ICANEPO Architecture

Deliverable:	D2.2
Full Title:	Architecture
Author(s):	Bernhard Rieder, AT BMJ
Version:	1.0
Date:	29/04/2024

Content

1. Introduction

ICANEPO - **I**nternational **C**ourt **A**pplicatio**N** for **E**uropean **P**ayment **O**rder – is an EU-funded project to develop an application for the management of court cases of the European Payment Order by judicial authorities. The application respects the requirements established by the EPO Regulation and will offer interfaces to nationally extend the application with implementations to respect national regulations building up on the EPO, such as languages, court fees, etc. The application will further support and enforce cross-border electronic communication via e-CODEX[1] as foreseen by the upcoming Judicial Digitalisation Regulation.

The project partners of Austria, Greece, Portugal and the Netherlands join forces within the 24 months of the project to create the application and to make it re-usable for further interested organisations.

[1] https://www.eulisa.europa.eu/Activities/Large-Scale-It-Systems/e-CODEX

2. Methodology

To design the software architecture for ICANEPO some principles and requirements had to be respected:

Business use-case independent core

It was the plan from the beginning of the project to build the application in a way that core functionality that can be re-used in future projects where other business use-cases will be implemented.

It was clear that the ICANEPO software is split into a use-case independent ICAN-Core and an ICANEPO extension that handles the EPO use-case specifically.

Exchangeability of services

Some parts of a Case Management System (CMS) might be in use already in some Member States (MS) which are probably embedded into national systems and environments. An example is that some MSs could have a Task Management System (TMS) already in place. The ICANEPO software should be able to integrate that TMS if the interfaces are respected.

An additional example could be an Identity Management (IDM) that is already used to authenticate and authorize users on a national level. It should also be possible to use that IDM in ICANEPO by implementing industry standards for authentication and authorization.

Customization of services

Business use-cases that derive from European regulations and directives must have national adaptations.

In the case of EPO the best example is, that it cannot be regulated on a European level how high the court fees are and how they are treated.

For those national adaptations it must be possible to implement national extensions into the ICANEPO software.

The software therefore offers interfaces that make it possible to be implemented and hooked into it.

Best possible independence from runtime environment

ICANEPO is a software that must be installed in a national IT environment.

As such IT environments can be very challenging and different, the software should use containerization to minimize the adaptations necessary for using it.

Architecture documentation

To document the architecture of the ICANEPO software, the arc42[2] template is used. The original structured architecture is written and stored within the documentation repository of the development environment. The documentation below is a consolidated version of the structured architecture documentation.

[2] https://arc42.org/

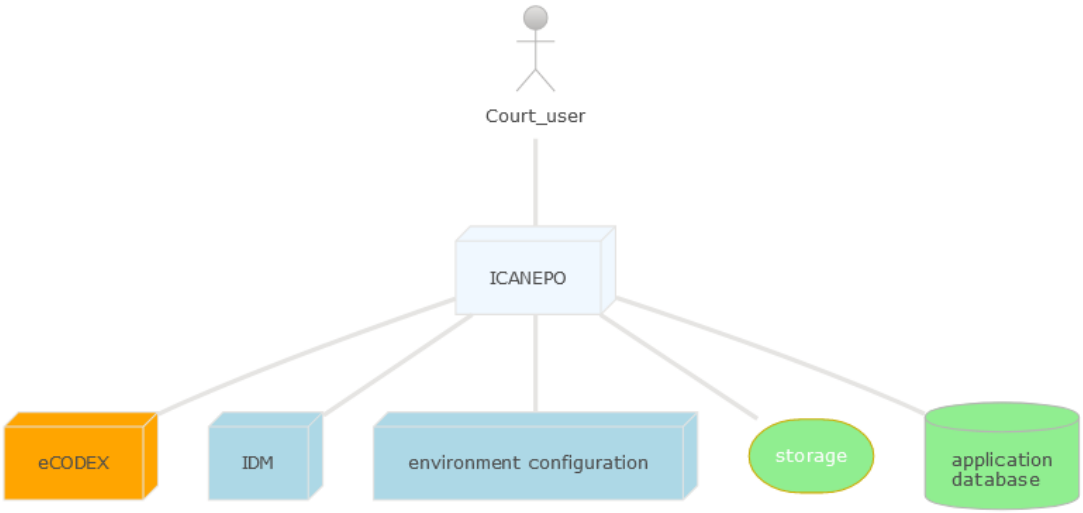
3. Architecture

It soon became clear that only a modular approach for an architecture would fulfil the goal of the project. One of the non-functional requirements was to have different modules, which could be replaced by already existing national modules. This would guarantee, that the ICANEPO application can be well integrated into the national application landscape. Also, a modular approach would be beneficial for future maintenance, as each module would be independent from other modules and thus easier to maintain, adapt and develop.

Since the arc42 template is used to document the software architecture, the following graphics represent different layers from different sections of the arc42 documentation.

Building block view

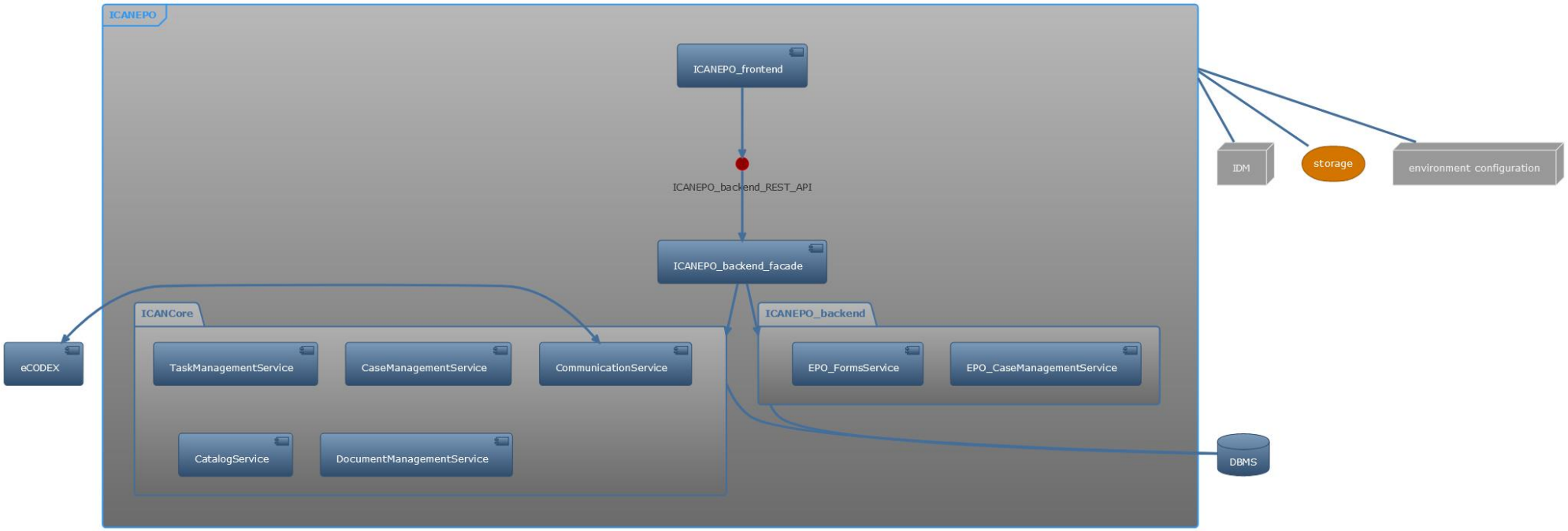
The graphic shows the overall whitebox view of the architecture:



This overall view shows surrounding systems that are required by the application.

Going one layer deeper into the system, the differentiation between a common core and an EPO specific implementation can be seen already.

The graphic shows the first layer of the software architecture:


ICANEPO
ICANEPO_frontend
ICANEPO_backend_REST_API
ICANEPO_backend_facade
ICANCore
TaskManagementService
CaseManagementService
CommunicationService
CatalogService
DocumentManagementService
ICANEPO_backend
EPO_FormsService
EPO_CaseManagementService
eCODEX
IDM
storage
environment configuration
DBMS

Here, the core services of the ICANEPO software can be seen. It has an EPO specific frontend which includes the graphical user interface. This cannot be split into use-case independent core and EPO implementation since the user interface is by design too explicit.

Between the frontend and the backend, a façade is used as software implementation pattern. It knows the interfaces from both ends, frontend, and backend, and translates between them.

The backend of the ICANEPO software is split into the ICAN-Core and the ICANEPO_backend. This way services can be implemented that are independent from each other. That makes them exchangeable, extendable, adaptable, and reusable. It also demonstrates that core services and services from the ICANEPO backend are building up on each other.

Runtime View

An exemplary scenario shows the interaction between the core services and the EPO specific services.

Incoming e-CODEX message leads to new task for organizational entity

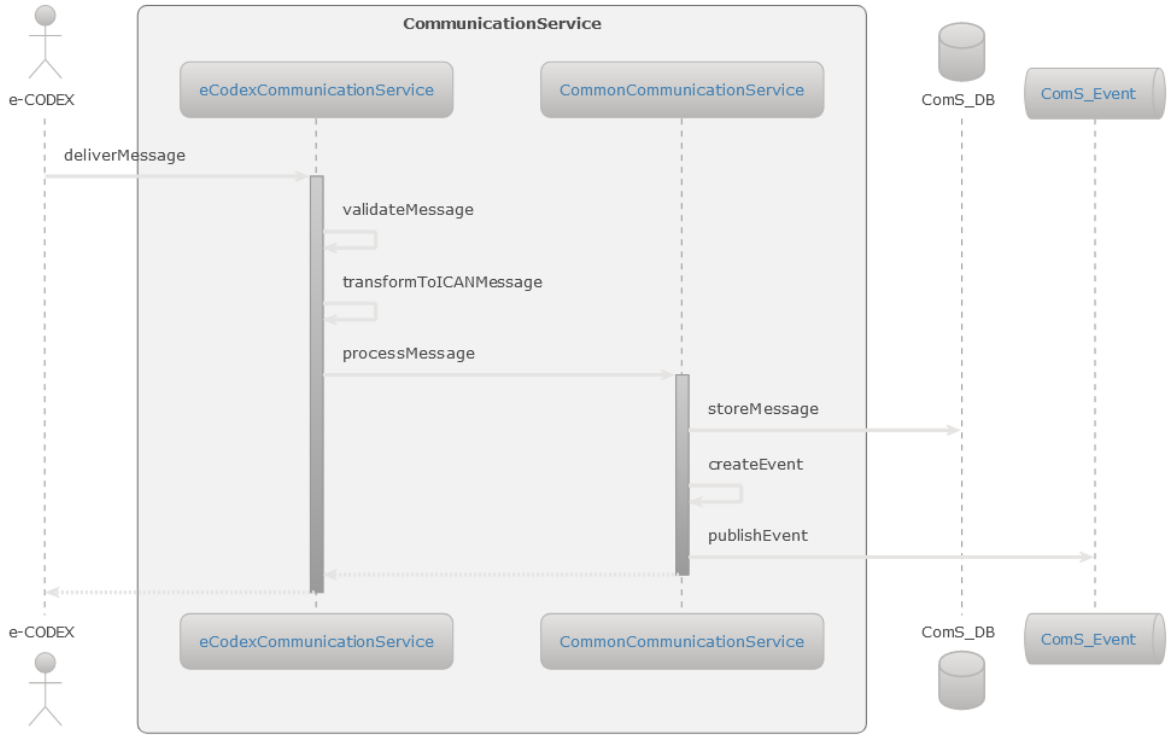
This scenario describes what happens, if an e-CODEX EPO business message is forwarded to ICANEPO.

The expected result of the scenario is:

- a Task for a user that assigns new cases to organizational entities within the user group if the e-CODEX EPO business message contains a new Form A that is not a case yet
- a Task for the user responsible to take next steps within the case if the e-CODEX EPO business message contains a follow-up form to the case.
- the confirmation to e-CODEX that the message is delivered to its final recipient.

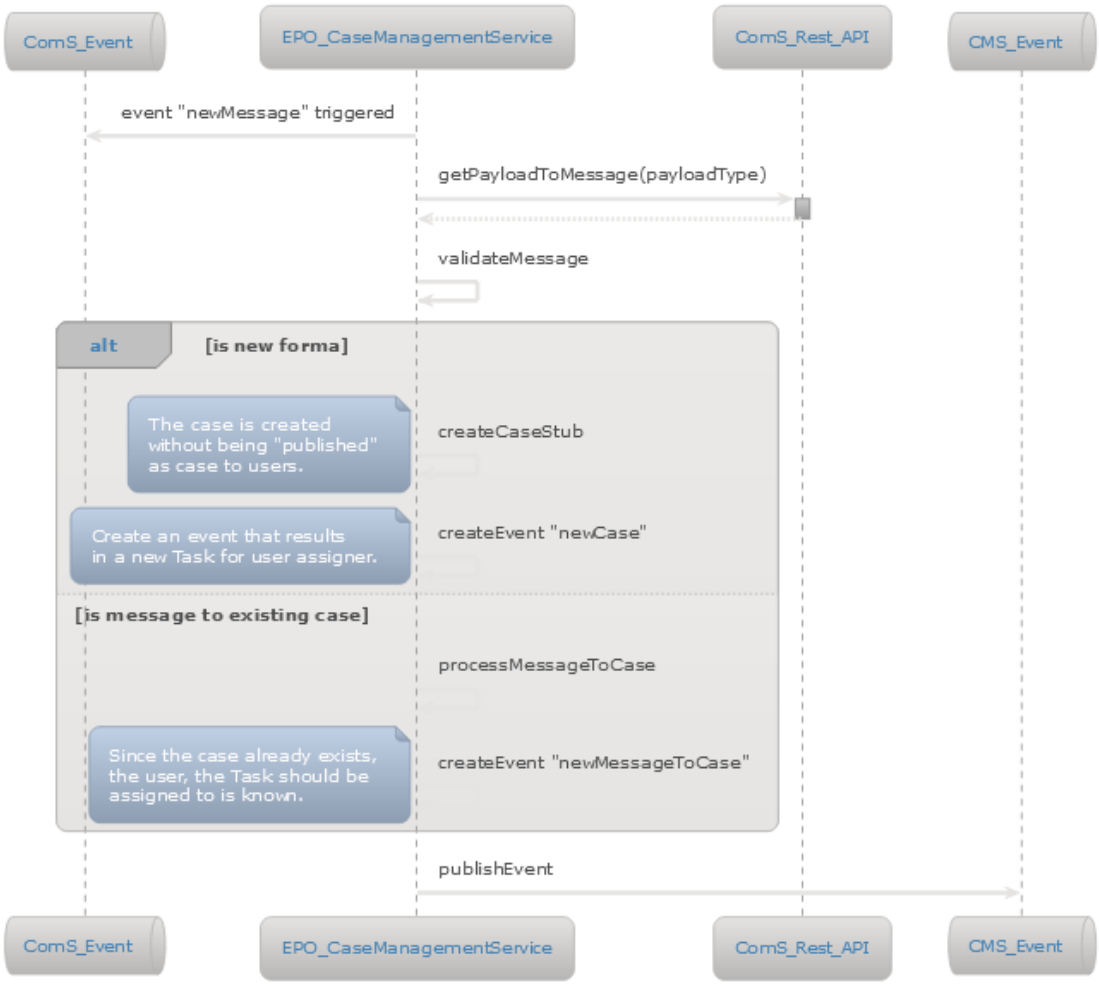
Incoming e-CODEX message processing

The sequence diagram shows how the incoming e-CODEX business message is received by ICANCore, transformed into an ICANMessage and forwarded to ICANEPO application.

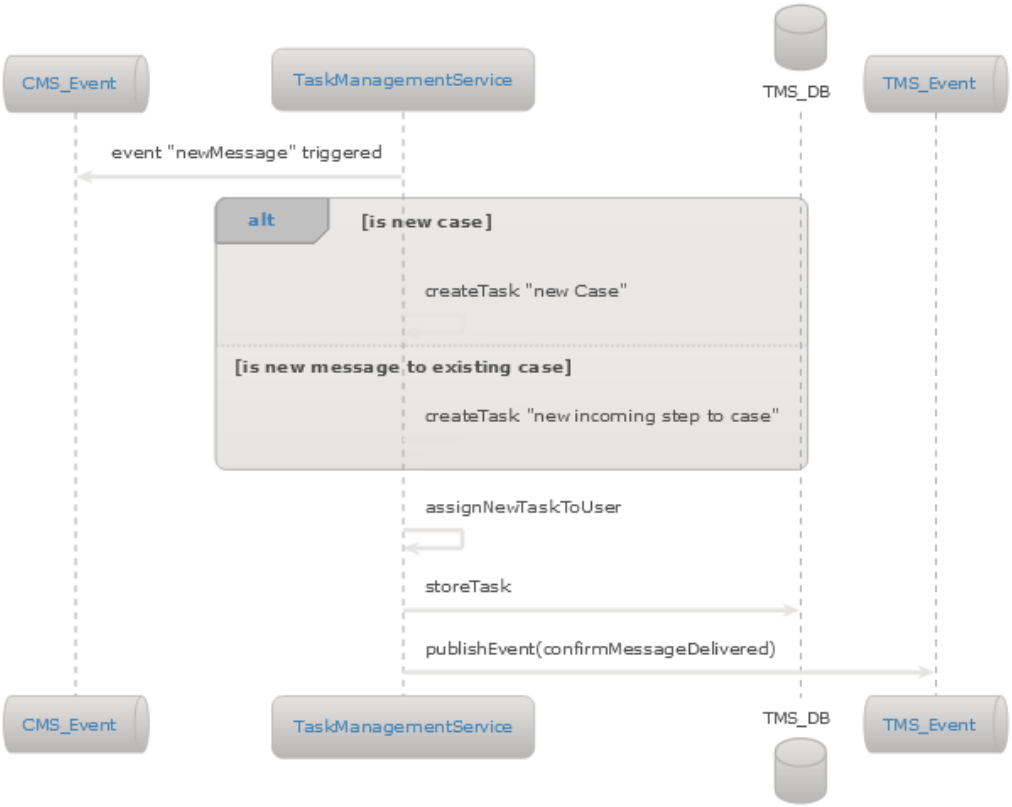


ICANEPOCase to create new Task event

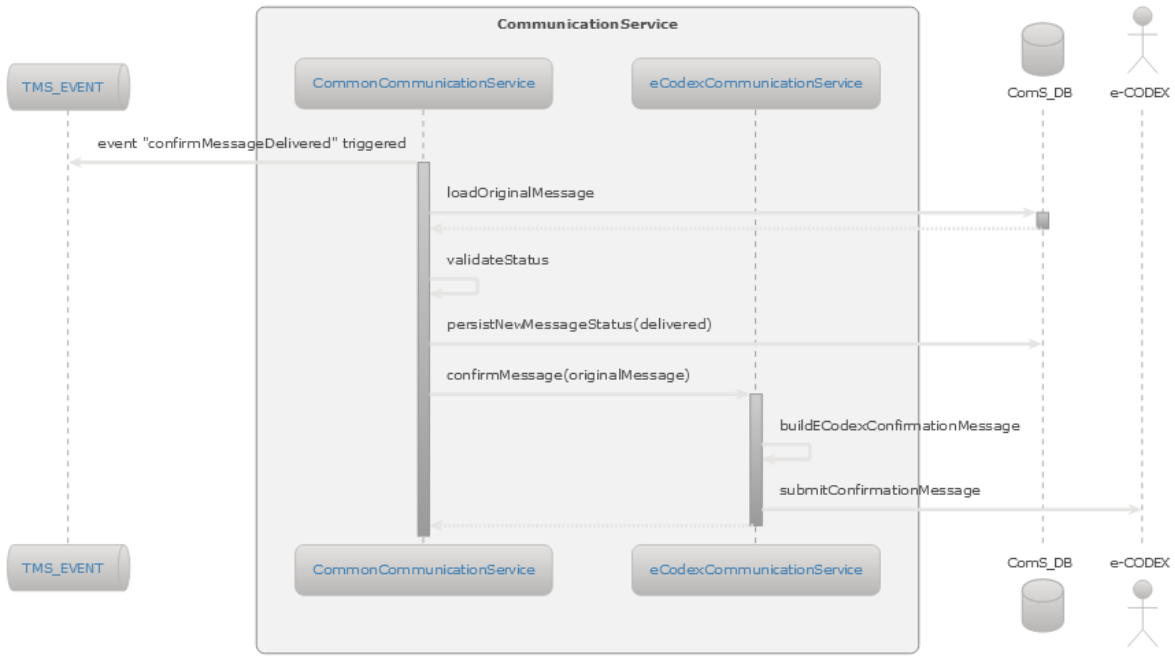
Diagram on a rather high level to show how a "newMessage" event is consumed and processed inside ICANEPOCase leading to an event to the TaskManagementService to create a task.



TaskManagementService to create and publish new task to user



CommunicationService to send to e-CODEX delivery confirmation trigger



4. Conclusion & Outlook

The software architecture is chosen to fulfil all requirements that are given by the project. The requirements are a direct result of the requirements engineering including interviews and study visits.

Just as the software implantation process is ongoing also the software architecture is adapted and extended along with the findings during the implementation phase.

So the architectural model described in this document is a living approach that needs to be adapted and extended through the whole project's lifetime.

The architectural documentation in arc42 will be part of the software delivery and will be made public alongside the result of the project.