and balance competing objectives.

Building upon the use cases and sustainability challenges identified in D2.1 Use Case Descriptions and D2.2 Sustainability State of the Art and Practice, and the requirements specified in D2.3 Requirements Specification and Mapping to Use Cases, this deliverable presents the MAST Reference Architecture. The purpose of the reference architecture is to provide a common conceptual framework for understanding, designing, and integrating sustainability-aware solutions across the diverse industrial domains represented within the project.

3.1 Background and motivation

The reference architecture captures the architectural concepts, quality attributes, design decisions, and interactions required to support sustainability assessment and trade-off management throughout the system lifecycle. It provides a high-level blueprint that interrelates the various methods, tools, and techniques developed in the project and defines the interfaces required for their integration. Particular attention is given to the relationship between technical and environmental sustainability concerns and to the mechanisms required to measure, analyse, visualise, and optimise sustainability-related trade-offs.

In addition to providing a common architectural foundation, the reference architecture serves as a basis for deriving use-case-specific architectural instances for the industrial pilots of Wirtek (WRT), Cleanwatts (CLW), and Canon Production Printing (CPP). By identifying architectural invariants that remain stable across domains and implementations, the MAST Reference Architecture facilitates reuse, interoperability, and extensibility while supporting the project's goal of establishing a novel and practical approach to sustainability-aware systems engineering.

This deliverable therefore provides both a conceptual model of sustainability-aware systems and an architectural blueprint for integrating sustainability assessment, decision support, optimisation, and reporting capabilities within a coherent ecosystem of tools and services. The resulting architecture forms the foundation for the design, implementation, and validation activities carried out in the subsequent work packages of the MAST project.

3.2 Methodology

The MAST Reference Architecture was developed using an iterative process of architecture elicitation, synthesis, and validation. Rather than designing an abstract architecture in isolation, the methodology was grounded in the requirements and architectural knowledge originating from the three industrial use cases.

The process started from the requirements defined in D2.3. Based on these requirements, an initial analysis was performed to identify the key functionalities and interactions required to support sustainability assessment and trade-off management. These functional building blocks were used as input for discussions with the industrial partners, who were subsequently asked to develop logical architectures of their envisioned MAST-enabled systems. The resulting architectures identified the major components, services, data sources, decision-support capabilities, and sustainability-related functions relevant to each domain.

The use-case-specific architectures were then analysed and compared in detail. Although the three use cases operate in different domains, they address similar sustainability challenges and exhibit recurring architectural patterns. The analysis focused on identifying common functional capabilities and data flows. Particular attention was paid to architectural elements that appeared consistently across multiple use cases, as these represent architectural invariants that are independent of a specific domain or implementation.

Based on this comparative analysis, the common architectural concepts were abstracted into a technology-independent reference architecture. This follows a widely adopted reference architecture development approach in which concrete solution architectures are generalized into a higher-level model that captures shared concerns while remaining independent of specific implementations. The resulting reference architecture defines the major functional areas, their responsibilities, and the principal data flows required to support sustainability-aware systems.

To validate the resulting architecture, the principal components of each use-case-specific architecture were mapped onto the elements of the reference architecture. This mapping verified that all relevant architectural concerns identified in the industrial use cases could be represented by the reference architecture and that no essential functionality had been lost during the abstraction process. The mappings also demonstrate the applicability of the architecture across different domains and system types.

The resulting methodology can therefore be characterized as a bottom-up, evidence-driven reference architecture development process. Starting from requirements and concrete industrial architectures, common architectural patterns were identified, synthesized into a generic architectural model, and subsequently validated through traceability and mapping back to the originating use cases.

3.3 Terminology and definitions

Term	Definition
Sustainability	The ability of a system to continue delivering value over time while balancing environmental, technical, economic, and other relevant concerns.
Sustainability Dimension	A particular aspect of sustainability that can be assessed and managed, such as environmental sustainability or technical sustainability.
Environmental Sustainability	Sustainability concerned with energy consumption, resource efficiency, carbon emissions, and environmental impact.
Technical Sustainability	Sustainability concerned with maintainability, evolvability, technical debt, and long-term system viability.
Sustainability Trade-off	A situation in which improving one objective or sustainability dimension negatively affects another objective or dimension.
Sustainability Management	The continuous process of observing, assessing, analysing, deciding upon, and improving sustainability-related properties of a system.
Sustainability Feedback Loop	The recurring cycle of observation, interpretation, analysis, decision support, optimization, intervention, and evaluation used to manage sustainability.
Managed System	The system, process, design, architecture, or organization whose sustainability properties are being assessed and/or improved.
Managing System	The system which monitors, assesses and improves the Managed Systems in terms of its sustainability dimensions.
Context Information	Information external to the managed system that influences sustainability assessments or decisions, such as carbon-intensity data, energy prices, regulations, weather conditions, or organizational policies.
Observation	Raw data collected from a managed system or its environment.
Interpretation	The transformation of observations into meaningful sustainability-related information, metrics, indicators, or models.
Forecasting	The process of estimating future values of sustainability-related metrics, indicators, or system states based on historical observations, current measurements, predictive models, or external information.
Sustainability Assessment	The evaluation of sustainability-related properties by stakeholers (e.g. software engineers, architects, technical leads) based on interpreted information.
Sustainability Analysis	The investigation of sustainability information to identify patterns, trends, risks, opportunities, or trade-offs.
Decision Support	Human-oriented capabilities that present sustainability information, analyses, forecasts, and recommendations to stakeholders.
Optimization	Automated reasoning capabilities that determine preferred actions or configurations based on defined objectives and constraints.
Intervention	A change applied to a managed system with the intention of influencing sustainability outcomes.
Control	The mechanism through which interventions are executed on the managed system.
Sustainability Metric	A quantitative measure used to evaluate a sustainability-related property, such as energy consumption, carbon emissions, maintainability score, or technical debt level.
Sustainability Indicator	A derived measure or aggregation of metrics used to support assessment and decision-making.
Stakeholder	A person, role, or organization that consumes sustainability information or participates in sustainability-related decision-making.

Reference Architecture	A technology-independent architectural model that identifies common functional building blocks, responsibilities, interfaces, and data flows for a class of systems.

4 MAST sustainability concept

The analyses performed in D2.1, D2.2, and D2.3 revealed that sustainability in software-intensive and cyber-physical systems is a multidimensional concern that cannot be reduced to a single metric or objective. While environmental sustainability aspects such as energy consumption, carbon emissions, and resource efficiency are increasingly important, technical sustainability aspects such as maintainability, evolvability and system longevity are equally relevant. Furthermore, decisions intended to improve one sustainability aspect may adversely affect another, making sustainability management a complex decision-making problem.

From the use cases, requirements, and architecture synthesis activities performed in WP2, three recurring themes emerged: sustainability is multidimensional, sustainability requires balancing trade-offs between competing objectives, and sustainability should be managed continuously throughout the system lifecycle. Together, these themes form the MAST sustainability concept and provide the conceptual foundation for the MAST Reference Architecture.

4.1 Sustainability as a Multidimensional Concept

MAST recognizes that sustainability encompasses multiple dimensions. In particular, the project focuses on environmental sustainability and technical sustainability. Environmental sustainability concerns the efficient use of energy and resources and the reduction of environmental impact. Technical sustainability concerns the long-term ability of systems to evolve, remain maintainable, and continue delivering value over time.

An important observation is that the mechanisms required to assess and improve sustainability are largely independent of the specific sustainability dimension under consideration. Whether the objective is to reduce energy consumption, lower carbon emissions, improve maintainability, reduce technical debt, or increase system longevity, similar activities are required: observations must be collected, interpreted, analysed, and translated into decisions and interventions. This common structure enables a single architectural framework to support multiple sustainability dimensions.

4.2 Sustainability as Trade-off Management

The use cases and requirements analysed in WP2 consistently demonstrated that sustainability decisions rarely involve a single objective. Instead, stakeholders must balance environmental sustainability, technical sustainability, performance, cost, reliability, productivity, usability, and other quality attributes. Improvements in one area may introduce negative effects in another.

MAST therefore primarily addresses trade-off management between sustainability dimensions . Rather than optimizing a single sustainability metric, stakeholders should be supported in

understanding the consequences of alternative decisions and selecting solutions that best satisfy their objectives and constraints. Sustainability management therefore requires mechanisms for comparing alternatives, evaluating impacts, and making trade-offs explicit and understandable.

4.3 Sustainability as a Managed Property

The multidimensional nature of sustainability and the need to continuously balance competing objectives imply that sustainability cannot be treated as a one-time assessment activity. Instead, sustainability must be managed throughout the system lifecycle as systems, environments, requirements, and usage patterns evolve over time.

MAST therefore treats sustainability as a system property that can be continuously observed, assessed, analysed, and improved. Sustainability management is not viewed as a periodic reporting activity, but as an ongoing process that supports both operational and strategic decision-making. This perspective applies equally to environmental sustainability concerns, such as energy consumption and carbon emissions, and to technical sustainability concerns, such as maintainability, evolvability, and technical debt.

4.4 From Sustainability Concept to Sustainability Feedback Loop

The three themes above naturally lead to a feedback-oriented view of sustainability management. If sustainability consists of multiple dimensions, if these dimensions interact through trade-offs, and if sustainability must be managed continuously, then sustainability management cannot be a one-time activity. Instead, it becomes an iterative process in which observations are collected from a system and its context, interpreted into sustainability-relevant information, analysed to identify opportunities and trade-offs, and used to support decisions and interventions. The effects of these interventions can subsequently be observed and evaluated, thereby closing the loop.

This sustainability feedback loop provides the conceptual foundation for the MAST Reference Architecture presented in the next section. The architecture operationalizes this concept by defining the functional building blocks and data flows required to support sustainability assessment, trade-off analysis, decision support, optimization, and sustainability-driven interventions.

5 MAST reference architecture

In this section, we present the MAST Reference Architecture. Derived from the analysis of the three industrial use cases WRT, CLW and CPP, it identifies the common functional capabilities, architectural building blocks, and data flows required to assess, monitor, and analyse sustainability aspects. The architecture provides a common blueprint for integrating the methods, tools, and techniques developed within the MAST project while remaining applicable across different domains and system types.

5.1 Objectives

The primary goal of the MAST Reference Architecture is to provide a common architectural framework for sustainability-aware systems that integrates sustainability concerns throughout the system lifecycle. The architecture aims to support the assessment, monitoring, analysis, visualization, and optimization of sustainability-related properties, while enabling informed trade-off decisions between competing quality attributes such as maintainability, performance, energy consumption, and carbon emissions. In doing so, it establishes a common conceptual foundation for the methods, tools, and techniques developed within the MAST project.

A second goal is to provide a reusable and extensible blueprint that can be applied across different industrial domains. The architecture must therefore capture the architectural invariants shared by the MAST use cases while remaining independent of specific technologies and implementations. Furthermore, it must support the integration of sustainability assessment tools, decision-support mechanisms, optimization services, and reporting capabilities through clearly defined interfaces and information flows. Finally, the architecture should facilitate traceability from sustainability-related observations and measurements to analyses, decisions, and interventions, thereby supporting continuous feedback loops and evidence-based sustainability management.

5.2 Architectural principles

The MAST Reference Architecture was derived from three use cases operating in very different domains: internet of things, energy community management, and industrial printing. Despite these differences, the analysis revealed a number of recurring architectural structures, assumptions, and design principles that remained stable across all use cases. These architectural invariants define the essence of the reference architecture and explain why it can be applied to different systems and sustainability concerns.

An important observation is that the architecture is not specific to environmental sustainability. Although energy consumption, carbon emissions, and resource efficiency are prominent concerns within MAST, the same architectural structure can be applied to technical sustainability aspects such

as maintainability, technical debt, evolvability and system lifetime. The architecture should therefore be viewed as a generic sustainability assessment and trade-off management architecture rather than as a domain-specific solution.
The reference architecture adheres to the following architectural principles.

1. **Sustainability feedback and decision loop**

The architecture shall support a closed feedback loop in which a system is observed, interpreted, analysed, and influenced through decisions or control actions. A fundamental characteristic of this loop is the separation between observation, interpretation, analysis, decision-making, and intervention. The architecture should reflect this principle by explicitly representing these activities as distinct concerns and by showing how they collectively form a closed feedback structure.

2. **Context-awareness through external information**

Sustainability assessments and decisions are influenced by factors outside the managed system itself. External information such as carbon-intensity data, energy prices, weather forecasts, market signals, regulations, or organizational policies may contribute to the observation, interpretation, forecasting, and analysis of sustainability-related information. The architecture should therefore allow external information to enter the sustainability management loop at multiple stages on the observation and reasoning side of the architecture, enabling sustainability assessments and decisions to be made within their broader operational context.

3. **Both automated and human-in-the-loop**

The architecture shall support both automated and human-in-the-loop sustainability management. Depending on the use case, decisions may be taken automatically, by human stakeholders, or through a combination of both. The architecture should distinguish between internal sustainability management capabilities and stakeholder-facing capabilities. Stakeholder-facing capabilities should form the highest level of abstraction within the architecture, providing access to sustainability information, analyses, recommendations, and decisions without exposing the internal mechanisms used to generate them.

4. **Broad system scope across lifecycle and context**

The observed or managed system is not limited to a running technical system. It may represent a design, architecture, software system, cyber-physical system, operational process, or organizational workflow. Furthermore, sustainability outcomes depend not only on the system itself but also on its environment, operational context, and usage patterns. The architecture should therefore represent the managed system in a sufficiently broad and technology-independent manner.

5. **Sustainability-dimension independence**

The architecture is independent of the specific sustainability dimension being studied. The same building blocks can be used to assess environmental sustainability, technical sustainability, or other sustainability concerns. While the metrics, models, and data sources may differ, the underlying architectural structure remains unchanged. The architecture should therefore avoid embedding sustainability-dimension-specific assumptions into its core building blocks.

6. **Recursive applicability**

The architecture may be applied to itself. Sustainability assessment, monitoring, optimization, and decision-support components can themselves become subjects of sustainability analysis. This recursive property is particularly relevant for AI models, telemetry infrastructures, and sustainability assessment services whose own sustainability characteristics may need to be evaluated. The architecture should therefore be defined in terms of generic responsibilities rather than domain-specific artefacts.

7. **Explicit trade-off management**

The architecture assumes that sustainability decisions involve balancing multiple, often conflicting objectives. Trade-offs between maintainability, energy consumption, carbon emissions, performance, cost, reliability, accuracy, and other quality attributes are therefore treated as first-class concerns. The architecture should therefore leave room for trade-off analysis and decision support, without constraining implementations to single-objective optimization or isolated sustainability metrics.

8. **Multi-timescale operation**

The architecture supports decision-making at multiple timescales. Some feedback loops operate in real time or near real time, while others support longer-term activities such as reporting, design evaluation, upgrade planning, lifecycle management, and strategic decision-making. The architecture should therefore employ terminology and abstractions that remain applicable across operational, tactical, and strategic timescales, without restricting implementations to a particular decision horizon.

9. **Flexible realization with traceable information flows**

Concrete architecture instances do not need to implement every building block explicitly. Capabilities may be omitted, merged, specialized, or replaced according to the needs of a particular domain. Regardless of the realization, sustainability information should remain traceable to its underlying data sources, assumptions, models, and calculations. The architecture should therefore define logical information flows and responsibilities rather than prescribing a fixed implementation structure.

10. **Technology independence and interface orientation**

The reference architecture does not prescribe specific technologies, platforms, algorithms, or implementation approaches. Instead, it defines functional responsibilities and logical interfaces

between building blocks, enabling different implementations while maintaining interoperability and reuse. The architecture should therefore focus on capabilities and interactions rather than concrete tools or products.

5.3 High-level architecture overview

In this section, we present the high-level MAST Reference Architecture, derived from the comparative analysis of the three industrial use cases. The architecture captures the common functional building blocks and data flows that were identified across the domains of internet of things, energy management, and industrial printing. Rather than prescribing a specific implementation, it provides a technology-independent view of how sustainability-related observations, analyses, decisions, and interventions can be organized within a coherent architectural framework. Figure 1 presents the resulting high-level reference architecture.

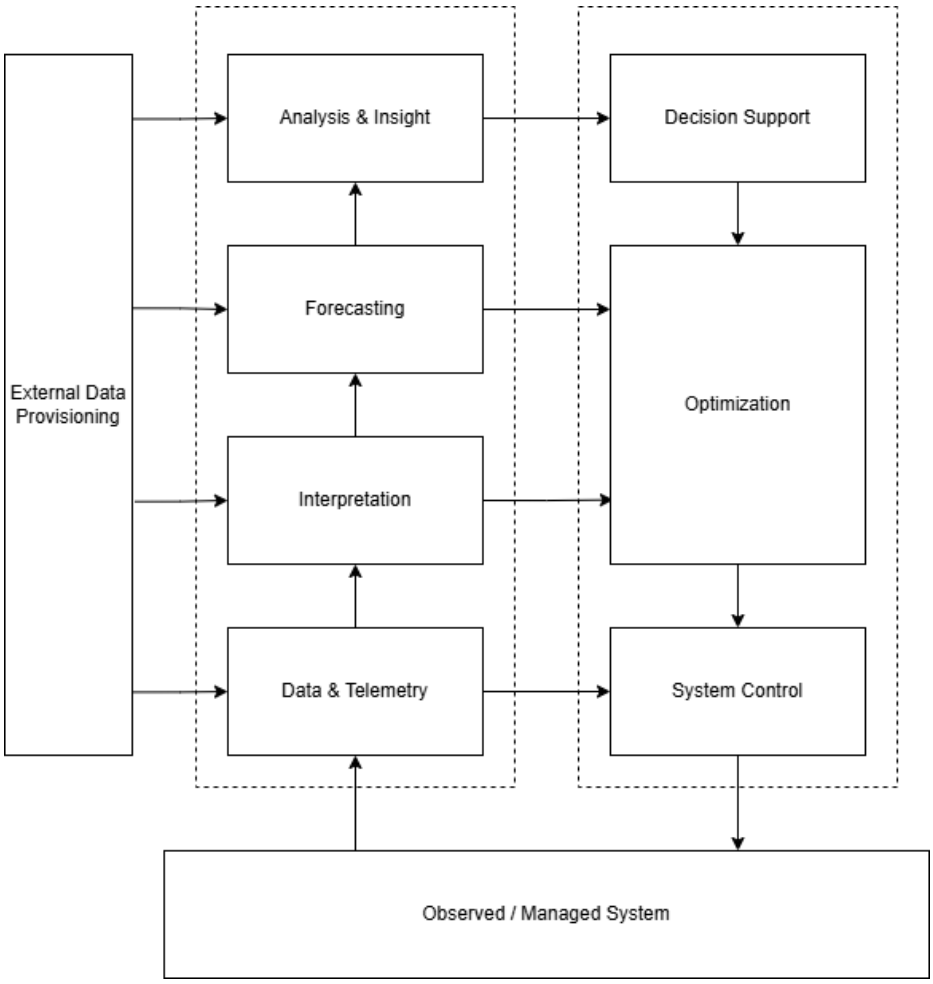


Figure 1: MAST Reference Architecture

The architecture is designed to realize the architectural principles identified in the previous section. It explicitly supports a closed sustainability feedback loop, distinguishes between observation, interpretation, analysis, decision-making, and intervention, enables the integration of contextual information from external sources, and accommodates both automated and human-in-the-loop sustainability management. Furthermore, the architecture remains independent of specific sustainability dimensions, technologies, and implementation choices, allowing it to be instantiated across different domains, lifecycle stages, and timescales.

5.4 Functional building blocks

The MAST Reference Architecture consists of a number of functional building blocks that together support the observation, assessment, analysis, optimization, and management of technical and environmental sustainability. The building blocks are technology-independent and represent architectural capabilities rather than concrete implementations. Below, we provide a table with an explanation for each of them.

Observed / Managed System	The Observed / Managed System represents the system whose sustainability characteristics are being assessed and improved. Depending on the use case, this may include software systems, cyber-physical systems, industrial equipment, energy assets, or organizational processes. The building block should be interpreted broadly and may encompass not only the technical system itself, but also relevant aspects of its operational context and usage, such as workloads, jobs, user behaviour, or environmental conditions. The managed system is the source of observations and the target of sustainability-related interventions.
System Control	The System Control building block represents the mechanisms through which sustainability-related decisions are translated into actions. Depending on the use case, these actions may consist of runtime control actions, configuration changes, workload scheduling, deployment decisions, process adaptations, engineering recommendations, or other interventions that influence the observed / managed system.
Data & Telemetry	The Data & Telemetry building block collects, aggregates, stores, and manages information about the observed / managed system and/or its operational context. Typical data sources include telemetry streams, sensor measurements, operational logs, usage data, workload information, system state information, and historical records. This building block provides the factual basis for sustainability assessment, analysis, and decision-making.

Interpretation	The Interpretation building block transforms observations and contextual information into sustainability-relevant information. This includes the calculation of sustainability indicators, derivation of metrics, application of sustainability models, maintainability assessments, estimation of environmental impact, and interpretation of system behaviour in terms of sustainability objectives. Its primary purpose is to bridge the gap between raw observations and sustainability-aware reasoning.
Forecasting	The Forecasting building block predicts future system behaviour, environmental conditions, resource demand, sustainability metrics, or operational outcomes. Forecasts provide a forward-looking perspective that supports proactive decision-making and enables optimization based on anticipated future states rather than only current observations.
Analysis & Insight	The Analysis & Insight building block evaluates sustainability-related information and identifies patterns, risks, opportunities, and trade-offs. It supports the assessment of alternative solutions, the evaluation of sustainability impacts, and the comparison of competing objectives. A central responsibility of this building block is to transform sustainability-related information into human-comprehensible insights that support understanding, discussion, and decision-making by stakeholders.
Decision Support	The Decision Support building block provides stakeholder-facing capabilities that expose sustainability information, analyses, forecasts, recommendations, and alternative courses of action. Typical services include dashboards, reports, alerts, scenario comparisons, trade-off visualizations, and configuration facilities through which stakeholders can specify preferences, objectives, constraints, or alternative implementation choices. Its primary purpose is to support human decision-making by enabling stakeholders to understand sustainability implications, evaluate alternatives, influence sustainability-related decisions, and make informed choices.
Optimization	The Optimization building block evaluates alternative actions or configurations and determines preferred solutions according to one or more objectives and constraints. Typical objectives include reducing energy consumption, minimizing carbon emissions, improving maintainability, reducing waste, increasing operational efficiency, or balancing competing sustainability concerns. Optimization may support both runtime adaptation and design-time decision-making.

External Data Provisioning	Many sustainability decisions depend on information originating outside the managed system. This building block provides access to external data sources such as carbon-intensity information, energy prices, weather forecasts, market signals, regulatory information, or contextual business data. Such information complements operational telemetry and enables context-aware sustainability analyses.

5.5 Data flows

The arrows in the reference architecture represent logical data flows between functional building blocks. They should not be interpreted as mandatory implementation-level interfaces. As discussed in Section 5.2, concrete architecture instances may omit, merge, specialize, or replace individual building blocks. In such cases, the corresponding flow can conceptually pass through the omitted building block and connect to the next relevant capability. For example, stakeholders may derive interventions directly from decision-support information without involving an explicit optimization capability. The flows described below should therefore be interpreted as logical relationships between architectural responsibilities rather than as a prescribed implementation structure.

Observed / Managed System → Data & Telemetry	The managed system provides observations about its operation, behaviour, usage, and state. These observations may consist of sensor data, logs, execution traces, energy measurements, job data, platform metrics, or other operational telemetry.
Data & Telemetry → Interpretation	Collected data is transformed into sustainability-relevant information. This flow supports the interpretation of raw measurements in terms of energy use, carbon impact, maintainability, operational efficiency, waste, or other sustainability indicators.
Data & Telemetry → System Control	Some control actions may depend directly on real-time telemetry without passing through higher-level analysis or optimization. This supports operational feedback, monitoring-based control, and fast corrective actions.
Interpretation → Forecasting	Interpreted sustainability and system-state information can be used as input for forecasting. This allows predictions to be based not only on raw data, but also on derived indicators and contextualized system states.
Interpretation → Optimization	Optimization requires a structured representation of the sustainability characteristics of the observed system. This flow provides optimization services with interpreted information about constraints, impacts, objectives, trade-offs, and system behaviour. Depending on the use case, this information may

	relate to runtime operation, system design, or alternative implementation choices.
Forecasting → Analysis & Insight	Forecasts are analysed to identify expected risks, opportunities, trends, and trade-offs. This enables proactive sustainability assessment rather than purely retrospective evaluation.
Forecasting → Optimization	Forecasts provide optimization services with information about expected future conditions. This supports decisions such as scheduling, workload shifting, operational planning, or selection between alternative execution strategies.
Analysis & Insight → Decision Support	Analysis results are transformed into stakeholder-oriented information that can be consumed through decision-support mechanisms such as dashboards, reports, alerts, recommendations, scenario comparisons, configuration facilities, and trade-off visualizations.
Decision Support → Optimization	Stakeholder decisions, preferences, constraints, objectives, and configuration choices can guide optimization activities. This flow represents the translation of human intent into parameters, policies, priorities, or optimization criteria that influence sustainability-related decision-making.
Optimization → System Control	Optimization results are translated into sustainability-related interventions that influence the observed or managed system. Depending on the use case, these interventions may take the form of control actions, configuration changes, deployment decisions, engineering recommendations, workload adaptations, schedules, or operational parameters.
System Control → Observed / Managed System	Sustainability-related interventions are applied to the observed or managed system, thereby closing the feedback loop. Depending on the use case, interventions may be automated, human-mediated, or implemented through organizational, engineering, or operational processes.
External Data Provisioning → Data & Telemetry	External data sources may enrich the telemetry layer with contextual information such as carbon intensity, energy prices, weather data, market information, or regulatory context. This allows operational measurements to be interpreted in relation to external conditions.
External Data Provisioning → Interpretation	External data can also be used directly in the interpretation of sustainability impact. For example, energy consumption can be translated into carbon emissions using carbon-intensity data, or operational behaviour can be assessed against externally defined thresholds or constraints.
External Data Provisioning → Forecasting	Forecasting services may require external input such as weather forecasts, market signals, energy prices, grid conditions, or

	carbon-intensity forecasts. This flow enables predictions that account for future environmental or operational context.
External Data Provisioning → Analysis & Insight	External data can support higher-level analysis, benchmarking, and trade-off assessment. It provides the contextual basis for comparing alternatives and interpreting sustainability outcomes beyond the boundaries of the managed system itself.

5.6 Extensibility

The MAST Reference Architecture is intentionally defined in terms of functional responsibilities and logical data flows rather than specific technologies, tools, or implementation approaches. This makes the architecture extensible and allows it to evolve as new sustainability concerns, assessment methods, optimization techniques, and decision-support capabilities emerge. New functionality can be incorporated by extending or specializing existing building blocks without requiring fundamental changes to the overall architectural structure. For example, in the CPP case (see Section 6.3), we might extend the functionality of forecasting by distinguishing between system-related forecasting (e.g., wear and tear of components) and behaviour-related forecasting (e.g., future incoming job trends).

Furthermore, the architecture does not require all building blocks to be instantiated in every implementation (architectural principle 9). Organizations may adopt only those capabilities that are relevant to their objectives, domain, and level of maturity. Additional capabilities can subsequently be introduced as requirements evolve, allowing implementations to grow incrementally while remaining aligned with the reference architecture.

The architecture is also intended to operate within a broader ecosystem of tools and systems. Existing monitoring platforms, telemetry infrastructures, engineering tools, digital twins, optimization engines, reporting solutions, and business applications can be integrated by mapping their functionality onto the architectural building blocks and data flows. Rather than replacing existing solutions, the MAST Reference Architecture provides a common framework for understanding, integrating, and extending sustainability-related capabilities across heterogeneous environments.

6 Example use cases

In this section, we demonstrate the applicability of the MAST Reference Architecture by relating it to the three industrial use cases that contributed to its development. For each use case, we first present the original architecture proposed by the industrial partner, then discuss its relationship to the architectural principles identified in Section 5.2 and finally show how the architecture can be mapped onto the MAST Reference Architecture. Together, these mappings provide evidence that the reference architecture is sufficiently general to capture the common concerns for the management of sustainability and related trade-offs of all three use cases while remaining applicable across different domains and implementation contexts.

6.1 The WRT use case

The Wirtek architecture extends the Wappsto platform with sustainability assessment and optimisation capabilities. The architecture introduces mechanisms for runtime instrumentation, telemetry collection, sustainability metric calculation, and sustainability-aware orchestration. A central architectural concept is the coexistence of alternative implementations of equivalent functionality, enabling trade-offs between technical and environmental sustainability to be analysed and acted upon. Sustainability information is continuously collected, analysed, and fed back into the platform, creating a closed sustainability management loop.

6.1.1 Detailed Architecture and Architectural Interpretation

Figure 2 presents the detailed architecture developed by Wirtek for integrating sustainability-aware operation into the Wappsto platform. The architecture introduces capabilities for measuring, analysing, and optimizing sustainability characteristics of the platform itself. For reasons of clarity, the figure distinguishes between an “observed system” part and an “observing system” part. In practice, however, both are really the same system, resulting in a self-referential architecture in which Wappsto observes, analyses, and optimizes its own behaviour.

At the operational level, platform microservices are instrumented through an Energy Measurement Layer, while a Wapp Registry maintains information about available execution variants and deployments. A Dynamic Scheduler uses sustainability information to select execution variants and schedule workloads according to temporal and geographical sustainability criteria. Sustainability assessment is supported by an Energy & CO_2 Attribution Backbone, external carbon-intensity data sources, and stakeholder-oriented dashboards that provide insight into energy consumption, carbon emissions, and sustainability trade-offs. Together, these capabilities establish a continuous sustainability feedback loop around the platform itself.

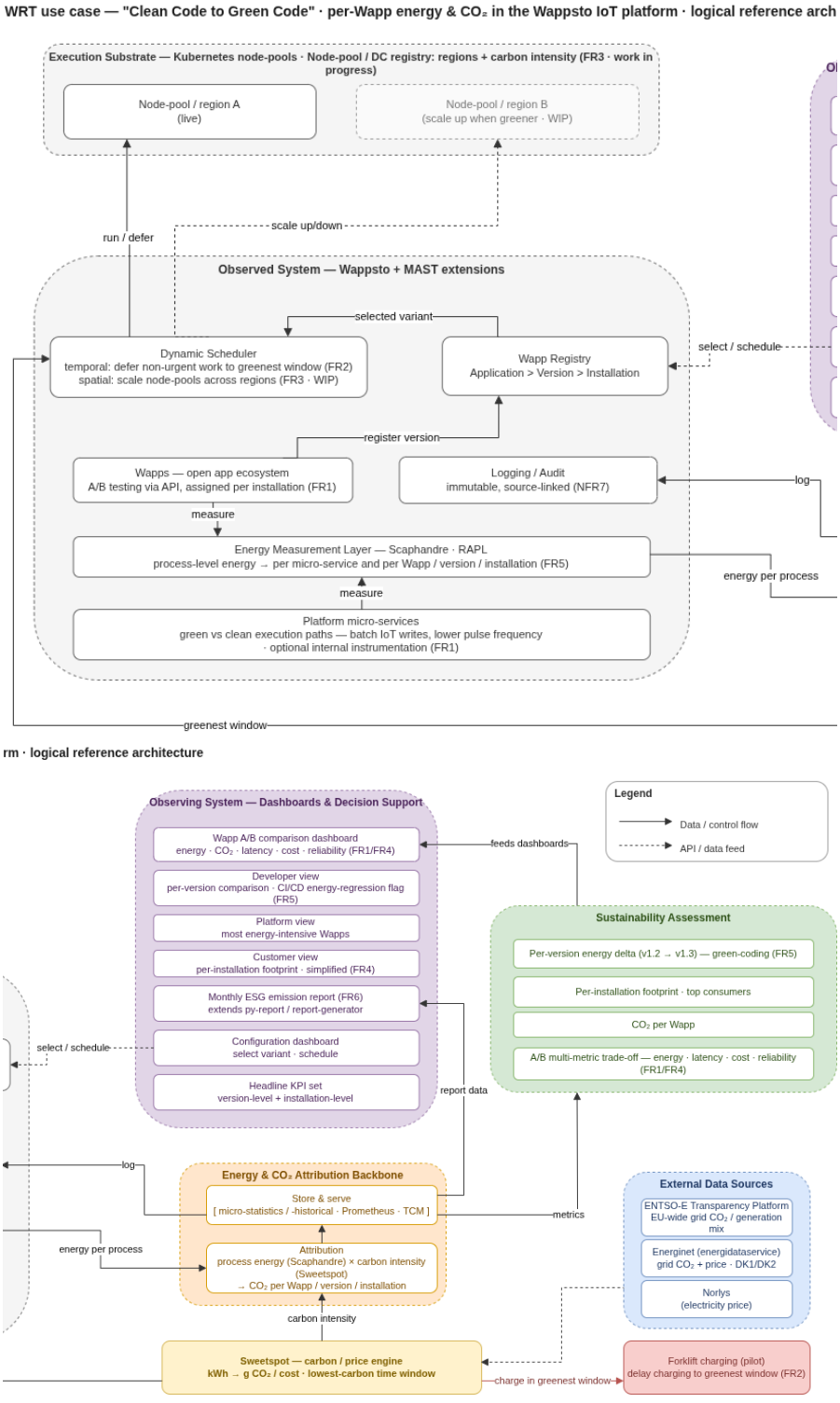


Figure 2: WRT Detailed Architecture

The WRT architecture starts from the Wappsto platform, its Wapps, microservices, and deployment infrastructure. These are the elements whose execution behaviour is measured, compared, scheduled, and improved. In terms of the MAST Reference Architecture, they therefore constitute the **Observed / Managed System**. The sustainability objective is to reduce the environmental impact of executing Wapps by selecting suitable Wapp versions, execution times, and deployment locations.

The Energy Measurement Layer collects measurements from the running platform and its execution infrastructure. Together with telemetry obtained from microservices, Wapps, and Kubernetes-based deployment infrastructure, it provides the operational data needed to reason about energy consumption and execution behaviour. These elements realize the **Data & Telemetry** building block of the reference architecture.

The Energy & CO_2 Attribution Backbone takes the collected measurements and attributes energy use and carbon impact to relevant software entities, such as Wapps, Wapp versions, installations, services, or execution paths. Because it transforms raw operational measurements into sustainability-relevant indicators, it realizes the **Interpretation** building block.

The Sustainability Assessment capabilities analyse the attributed sustainability information and compare different execution alternatives. They evaluate how Wapp versions, deployment locations, execution times, and operational choices affect energy use, carbon footprint, performance, and related trade-offs. These capabilities realize the **Analysis & Insight** building block.

The dashboards, reports, and developer-facing feedback mechanisms expose the resulting sustainability information to human stakeholders. They provide insight into current and historical carbon impact, compare implementation variants, support configuration choices, and provide auditable reporting. These elements realize the **Decision Support** building block.

The Wapp Registry and Dynamic Scheduler jointly determine which Wapp version should run, when it should run, and where it should run. The Wapp Registry provides information about available Wapp versions and their characteristics, while the Dynamic Scheduler uses this information together with carbon-intensity and cost data to select suitable execution options. These elements realize the **Optimization** building block.

The Kubernetes deployment infrastructure and related execution-control mechanisms apply the selected scheduling and deployment decisions to the running platform. By enforcing selected versions, locations, and execution times, these elements realize the **System Control** building block and close the feedback loop back to the managed Wappsto platform.

The resulting interpretation shows that the WRT architecture is a self-referential instance of the MAST Reference Architecture: the same platform that is being observed and optimized also hosts important parts of the sustainability-management machinery.

6.1.2 Mapping to the MAST Reference Architecture

Figure 3 visualizes the mapping of the main architectural elements of the WRT architecture onto the functional building blocks of the MAST Reference Architecture. The mapping confirms that all major capabilities present in the WRT architecture can be represented using the concepts of the reference architecture while preserving their original responsibilities and interactions. It further illustrates the self-referential nature of the use case, in which the Wappsto platform simultaneously acts as the managed system and hosts important parts of the sustainability-management capabilities used to observe, analyse, and optimize its own behaviour.

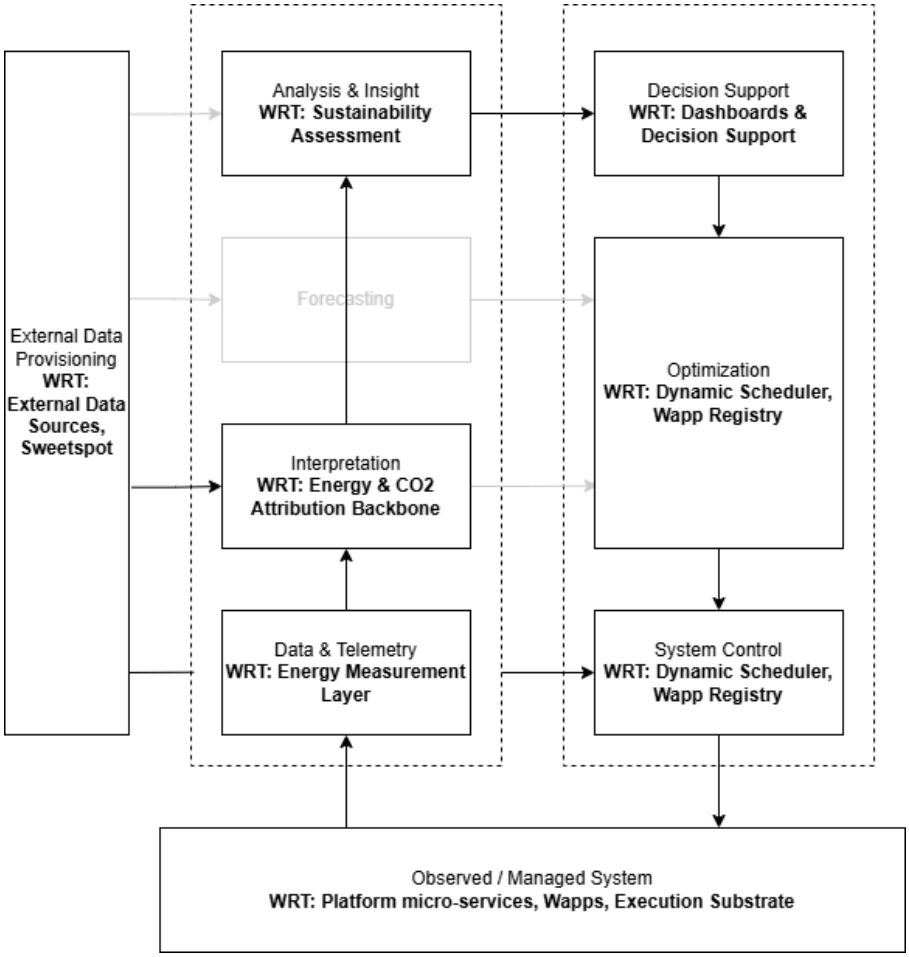


Figure 3: WRT Reference Architecture Mapping

6.2 The CLW use case

The Cleanwatts architecture combines monitoring, forecasting, optimisation, and control capabilities for renewable energy communities. The architecture integrates distributed energy resources, forecasting services, and optimisation engines within a common sustainability management framework whose primary objective is to reduce both the cost and carbon footprint of energy consumption by continuously selecting the most appropriate energy sources, storage resources, and flexibility mechanisms. In addition to optimising energy-community operation, the architecture also supports the assessment of the sustainability characteristics of the forecasting and optimisation services themselves, enabling trade-offs between prediction quality, computational cost, energy consumption, carbon footprint, and optimisation effectiveness to be analysed. As a result, the architecture exhibits a self-referential characteristic in which the sustainability-management capabilities are themselves subject to sustainability assessment.

6.2.1 Detailed Architecture and Architectural Interpretation

Figure 4 presents the detailed architecture developed by Cleanwatts for sustainability-aware management of renewable energy communities. At its core, the architecture combines AI-based forecasting, optimization, and control capabilities to continuously determine how available energy resources should be utilized in order to reduce both the cost and carbon footprint of energy consumption. External data sources, including weather forecasts and energy market information, are combined with operational measurements to support load and photovoltaic forecasting, while optimization services for collective self-consumption, flexibility participation, and scheduling determine suitable operational strategies. These strategies are subsequently enforced through a control layer that interacts with distributed energy resources, including photovoltaic systems, battery storage systems, electric vehicles, and flexible loads. The architecture further incorporates telemetry and energy-monitoring capabilities, stakeholder-facing dashboards, and a dedicated AI sustainability assessment capability that evaluates forecasting and optimization services in terms of model accuracy, runtime energy consumption, carbon emissions, and computational cost.

The CLW architecture can be interpreted in multiple ways from the perspective of the MAST Reference Architecture. In the following, we consider the distributed energy resources managed by the platform as the observed system. Under this interpretation, photovoltaic systems, battery storage systems, electric vehicles, and flexible loads constitute the system whose behaviour is monitored, analysed, and controlled in order to reduce both the cost and carbon footprint of energy consumption. An alternative interpretation is also possible, in which the AI forecasting and optimization services themselves become the observed system. Such an interpretation is relevant for assessing trade-offs between prediction quality, computational cost, energy consumption, and carbon footprint of the AI services. The following discussion focuses on the first interpretation, which provides a representative example of how the CLW architecture can be mapped onto the MAST Reference Architecture.

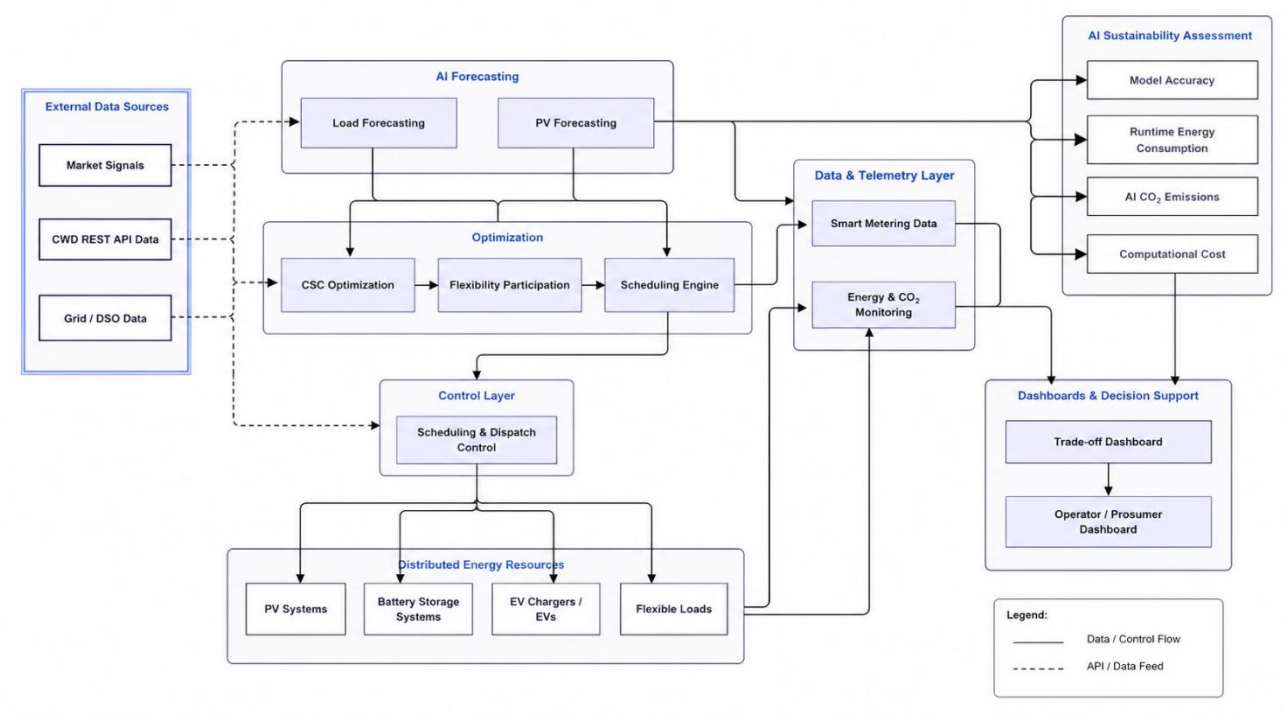


Figure 4: CLW Detailed Architecture

Starting from the detailed CLW architecture, the Data & Telemetry layer clearly fulfils the **Data & Telemetry** function. External data sources, such as weather forecasts, market information, and grid-related information, provide contextual information used by forecasting and optimization services. These elements correspond to **External Data Provisioning**.

The forecasting services, including load forecasting and photovoltaic forecasting, transform operational and external data into predictions about future demand and production. These services provide forward-looking information required for sustainability-aware operation and therefore fulfil the Forecasting function. The optimization services for collective self-consumption, flexibility participation, and scheduling determine how available resources should be used to satisfy cost, energy, and carbon-related objectives. These services therefore fulfil the **Optimization** function.

The control layer translates optimization outcomes into scheduling and dispatch actions for the distributed energy resources. It therefore fulfils the **System Control** function by applying decisions to the managed energy assets. The dashboards and decision-support services expose operational KPIs, sustainability indicators, and trade-offs to operators, prosumers, and researchers. These elements fulfil the **Analysis & Insight** and **Decision Support** functions, depending on whether they primarily analyse sustainability outcomes or present them to stakeholders for interpretation and action. Together, these elements form a sustainability feedback loop around the distributed energy resources.

6.2.2 Mapping to the MAST Reference Architecture

Figure 5 presents the mapping of the Cleanwatts use case onto the MAST Reference Architecture. In this mapping, the distributed energy resources (DERs) and their operation within renewable energy communities are considered the observed system; consequently, the AI sustainability assessment component from the detailed Cleanwatts architecture is deliberately not mapped, as this component assumes the AI parts of the system to be the observed system. The mapping shows how monitoring, forecasting, optimization, control, and decision-support capabilities support the management of energy production, storage, consumption, and flexibility across the community. It also reflects the detailed Cleanwatts architecture by showing External Data Provisioning as a direct input to Optimization.

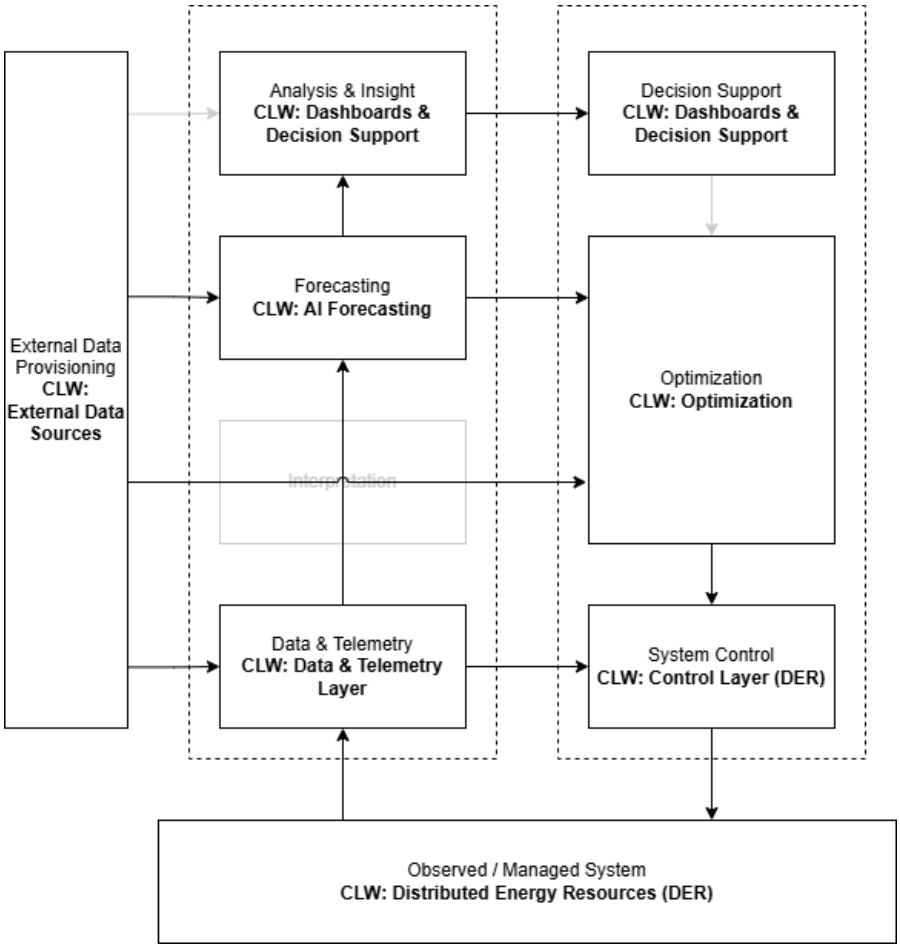


Figure 5: CLW Reference Architecture Mapping (DER perspective)

6.3 The CPP use case

The Canon Production Printing architecture addresses sustainability management in digital production printing systems. The architecture combines operational telemetry, configuration information, workload characteristics, and external sustainability data sources to assess and improve sustainability outcomes. A distinguishing characteristic is the explicit consideration of both the printing system itself and its usage context, allowing sustainability information to be analysed at the level of subsystems, system configurations, jobs, and operational practices.

6.3.1 Detailed Architecture and Architectural Interpretation

Figure 6 presents the detailed architecture developed by Canon Production Printing for sustainability management of digital printing systems throughout their lifecycle. In the context of this document, we call it “detailed architecture”, since it is detailed in comparison with the high-level reference architecture; in fact, it is an extremely abstract representation of the actual system and its surroundings, which is dramatically complex. The architecture combines systems under development, systems to be deployed, and systems operating in the field within a common sustainability-management framework. A central **Data & Telemetry** Backbone collects and correlates information originating from sensors, system states, usage data, field data, and historical system configurations. This information is processed by a Sustainability Intelligence layer containing energy models, usage-context models, and sustainability state representations, which support trade-off analysis and architectural evolution. The resulting insights are used by decision-support capabilities that address power-state strategies, rejection thresholds, scheduling decisions, and other sustainability-related interventions. By integrating design-time, deployment-time, and runtime data flows, the architecture establishes a continuous feedback loop between system operation and future system evolution.

For the CPP architecture, several interpretations are possible because the detailed architecture explicitly covers design-time, deployment-time, and runtime concerns. In the following, we consider only the runtime perspective, in which the digital printing system operating in the field is treated as the system whose sustainability characteristics are observed, analysed, and influenced. Under this interpretation, the installed printer system, its subsystems, control software, sensor infrastructure, and runtime usage context together constitute the **Observed / Managed System**.

The telemetry, logging, device sensor data, job data, system-state information, and field data collected from the running printer provide the factual basis for sustainability reasoning. These elements therefore fulfil the **Data & Telemetry** function. External information, such as carbon-intensity data or other contextual sustainability data, provides additional context for interpreting runtime behaviour and therefore fulfils the **External Data Provisioning** function.

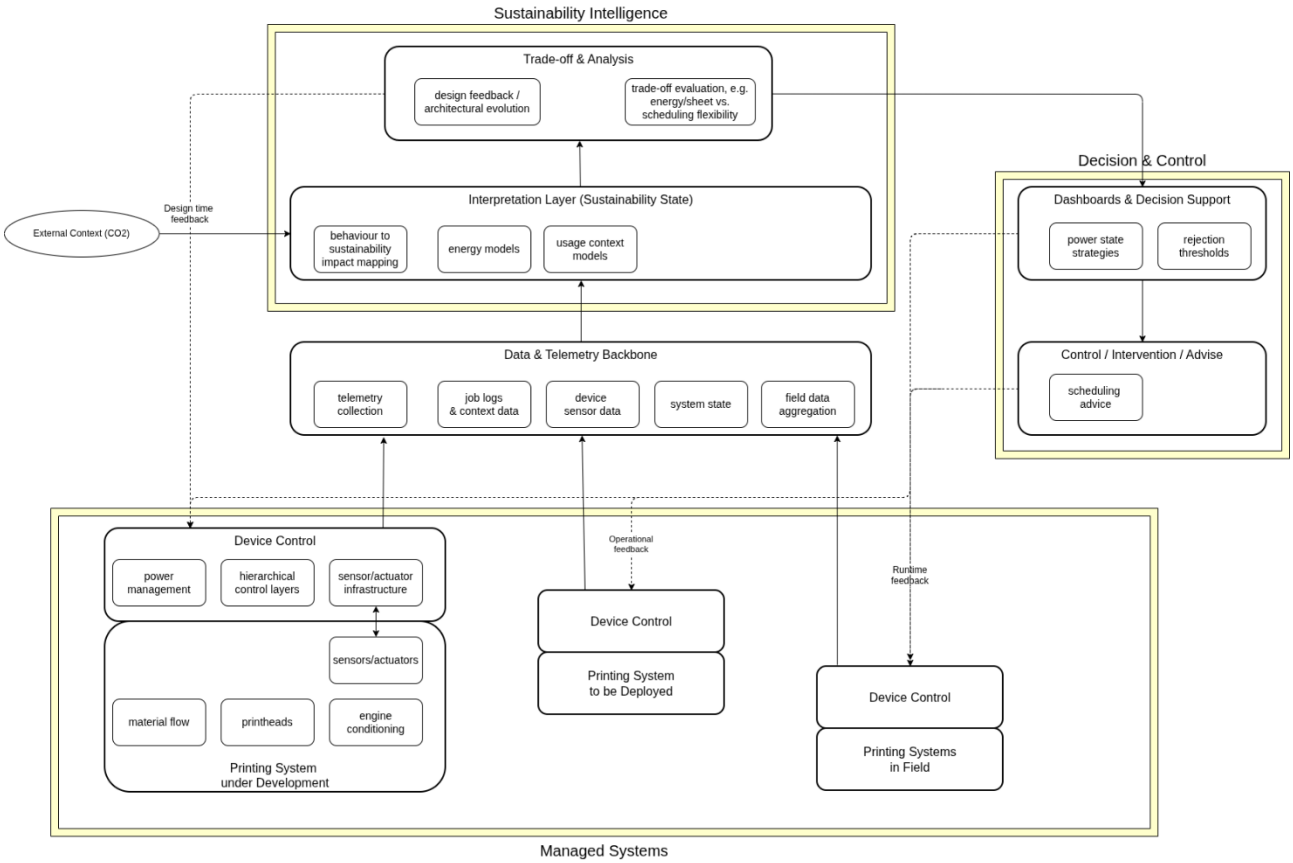


Figure 6: CPP "Detailed" (actually very high-level) Architecture

The Interpretation Layer in the detailed architecture transform runtime observations into sustainability-relevant information. Energy models, usage-context models, behaviour-to-impact mappings, and sustainability state representations relate system behaviour and usage behaviour to sustainability impact. These elements therefore fulfil the **Interpretation** function.

The Trade-off & Analysis set of components use the interpreted sustainability state to enable analysis of the sustainability performance, such as comparison of different power-management strategies. These capabilities therefore fulfil the **Analysis & Insight** function.

The dashboards and stakeholder-facing interfaces expose runtime sustainability information, diagnostics, recommendations, and trade-off information to operators, service engineers, sustainability officers, and other stakeholders. They also allow them to choose rejection thresholds, scheduling strategies, or other operational settings. These elements therefore fulfil the **Decision Support** function. Also belonging to the Decision Support function is the Design feedback, as it requires a manual intervention to decide whether certain information is relevant for design improvements.

Finally, the device-control mechanisms that apply these choices to the running printer fulfil the **System Control** function, thereby closing the runtime sustainability feedback loop. Note that we deliberately left the Forecasting and Optimization functional building blocks empty.

6.3.2 Mapping to the MAST Reference Architecture

Figure 7 presents the mapping of the Canon Production Printing use case onto the MAST Reference Architecture. In this mapping, only the runtime perspective is considered, meaning that the printing system operating in the field and its associated usage context are treated as the observed system. Design-time and deployment-time concerns, although present in the detailed architecture, are therefore deliberately excluded from the mapping. The mapping illustrates how telemetry, sustainability intelligence, trade-off analysis, decision-support, optimization, and control capabilities together form a sustainability feedback loop around the operational printing system.

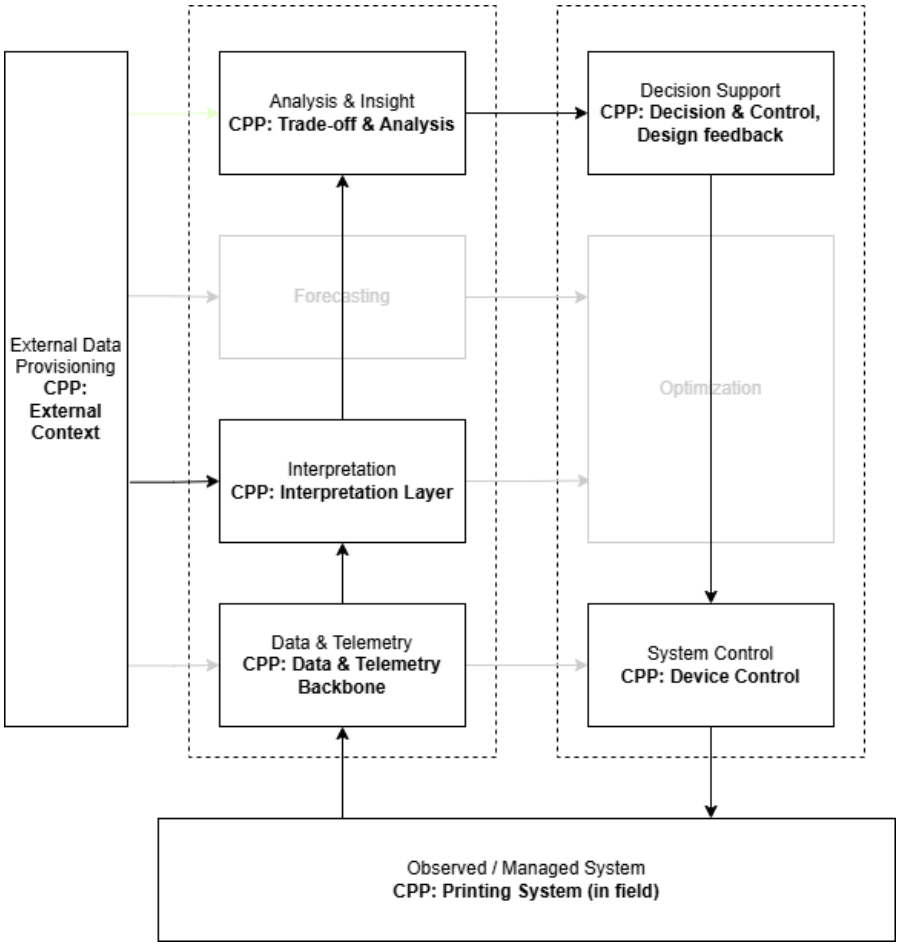


Figure 7: CPP Reference Architecture Mapping (in field perspective)

7 Conclusion

In this deliverable, we presented the first version of the MAST Reference Architecture for sustainability-aware software-intensive and cyber-physical systems. The architecture was derived through a bottom-up process starting from the three MAST use cases. After identifying common architectural structures, responsibilities, and data flows in the detailed architectures of Wirtek, Cleanwatts, and Canon Production Printing, these commonalities were consolidated into a technology-agnostic reference architecture. The resulting architecture was subsequently validated by mapping the detailed use-case architectures onto the proposed architectural building blocks.

The MAST Reference Architecture is centred around a sustainability feedback loop that combines data collection, interpretation, analysis, decision support, optimization, and control. The architecture provides a common conceptual framework for systems that aim to monitor, assess, and improve sustainability-related properties. A key characteristic of the architecture is that it remains independent of any particular sustainability dimension. While environmental sustainability, including energy consumption and carbon footprint, is an important driver in the current use cases, the same architectural structure can equally be applied to technical sustainability concerns such as maintainability, technical debt, modularity, and system evolution.

The architecture further demonstrates applicability across different domains and abstraction levels. The three use cases illustrate how the same architectural concepts can be instantiated in cloud-native software platforms, renewable energy communities, and industrial cyber-physical systems. The analysis also revealed several recurring architectural principles, including the use of closed feedback loops, the integration of external information sources, the combination of automated and human decision-making, the consideration of multiple timescales, and the possibility of treating both the managed system and its usage context as subjects of sustainability analysis. In addition, two of the use cases demonstrated self-referential architectures in which sustainability-management capabilities themselves become objects of sustainability assessment.

The mappings presented in this deliverable provide evidence that the proposed architectural building blocks are sufficiently expressive to capture the essential sustainability-management capabilities present in all three use cases. At the same time, the mappings show that the reference architecture does not prescribe a single implementation strategy. Different systems may realize the same architectural functions using different technologies, deployment models, optimization mechanisms, and stakeholder interactions while remaining consistent with the reference architecture. Although the examples presented in this deliverable primarily focus on runtime sustainability management, the same architectural concepts are equally applicable at design time. This is particularly relevant for technical sustainability concerns, such as maintainability, technical debt, modularity, and system evolution, which are often assessed before deployment rather than during operation.

As MAST progresses, the reference architecture provides a common foundation for the methods, tools, and technologies developed in subsequent work packages. It offers a shared vocabulary for discussing sustainability-management capabilities, supports the identification of reusable architectural patterns across domains, and provides a framework for positioning future project results. In addition, the architecture provides a conceptual bridge between design-time and runtime sustainability management. When applied in a design-time context, many existing tools used throughout CI/CD pipelines, quality-assurance processes, architecture analysis workflows, technical-debt management activities, and sustainability-assessment frameworks can be interpreted as realizations of particular architectural building blocks. This creates the opportunity to position a broad range of existing and future sustainability-management methods, tools, and techniques within a single coherent framework spanning the complete system lifecycle. The architecture should therefore be regarded as both a consolidation of the current understanding of sustainability-aware systems within MAST and a foundation for future refinement based on the outcomes of the project's implementation and validation activities.