ARROWHEAD

Document title
**Fundational principles**
Date
**2023-05-09**
Author
**Jerker Delsing**
Contact
**jerker.delsing@ltu.se**

Document type
**Foundational principles**
Version
**4.6.1**
Status
**DRAFT**
Page
**1 (6)**

# Fundational principles

## Eclipse Arrowhead

**Abstract**
This is the template for System Description (SysD document) according to the Eclipse Arrowehad documentation structure.

# Contents

# 1 Overview

This document describes the fundational principels for Eclispe Arrowhead.

The rest of this document is organized as follows. In Section 1.1, we reference major prior art capabilitites of the system. In Section 1.2, we the intended usage of the system. In Section **??**, we describe fundmental properties provided by the system. In Section **??**, we describe de-limitations of capabilitites ofn the system. In Section **??**, we describe the abstract service functions consumed or produced by the system. In Section **??**, we describe the security capabilitites of the system.

## 1.1 Significant Prior Art

Eclipse Arrowhead has its roots in service oriented arcitecture, SOA, and its use for primarily far edge, edge and fog automation and digitalsiation with interoperability to the cloud level.

A set of EU projects have built the foundation for whats now Eclipse Arrowhead, current version 4.6.1 with the specifications for v5.0 in the works. These projects are:

- Socrades
- IMC-AESOP
- Arrowhead
- Productive4.0
- Arrowhead Tools
- AIMS5.0
- Arrowhead fPVN

A couple of basic architectture ideas has been around since the prior art projects:

- Socrades:
  - Hard real time control using internet protocols
- IMC-AESOP:
  - Objective to be capable of implementing real world SCADA and DCS systems.
  - The local cloud concept was born. Local clouds are self contined for its intended operation enabling local security, protection and if equiped with TDMA network MAC real time properties can be achived.
  - System are self contained for its intended operation, e.g. owning and responsible for its own data storage and compitational resources.
  - Arrowhead:
    - Objective to be interoperability to legacy and internet protocols and being Open Source.
    - Basic SOA foundation established, Look-up, Late binding and Lossely coupled.
    - Mandatory core systems defined: ServiceRegistry, Orcehstration, Authorisation
    - Interoperability enabled through translation dynamically instatiated when needed.
    - v3.3 released as open source
  - Productive4.0:
    - Arrowhead Framework becomes Eclipse Arrowhead architecture and implementation platform
    - Extending the implementation platform - teh Arrowhead technology stack is defined
    - v4.5 released
  - Arowheead Tools
    - Objective to reduce engineereing cost with 20-50%
    - Extending the Eclipse Arrowehad technology stack

  - ⋆ v4.6 released
  - ⋆ Achived 30-95%engineering cost and time reduction in 28 industrial use cases along the extended IEC 81346 engineering process.

The current comprehensive high level architecture description of Eclipse Arrowhead architecture is the book “IoT Automation - Arrowhead framework” [1]. The currently released core systems and associated documentations are avialable at `www.github.com/eclipsearrowhead`.

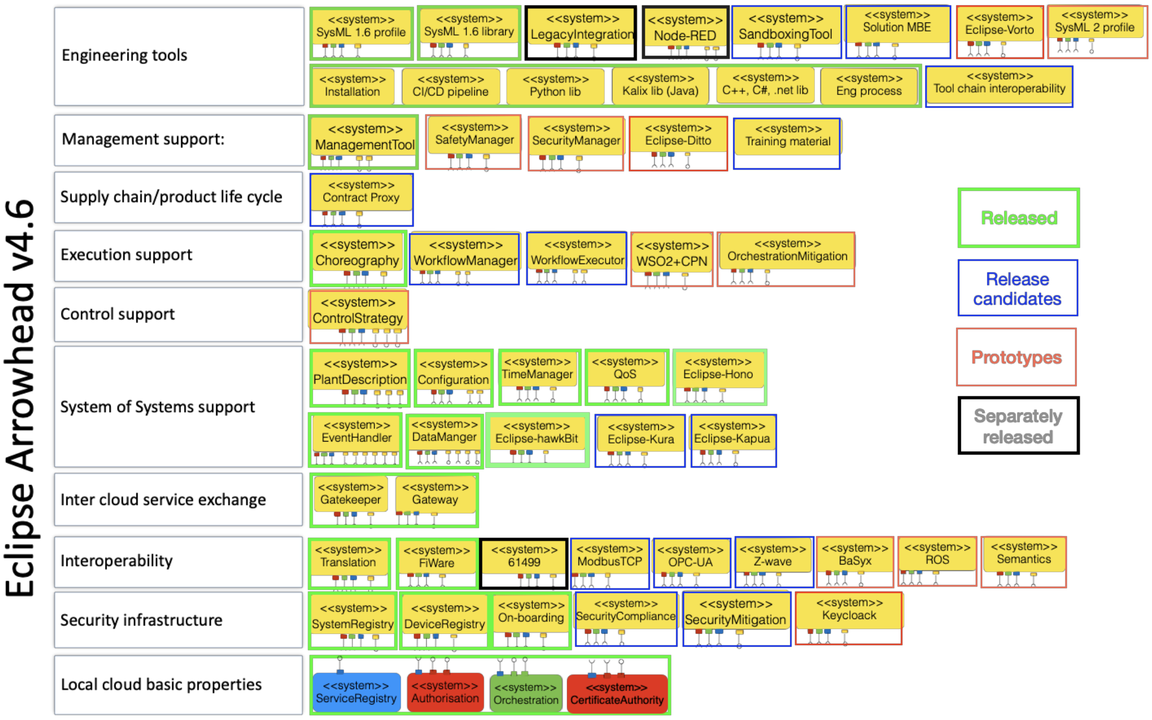


Figure 1: The Eclipse Arrowhead technology stack an dassoicated microsystems, released, relase candidates and protoypes.

## 1.2 Eclispe Arrowhead architecture philophosy

The architecture philiposy is based on the following key technology decission and objectives:

- Key technology decisions
  - A fully distributed microservice SOA approach shall be used
  - Support for design and run time engineering
  - A set of microsystems, the technology stack cf. Figure 1, shall be provided enabling the implementation of automation and digitalisation solutions
    - ⋆ Three core microsystems considered as primary and almost mandatory, ServiceRegistry, Orcehstration, Authorisation, enabling Look-up, Late binding and Lossely coupling.
    - ⋆ A set of support microsystem will be defined and implemented covering the technology stack cf. Figure enabling implementation automation architectures like ISA-95 and RAMI4.0. 1.
  - Basic microsystem properties: A micosystem can be stateless or statefull. If statefull the microsystem is responsible for its own data stoarage perferable using a database and a well established/standardised datamodel.
  - The local cloud concept shall be used providing segmentation and protection of functionall properties enabling differentieade security, safety, and real time properties within a solution architecture with managed access into a segment and between segements.

  - Network technology agnostic, allowing for different network properties inside different local clouds.

- Technology objectives:
  - Interoperability support at Service level shall be provided regrding: SOA, IP and legacy protocols, encodings, compressions, security, data models, trough translators or dedicated adaptors. For data model interopeability between major standards like e.g. ISO10303, ISO 15926, IEC 81346 are prioritised.
  - Security shall be supported at service exchange level with authentication, authorisation and audit. Security at finer granularity that service level is being addressed. Securityy is highly recommended but can be discarded if desired.
  - Secure on-boarding: On-boarding based on authentication of devices, microsystems and microservices shall be supproted.
  - Support for multipe strategy direction. A strategy directions may be: Security, LifeCycle, Maintenance, Business, Audit, Monitoring, BusinessAdminstration, BusinessModels. This will also require ways of addressing interdependecies between the various stratgy directions.

I set of terminologies used in the Eclipse Arrowhead devlopements and supporting project are important to highlight. The following are important to higlight since they may have different meaning in other domains, contexts and projects. The document .... by Emanuel Palm and Jerker Delsing has defined the most important vocabulary within the Eclipse Arrowhead development [**?**]. Some important updates are:

- Service -> Microservice
- System -> Microsystem

# 2 Implementation specifics

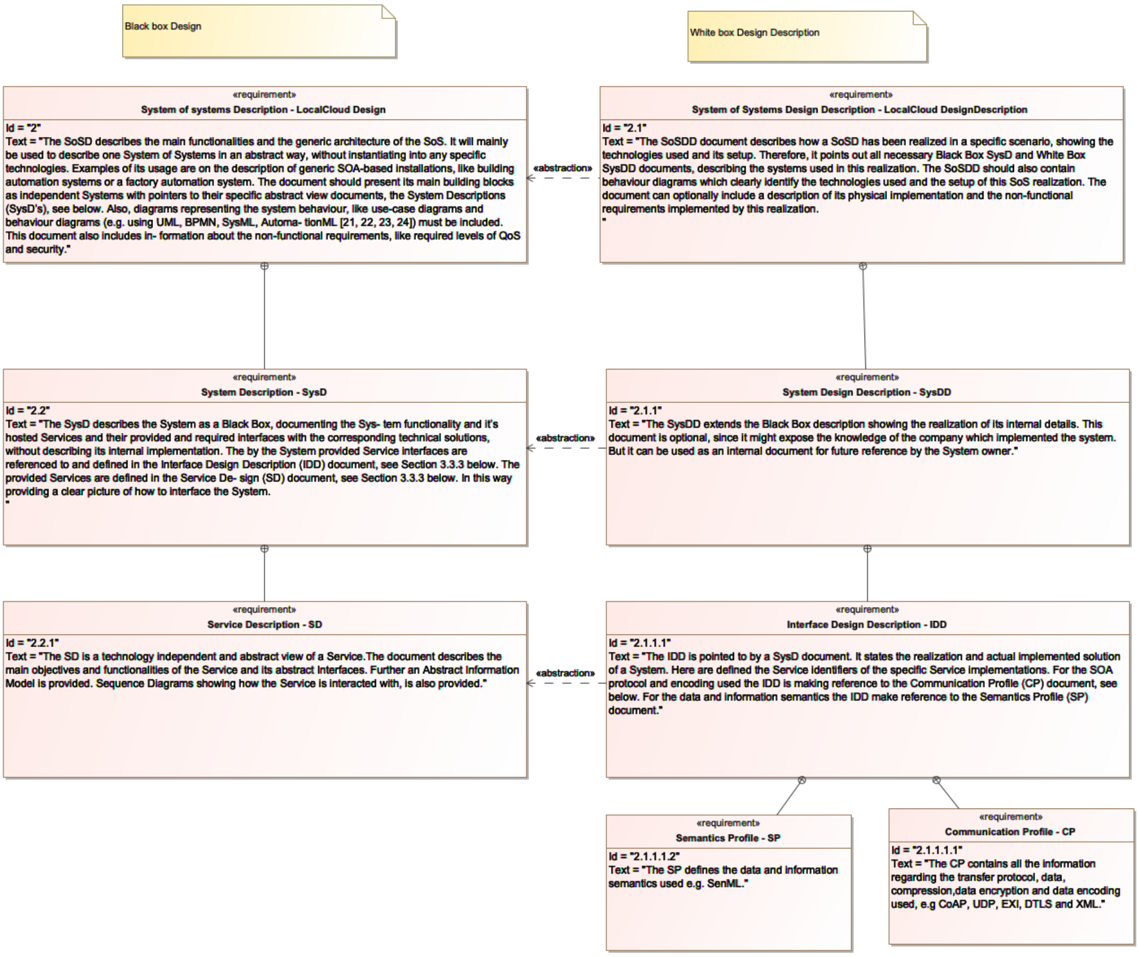
For the current implementations a set of technologies has been choosen for various reasons ranging from simple to use over “this are the technlogies I’m familiar to” to ther are the technologies “which provides what I what”. Technologies used for implementations of Eclipse Arrowhead microsystem are:

- Protocols: HTTP (REST), CoAP, MQTT, Websocket
- Network protocols: Ethernet, WiFi, 802.15.4
- Datamode standards used: SenML
- Security protocol: X.509 certificates and tokens, see **??**
- Programing language: Java, Python
- Libraries: Java, C
- High performance: GO lang

For documentation of the Eclipse Arrowhead and solutions based thereon a documentation structure has been adopted, cf. Figure 2.

# 3 References

[1] J. Delsing, Ed., *IoT Automation - Arrowhead Framework*. CRC Press, Feb. 2017, no. ISBN 9781498756754.



«requirement»
System of systems Description - LocalCloud Design
Id = "2"
Text = "The SoSD describes the main functionalities and the generic architecture of the SoS. It will mainly be used to describe one System of Systems in an abstract way, without instantiating into any specific technologies. Examples of its usage are on the description of generic SOA-based installations, like building automation systems or a factory automation system. The document should present its main building blocks as independent Systems with pointers to their specific abstract view documents, the System Descriptions (SysD's), see below. Also, diagrams representing the system behaviour, like use-case diagrams and behaviour diagrams (e.g. using UML, BPMN, SysML, Automa- tionML [21, 22, 23, 24]) must be included. This document also includes in- formation about the non-functional requirements, like required levels of QoS and security."

«requirement»
System of Systems Design Description - LocalCloud DesignDescription
Id = "2.1"
Text = "The SoSDD document describes how a SoSD has been realized in a specific scenario, showing the technologies used and its setup. Therefore, it points out all necessary Black Box SysD and White Box SysDD documents, describing the systems used in this realization. The SoSDD should also contain behaviour diagrams which clearly identify the technologies used and the setup of this SoS realization. The document can optionally include a description of its physical implementation and the non-functional requirements implemented by this realization.
"

«requirement»
System Description - SysD
Id = "2.2"
Text = "The SysD describes the System as a Black Box, documenting the Sys- tem functionality and it's hosted Services and their provided and required interfaces with the corresponding technical solutions, without describing its internal implementation. The by the System provided Service interfaces are referenced to and defined in the Interface Design Description (IDD) document, see Section 3.3.3 below. The provided Services are defined in the Service De- sign (SD) document, see Section 3.3.3 below. In this way providing a clear picture of how to interface the System.
"

«requirement»
System Design Description - SysDD
Id = "2.1.1"
Text = "The SysDD extends the Black Box description showing the realization of its internal details. This document is optional, since it might expose the knowledge of the company which implemented the system. But it can be used as an internal document for future reference by the System owner."

«requirement»
Service Description - SD
Id = "2.2.1"
Text = "The SD is a technology independent and abstract view of a Service.The document describes the main objectives and functionalities of the Service and its abstract Interfaces. Further an Abstract Information Model is provided. Sequence Diagrams showing how the Service is interacted with, is also provided."

«requirement»
Interface Design Description - IDD
Id = "2.1.1.1"
Text = "The IDD is pointed to by a SysD document. It states the realization and actual implemented solution of a System. Here are defined the Service identifiers of the specific Service implementations. For the SOA protocol and encoding used the IDD is making reference to the Communication Profile (CP) document, see below. For the data and information semantics the IDD make reference to the Semantics Profile (SP) document."

«requirement»
Semantics Profile - SP
Id = "2.1.1.1.2"
Text = "The SP defines the data and information semantics used e.g. SenML."

«requirement»
Communication Profile - CP
Id = "2.1.1.1.1"
Text = "The CP contains all the information regarding the transfer protocol, data, compression,data encryption and data encoding used, e.g CoAP, UDP, EXI, DTLS and XML."

Figure 2: The Eclipse Arrowhead documentation structure.

# 4 Revision History

## 4.1 Amendments

Revision history and Quality assurance as per examples below

| No. | Date | Version | Subject of Amendments | Author |
|---|---|---|---|---|
| 1 | 2023-05-08 | 4.6.1 | | Jerker Delsing |
| 2 | | | | |
| 3 | | | | |

## 4.2 Quality Assurance

| No. | Date | Version | Approved by |
|---|---|---|---|
| 1 | 2022-01-10 | 4.6.1 | |