D4.1

Software State of the Art

This project has received funding from the European Union's Horizon 2020 Innovation Programme under Grant Agreement No 822717.

Project

Acronym: **MICADO**
Title: Migrant Integration Cockpits and Dashboards

Coordinator: HafenCity University Hamburg

Reference: 822717
Type: Innovation Action
Programme: HORIZON 2020
Theme: Addressing the challenge of migrant integration through ICT-enabled solutions (DT-MIGRATION-06-2018)

Start: 1 January 2019
Duration: 42 months

Website: www.micadoproject.eu

Consortium:

- **HAFENCITY UNIVERSITÄT HAMBURG** (HCU), Germany
- **FREIE UND HANSESTADT HAMBURG** (FHH), Germany
- **HAMBURGISCHES WELTWIRTSCHAFTSINSTITUT GEMEINNÜTZIGE GMBH** (HWWI), Germany
- **UNIVERSITEIT ANTWERPEN** (UANTWERPEN), Belgium
- **OPENBAAR CENTRUM VOOR MAATSCHAPPELIJK WELZIJN VAN ANTWERPEN** (OCMW Antwerpen), Belgium
- **INTEGRATIE EN INBURGERING ANTWERPEN** (Atlas Antwerpen), Belgium
- **DIGIPOLIS** (DIGIPOLIS), Belgium
- **ALMA MATER STUDIORUM - UNIVERSITA DI BOLOGNA** (UNIBO), Italy
- **AZIENDA PUBBLICA DI SERVIZI ALLA PERSONA CITTA DI BOLOGNA** (ASP Bologna), Italy
- **CONSORZIO PER IL SISTEMA INFORMATIVO** (CSI PIEMONTE), Italy
- **COLEGIO PROFESIONAL DE POLITÓLOGOS Y SOCIÓLOGOS DE LA COMUNIDAD DE MADRID** (CPS), Spain
- **CONSEJERIA DE POLITICAS SOCIALES Y FAMILIA COMUNIDAD DE MADRID** (CPSF-DGSSIS-CM), Spain
- **UNIVERSIDAD REY JUAN CARLOS** (URJC), Spain
- **TECHNISCHE UNIVERSITÄT WIEN** (TU WIEN), Austria
- **SYNYO GmbH** (SYNYO), Austria

Deliverable

Number:	**D4.1**
Title:	**Software State of the Art**
Lead beneficiary:	CSI
Work package / Task:	WP4 / T4.1
Dissemination level:	Public (PU)
Nature:	Report (RE)
Due date:	30/11/2019
Submission date:	30/11/2019
Authors:	**Luca Gioppo**, CSI
Contributors:	Benedikt Seitzer (HCU) Jan Blondé (DIG) Micael Gallego Carrillo (URJC) Carmen Munteanu (SYNIO)
Review:	Jörg Noennig (HCU) Jan Barski (HCU) Nicole Schubbe (FHH)

Acknowledgement: This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No 822717.

HISTORY OF CHANGES			
version	**date**	**Comment**	**author(s)**
V 1.0	01/08/2019	First draft	Luca Gioppo (CSI)
V 1.1	13/09/2019	Completed content and collected feedback from WP4 partners and co-creation	Luca Gioppo (CSI) Benedikt Seitzer (HCU)
V 1.2	14/09-15/11/2019	Contributions	Jan Blondé (DIG), Micael Gallego Carrillo (URJC), Carmen Munteanu (SYNYO)
V 1.3	26/11/2019	Reviews	Jörg Noennig (HCU), Jan Barski (HCU), Nicole Schubbe (FHH)
V 2.0	28/11/2019	Final version	Luca Gioppo (CSI)

Executive summary

This document, building on the work done in D1.3, will detail the software state of the art of the software components needed to develop the MICADO solution.

The methodology used to evaluate the software will be explained; the main requirements for each component will be described and the evaluation process will also be explained, first in general and, where needed, for specific components should it require specific evaluation.

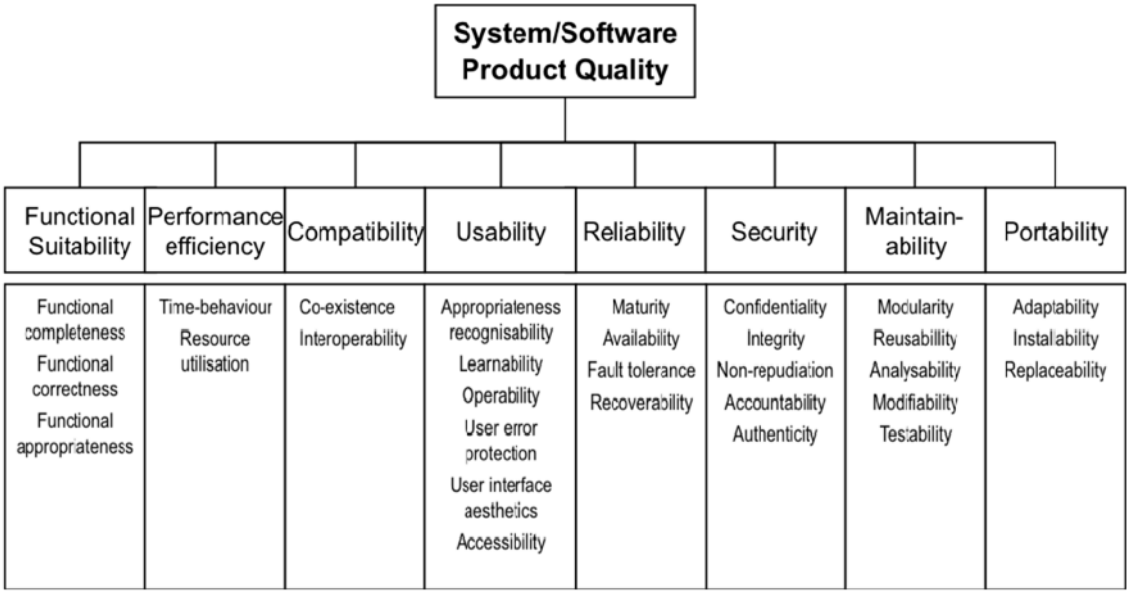
These components will be part of an architecture that will be briefly described here, as it will be addressed in D4.2.

Table of Content

1 General components requirements and evaluation methodology

MICADO will inspire its software evaluation from the ISO 25010 standard identifying the characteristics that are meaningful for the scope of the project.

The ISO 25010 proposes a framework for evaluating software based on a set of characteristics summarized in the table below.



Each characteristic will be described adding a context of application within MICADO, considering that some characteristics have different values for the two different stakeholders of the system: the end user, the development team.

For the final user the MICADO solution will be seen as an integrated ‘unicum’, thus the characteristics of the complete platform will receive a partial contribution from the single components; for the development team the selection will be based on elements invisible to the end users besides the ones that will contribute to the final outcome.

MICADO will also assign other evaluation criteria for the evaluation of the components: partner readiness and partner ownership.

Partner readiness	Represents partner’s existing knowledge in the examined component, giving the project a boost in the productivity during development either by having the partner being directly involved in the development or giving support to the team
Partner ownership	Represents partner’s willingness to own the component even after the live of the project.

These criteria will help discriminate between equal software products; having a partner already skilled in the product and also willing to continue to use the software deliverable even after the

life of the project will guarantee the exploitability of the solution and the life beyond the time scope of the MICADO project.

1.1. Evaluation framework

1.1.1 Functional suitability

Degree to which a product or system provides functions that meet stated and implied needs when used under specified conditions

sub-category	Definition	end user	DevOp	MICADO context
functional completeness	degree to which the set of functions covers all the specified tasks and user objectives		x	This is the core category used to determine if a component does what the project needs
functional correctness	degree to which a product or system provides the correct results with the needed degree of precision	x	x	this category applies to component like dashboard or chatbot where the interaction with the end user could require a level of precision or accuracy (the accuracy of a map or the proper response of the chatbot to the user interaction)
functional appropriateness	degree to which the functions facilitate the accomplishment of specified tasks and objectives	x	x	In MICADO the main goal is to ease the relation between the migrant and the Public context, all the features that the component can offer to implement this new paradigm are evaluated under this category

1.1.2 Performance efficiency

Performance efficiency is performance relative to the amount of resources used under stated conditions.

sub-category	Definition	end user	DevOp	MICADO context
Time-behaviour	Degree to which the response and processing times and throughput rates of a product or system, when performing its functions, meets requirements			
Resource Utilisation	Degree to which the amounts and types of resources used by a product or system, when processing its functions, meets requirements		x	For MICADO, it is important to choose components that use as less resources as possible since the final solution will have to be deployed on premise of the Public Administration and the less resources are needed the higher are the chances to be adopted
Capacity	Degree to which the maximum limits of the product or system, parameter meets requirements			

1.1.3 Compatibility

Compatibility is the degree to which a product, system or component can exchange information with other products, systems or components, and/or perform its required functions, while sharing the same hardware or software environment.

sub-category	Definition	end user	DevOp	MICADO context

Co-existence	Degree to which a product can perform its required functions efficiently while sharing a common environment and resources with other products, without detrimental impact on any other product		x	This is a partial characteristic since is linked to resource utilization; if a component needs to have a specific dedicated environment or dedicated hardware it could add resources consumption. On all other cases since the deployment strategy of the platform will be by using container technology this category should be neutralized
Interoperability	Degree to which two or more systems, products or components can exchange information and use the information that has been exchanged		x	This is one of the most important categories for the Dev actor: MICADO needs to have tools that can interoperate using API

1.1.4 Usability

Usability is the effectiveness, efficiency and satisfaction in a specified context of use. Usability can either be specified or measured as a product quality characteristic in terms of its sub-characteristics or specified or measured directly by measures that are a subset of quality in use.

sub-category	**Definition**	**end user**	**DevOp**	**MICADO context**
Appropriateness Recognisability	Degree to which users can recognise whether a product or system is appropriate for their needs			
Learnability	Degree to which a product or system enables the user to learn how to use it with effectiveness, efficiency in emergency situations		x	Components need to be easy to learn for the development team since there is a fairly high number

				of components to manage
Operability	Degree to which a product or system is easy to operate, control and appropriate to use		x	Operation activity will have to be well explained and each component will need to have a clear management documentation
User Error Protection	Degree to which a product or system protects users against making errors			
User Interface Aesthetics	Degree to which a user interface enables pleasing and satisfying interaction for the user. In the case of the Switching Programme, the term “user” refers to the system user, not a market participant, consumer, or other user.			This will not be a requirement since all the components will be used through API in a headless approach
Accessibility	Degree to which a product or system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use. Includes the concept of access control where specific users have limited access to system features and functions.			This will not be a requirement since all the components will be used through API in a headless approach

1.1.5 Reliability

Reliability is the degree to which a system, product or component performs specified functions under specified conditions for a specified period of time.

sub-category	**Definition**	**end user**	**DevOp**	**MICADO context**
Maturity	Degree to which a system, product or component meets needs for reliability under normal operation		X	Since the project will look for its components in the Open Source ecosystem the care in finding a

				mature product will be high
Availability	Degree to which a product or system is operational and accessible when required for use			
Fault Tolerance	Degree to which a system, product or component operates as intended despite the presence of hardware or software faults			The fault tolerance of the MICADO architecture will be delegated to its deployment model
Recoverability	Degree to which, in the event of an interruption or a failure, a product or system can recover the data directly affected and re-establish the desired state of the system			The recoverability of the MICADO architecture will be delegated to its deployment model

1.1.6 Security

Degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization

sub-category	**Definition**	**end user**	**DevOp**	**MICADO context**
confidentiality	degree to which a product or system ensures that data are accessible only to those authorized to have access		x	Each component will have to offer an ACL system to allow the definition of a set of MICADO roles that will span through different software products
integrity	degree to which a system, product or component prevents unauthorized access to, or		x	As above the ACL system will have to enforce this requirement

	modification of, computer programs or data			
non-repudiation	degree to which actions or events can be proven to have taken place, so that the events or actions cannot be repudiated later			MICADO does not have this requirement for the single components
accountability	degree to which the actions of an entity can be traced uniquely to the entity			MICADO does not have this requirement for the single components
authenticity	degree to which the identity of a subject or resource can be proved to be the one claimed			MICADO components will be used as backend software mostly used through internal account users

1.1.7 Maintainability

Maintainability is the degree of effectiveness and efficiency with which a product or system can be modified by the intended maintainers. Modifications can include corrections, improvements or adaptation of the software to changes in environment, and in requirements and functional specifications. Modifications include those carried out by specialized support staff, and those carried out by business or operational staff, or end users. Maintainability includes installation of updates and upgrades. Maintainability can be interpreted as either an inherent capability of the product or system to facilitate maintenance activities, or the quality in use experienced by the maintainers for the goal of maintaining the product or system.

sub-category	**Definition**	**end user**	**DevOp**	**MICADO context**
Modularity	Degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components		x	This is a fundamental requirement for MICADO
Reusability	Degree to which as asset can be used in more than one system, or in building other assets			

Analysability	Degree of effectiveness and efficiency with which it is possible to assess the impact on a product or system of an intended change to one or more of its parts, or to diagnose a product for deficiencies or causes of failures, or to identify parts to be modified		x	Components will have to produce usable logs that will be centrally collected to allow to manage the complexity of the system and help operations to manage it
Modifiability	Degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing product quality			This is not a requirement since the goal of MICADO is to use product without modifying it.
Testability	Degree of effectiveness and efficiency with which test criteria can be established for a system, product or component and tests can be performed to determine whether those criteria have been met			In the selection MICADO assume that software components are "production ready" and tested. Testing will be done at integration level.

1.1.8 Portability

Portability is the degree of effectiveness and efficiency with which a system, product or component can be transferred from one hardware, software or other operational or usage environment to another.

sub-category	**Definition**	**end user**	**DevOp**	**MICADO context**
Adaptability	Degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage environments			MICADO needs to find software that can be deployed on different Public Administration data centre: this will be achieved by using container technology so this category will be satisfied by this architectural decision
Installability	Degree of effectiveness and efficiency in which a product or system can be successfully			As with the previous category the

	installed and/or uninstalled in a specified environment			deployment strategy will satisfy this need.
Replaceability	Degree to which a product can replace another specified software product for the same purpose in the same environment		x	This is an important strategical category for MICADO since the need to exchange a single component could be high: it could be surpassed in time by other open source solution or the end user could want to continue to use its own existing technology for a specific MICADO's task

1.2. Final evaluation set

Evaluation element	**end user**	**DevOp**
Functional completeness		x
Functional correctness	x	x
Functional appropriateness	x	x
Resource Utilisation		x
Co-existence		x
Interoperability		x
Learnability		x
Operability		x
Maturity		x
Confidentiality		x
Integrity		X
Modularity		X
Analysability		x

Replaceability		x

All these evaluation elements will have an easy scoring system for this selection:

0	Feature not satisfied
1	Feature partially satisfied
2	Feature fully satisfied

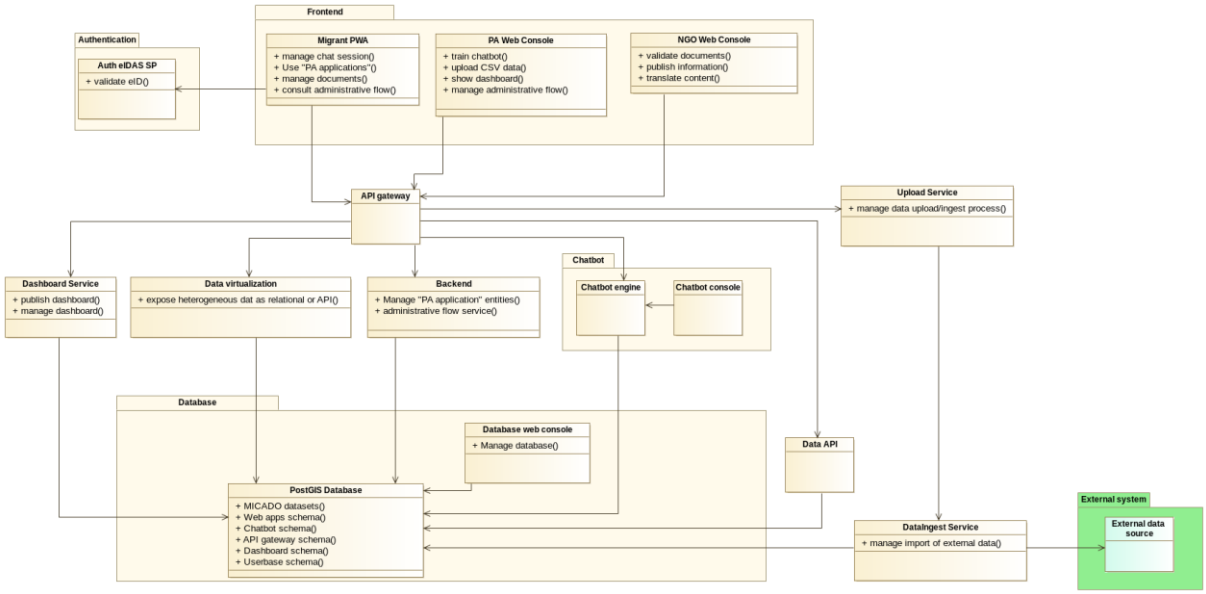
Functional elements will be matched against the specific technical requirements listed in each component chapter and harmonized in the above scale.

Another impact on the evaluation process will be the competence in the technology of the partners that will keep the MICADO solution alive after the end of the project: to ease the sustainability of the MICADO solution in presence of different existing solutions it will be given the preference to software components already in use or adoptable by the future maintaining partner(s).

2 Architectural draft and deployment model

The following architectural diagram represents a draft of the final MICADO's architecture.

The diagram shows the main components, however represented as a 'black box', whose content will be determined in detail once the project will choose to adopt a particular software product for the implementation. As can be seen, the MICADO architecture foresee the integration with external sources of the Public Administration through a specific component that will enable data exchange through heterogeneous approaches.



The design approach is to have three separated frontends: one for the migrants (that will have constraints on usability and devices on which it will run) designed to be used on a smartphone

and the other two for Public Administration and NGOs that can be a more traditional web application.

Both will relate to the backend through an API gateway that will provide a single and homogeneous security and access point towards the different set of products providing the frontend with a single black box approach.

In this way the components in the backend can be changed, updated, evolved, and maintained (provided that the same API contract is respected), without affecting the frontend application.

This is a strong requirement for the architecture since, as can be seen, it will be composed by a heterogeneous set of software products, while there is the need to offer to the frontend a single overall view.

In the backend we find a main database that must have spatial capabilities to allow the representation of geographically and spatially represented data.

On this, in different schemas, all the components will persist their needed information whether it be simple configuration or business-related data.

Other components that will be part of the MICADO design are:

- Chatbot service
- Dashboard service
- Ingestion module
- Management service
- Upload service

In the initial design options, there was the ambition to include also features to manage OpenBadges for the certification of skills; during the co-creation workshops done in the WP2 tasks this proposal did not receive high interest by the stakeholders, more interested in the other aspects of the project. It has, thus, been decided to set this feature set aside for a re-evaluation after the second iteration of the development for the final inclusion decision.

Another important aspect of the architecture is the concept of ‘swappability’: when possible some components could be swapped with existing, compatible components already present on premise of the adopting administration. This choice is not applicable to all components of the platform and it will be marked on specific components

2.1. Deployment model

The deployment model for MICADO will be done using a containerization approach.

This means that all the software processes of every components will have its own docker image and one or more dedicated container.

Virtual networks will be used to connect different containers and the overall topology will be described in a file that will be used to “run” the MICADO solution.

This approach will enable to adopt a first development approach to use a simple single machine minimal installation useful for demo and development purpose and to migrate on a clustered environment.

This means that the first development solution will be based on a docker-compose solution, while the production solution will be proposed as a Kubernetes workload.

3 Components state of the art

In this chapter the main components listed in the architectural diagram are evaluated against the state of the art available in the open source ecosystem. This to converge on the identification of a candidate to be used in the MICADO's implementation.

3.1. Identity service

User authentication and management is a main feature of each software solution. In MICADO, this requirement overlaps with the need to properly manage user data within the GDPR compliance and provide easy access for users allowing them to use existing credentials obtained from official existing services.

Usage in MICADO

One of the main goals is to follow the citizen centric approach and ask for as little information as possible.

This will start from the login option that will integrate with the eID standard to try to recover the user information through the already delivered digital identities.

The goal is to provide a Shibboleth Service Provider (SP) properly integrated with the Identity Providers (IdP) so that the end user will be able to use its own credential in the system.

This will allow MICADO to identify the user safely and following EU standard.

Both Italy and Germany have their national digital ID system integrated with the eID standards.

Effort will be done to evaluate the usability of these components in the context of migrants.

Solutions will have to be taken in the cases where migrant users do not jet belong to the formal digital ID environment.

Technical requirements

Since eIDAS is a EU proposed standard is not easy to find products compliant with these protocols.

The effort to keep the pace with the evolution of these sensitive and evolving components calls for a product that manage to comply "out of the box" to them.

Technical requirements
MUST adopt eIDAS protocol
MUST manage different set of user groups with different authentication schema

Software state of the art

A complete solution in this environment is the WSO2 Identity Server, one of the few open source software of enterprise level to be compliant with the eIDAS specification.

There is also the integration package, released by the EU as a sample of how to set up an eIDAS NODE, available.

Product	Licence	Web site	Release
eIDAS integration package	EUPL	https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/eIDAS-Node+Integration+Package	2.3.1
WSO2 Identity Server	Apache License, Version 2.0	https://rasa.com/	5.9

Evaluation

Evaluation element	WSO2 Identity Server	eIDAS kit
Functional completeness	2	1
Functional correctness	2	2
Functional appropriateness	2	2
Resource Utilisation	2	2
Co-existence	2	2
Interoperability	2	1
Learnability	2	1
Operability	2	1
Maturity	2	1
Confidentiality	2	2
Integrity	2	2
Modularity	2	2
Analysability	2	2
Replaceability	2	2
Totals	**28**	**23**

Project choice

Considering also other choices done in the overall architecture and the availability of a commercial product with many features useful for the project AND compliant with the EU standard the choice for the platform is WSO2 Identity Server.

3.2. API gateway

Usage in MICADO

An API gateway component is needed to decouple the frontend application from the heterogeneous number of backend services that the MICADO solution will integrate.

Technical requirements

The main requirements for this component are:

- high efficiency: the gateway needs not to introduce latency or overhead in the communication
- enforce security: the gateway can help providing access security even to backend services that could not have an own solution
- ease of use: the gateway needs to offer management GUI, possibly with mocking feature so that it will be possible for the development team to simulate the existence of a proper backend endpoint (versioning the potential evolution of the API) without having to depend on the real backend system existence. This is an important feature given the distributed nature of the development team of the consortium.

Software state of the art

Product	Licence	Web site	Release
Kong	Apache License, Version 2.0	https://konghq.com/	1.4
WSO2 API manager	Apache License, Version 2.0	https://wso2.com/api-management/	3.0
Fusio	GNU Affero General Public License v3.0	https://www.fusio-project.org/	1.8.0

The candidates are: WSO2 API manager, KONG and Fusio. All are open source components with similar set of features

WSO2 provides API Manager features include design and prototyping for SOAP- or REST-style APIs, governance policies, access control with OAuth2, monetization, and prototyping support; it accepts OpenAPI (Swagger) definitions, manages throttling and access level and integration with Identity provider

The Kong Community Edition is a microservices API gateway; platform-agnostic offers API and microservices management features: WebSockets, common language infrastructure

capability, and great visualizations for monitoring. Many features are released as commercial plugins.

Fusio allows API construction from various data types. Fusio comes with lifecycle management features such as a back-end dashboard for admin control. It also includes developer portals, documentation, JSON validation for incoming requests, rate limiting, and scope handling to match user permissions.

API Management Features

	WSO2 API manager	Kong	Fusio
Access Control	x	x	x
Analytics	x	x	x
API Design/Prototyping	x		
API Lifecycle Management	x		x
Dashboard			x
Developer Portal	x		x
Testing Management	x		
Threat Protection	x		
Traffic Control	x	x	
Version Control	x		x

Evaluation

Evaluation element	RASA	Chatterbot	Botpress
Functional completeness	2	1	1
Functional correctness	2	2	2
Functional appropriateness	2	2	2
Resource Utilisation	2	2	2
Co-existence	2	2	2
Interoperability	2	1	2
Learnability	1	1	1
Operability	2	1	1
Maturity	2	1	2
Confidentiality	2	2	2

Integrity	2	2	2
Modularity	2	2	2
Analysability	2	2	2
Replaceability	2	2	2
Totals	**27**	**23**	**25**

Project choice

Mainly due to the prototyping feature and the experience in having this software already in operation in enterprise context by one of the technical partners of MICADO, the WSO2 API manager will be the product chosen for the API gateway features of the MICADO's architecture.

Swappability

In a production environment the API gateway represents one of the potential swappable components of the architecture since its main task if to be transparent.

3.3. Frontend technology

Modern frontend solution leverage on JavaScript framework to develop web client interfaces.

These clients interact with the backend to provide the MICADO's features to the end users.

Usage in MICADO

As previously stated there will be three web clients. One to be used mainly on the smartphone, the other two exclusively as a web interface.

Technical requirements

Since the project cannot adopt different framework for the three developments, for efficiency reasons, the chosen framework will have to be enough versatile to provide the whole spectrum of requirements.

On these web applications will converge the entire set of features; interacting with a chatbot, consulting PA's processes flows, uploading documents, getting information using audio medium.

This will require a framework with a good community that will offer the needed integrations with the specific libraries that provide the specific integrations.

Technical requirements
MUST have a chatbot conversation widget
MUST generate PWA clients
MUST manage multilingual interfaces

MUST be lightweight to be used on smartphones without requiring too many resources or bandwidth

Software state of the art

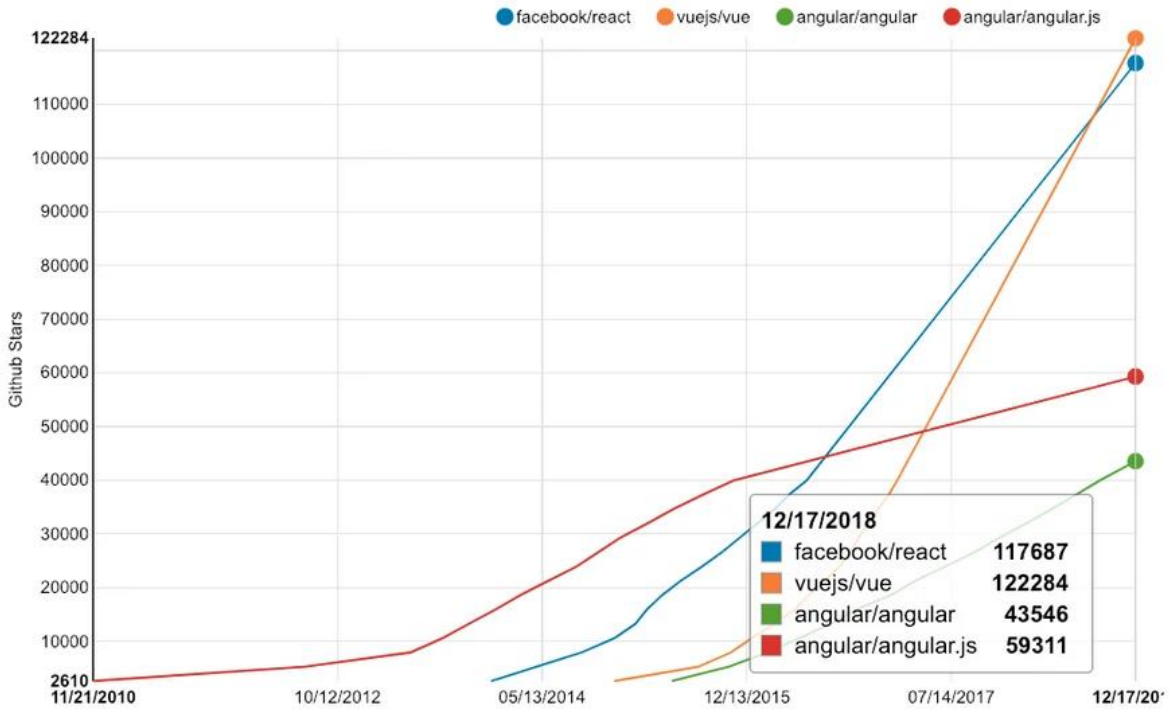
The three main competitors in this area are the JavaScript frameworks: Vue.js, Angular and React.

These are widely used solutions supported and adopted by big players; in terms of feature completeness are similar; the distinction is on the performances and small footprint on the client side.

Another important factor is that the framework will have to be quickly learned by the development teams not already skilled in it and thus the need for easiness in adoption is another distinguishing element.

Product	Licence	Web site	Release
VUE.js	MIT Licence	vuejs.org	3.12
Angular	MIT Licence	angular.io	8
React	MIT Licence	reactjs.org	16.6.3

	Angular	React	Vue
Initial release	2010	2013	2014
Approx. size (KB)	500	100	80
Used by	Google, Wix	Facebook, Uber	Alibaba, GitLab



Evaluation

Evaluation element	VUE.js	Angular	React
Functional completeness	2	2	2
Functional correctness	2	2	2
Functional appropriateness	2	2	2
Resource Utilisation	2	1	1
Co-existence	2	2	2
Interoperability	2	1	2
Learnability	2	1	1
Operability	2	2	2
Maturity	2	2	2
Confidentiality	2	2	2
Integrity	2	2	2
Modularity	2	2	2

Analysability	2	2	2
Replaceability	2	2	2
Totals	**28**	**25**	**26**

Project choice

Considering the need of versatility and lightweights on the browser side the choice of MICADO goes in the VUE.js framework.

3.4. Chatbot service

Brief explanation of what the chatbot will do

Usage in MICADO

Chatbot will be used to convey FAQ information in a more user-friendly way to users that could relate to a smart search engine that can guide the user towards the answer he was looking for.

In MICADO the chatbot will have a user interface in the MICADO app that will talk to a server component sending the user messages to the server and getting the interactive answers in return.

The data will be designed to provide a known set of questions, leaving the answers to be customized by each pilot site with the proper content from the PA.

The question set will emerge from the co-creation tasks and will represents the most required information by the individual users.

This set of questions will be placed on an EXCEL spreadsheet to allow the end user to provide the answers in an easy way; this file will be than ingested by the MICADO system and go to represent the answers given by the chatbot.

Questions will be presented to the end user not in a written manner but as a set of options read and shown to the end user as graphical buttons so that he will be guided in the chatbot interaction.

This design approach is due to the impossibility to build a chatbot able to properly understand spoken language in all the migrants' languages; instead, it uses the chatbot ability to interact with the end user through a guided set of stories.

Answers will not be written as plain text in the chatbot story but will be saved as single code hash that will be used to retrieve the proper language translation of the answer.

This approach will also help training the chatbot.

The design and integration effort will be to associate audio feedback with the interaction.

A requirement for this feature is that the migrant will have to declare the language that he speaks and understand or the chatbot will have to fall back on the local language.

Technical requirements

The chatbot software will have to satisfy these requirements:

Technical requirements
MUST manage stories presented also as buttons
MUST manage list of the available stories (even if as a introductory story)
MUST manage to associate a faction to the return answer to enable the retrieval of the proper translation based on the ID of the answer
HAS to be accessible through API
The description of the user interaction with the BOT SHOULD be saved in a database

Software state of the art

The updated list of chatbot services that the project will evaluate is listed in the table below summarizing some of the main characteristic of the product.

Product	Licence	Web site	Release
RASA	Apache License, Version 2.0	https://rasa.com/	1.4.5
Chatterbot	BSD 3-Clause	https://github.com/gunthercox/ChatterBot	1.0.5
Botpress	GNU Affero General Public License v3.0	https://botpress.io	12.2.3

Evaluation

Evaluation element	RASA	Chatterbot	Botpress
Functional completeness	2	1	1
Functional correctness	2	2	2
Functional appropriateness	2	2	2
Resource Utilisation	2	2	2
Co-existence	2	2	2
Interoperability	2	1	2
Learnability	1	1	1

Operability	2	1	1
Maturity	2	1	2
Confidentiality	2	2	2
Integrity	2	2	2
Modularity	2	2	2
Analysability	2	2	2
Replaceability	2	2	2
Totals	**27**	**23**	**25**

Project choice

Since there is already a good experience in one developing partners in the RASA technology with extensions in managing stories within the database, this will be the choice for the MICADO platform.

3.5. Database

A database store data in a persistent way allowing it to be searched and retrieved by applications.

The way data is stored affect how it will be retrieved in the future.

Usage in MICADO

In MICADO use case there is the will to manage information linked to the spatial information and this require the storage engine to be able to store and analyse data in relation to the concept of spatial concept.

An example of use case could be “get all migrants of a certain nationality and tell me where are living in the city” being able to see that information plotted on a map or “tell to this user the providers offering language courses nearest to his house”.

Technical requirements

MICADO need a relational database with good spatial engine.

The engine must also be able to crypt data so that in case of data breach the content is safe from unauthorized access.

Technical requirements
MUST offer spatial representation of data
MUST offer spatial queries

MUST manage encryption at engine level

Software state of the art

The two main open source database engines are MySQL and PostgreSQL. Both widely used in enterprise environment, with strong community and support.

Both are complete solutions with proven adoption. On the spatial component PostgreSQL, with its PostGIS extension, has e best leverage on performance and completeness.

Product	Licence	Web site	Release
MySQL	GPL	https://www.mysql.com/	8.0.18
PostgreSQL	Postgres Licence	https://www.postgresql.org/	12.1

Evaluation

Deciding between the two engines is a matter of compatibility matrix with the other middleware components of the MICADO's architecture.

RDBMS	PostgreSQL	MySQL
Governance	It is free and open-source and has been released under the PostgreSQL license, similar to MIT license	Even though the source code of MySQL is available, Oracle Corporation offers certain paid versions for commercial use
SQL Compliance	PostgreSQL is largely SQL compliant and meets nearly all core features of the SQL standard	MySQL is partially SQL compliant and does not implement the full SQL standard
Supported Platforms	Solaris, Windows OS, Linux, OS X, Unix-OS and Hp-UX OS	Solaris, Windows OS, Linux, OS X, and FreeBSD OS
Programming Languages Support	C/C++, Java, .Net, R, Perl, Python, JavaScript, and others	C/C++, Erlang, PHP, Lisp, Go, Perl and others
Security	PostgreSQL offers native SLL support for connections for encryptions	A lot of security features are built in MySQL and it is highly secure
Access methods	Support all standards	Support all standards

Replication	PostgreSQL can perform master-slave replication and other types of implementation can be put into practice using third-party extensions	MySQL uses master-master replication and can perform master-slave replication
Performance	Widely used in large systems where read and write speeds are crucial and require execution of complex queries	Widely chosen for web-based projects that require a database simply for data transactions
Community support	A very strong and active community that works to improve existing features and cement its place as the most advanced database	A large community which focuses on maintaining the existing features while new features are released seldom

Evaluation element	MySQL	PostgreSQL
Functional completeness	1	2
Functional correctness	2	2
Functional appropriateness	2	2
Resource Utilisation	2	2
Co-existence	2	2
Interoperability	2	2
Learnability	2	2
Operability	2	2
Maturity	2	2
Confidentiality	2	2
Integrity	2	2
Modularity	2	2
Analysability	2	2
Replaceability	2	2
Totals	**27**	**28**

Project choice

Due to the better spatial features PostgreSQL, with the PostGIS extension, will be the database engine used in MICADO.

Swappability

The role of a database is to store data and there are many, including commercial ones, options that could represent an alternative to PostgreSQL. The modern development is oriented towards a database independence and the MICADO platform does not use, at the state of the design, specific features available only in PostgreSQL; therefore, this is a swappable component.

3.6. Data Integration

Data integration tools aim is to empower the user to work with heterogeneous data sources without having to move data around.

It is often not possible or meaningful to duplicate data from one system to another, especially when this operation is done within the datacentre of the data owner. Duplication of data has the problem of stale data, of having to update it and to have to manage ETL processes.

Enterprise information integration (EII) solutions operate with direct access to the data source, whether it be a relational database or another type of data storage and virtualize it presenting to the end user a relational view of the data allowing to create new relations between entities.

Usage in MICADO

MICADO has the goal to be able to integrate with the system of the operating PA without representing a duplication experience.

This tool will be placed on the edge of the platform to be able to connect to the data sources and mapping the existing data towards the precompiled data sources designed by MICADO.

This will allow ingestion of data in the system with ease.

Technical requirements

Technical requirements
MUST be able to create virtual databases accessing an heterogeneous sources of data

Software state of the art

The two main players examined in this field are Dremio end TEIID.

The first is a more productized solution with a web interface, the second is a solid and proven solution by RedHat that now is migrating as an integrated service inside OpenShift platform.

Product	Licence	Web site	Release
Dremio	Apache License, Version 2.0	https://www.dremio.com/	4.0.5
TEIID	Apache License, Version 2.0	https://teiid.io/	12.3.1

Both solutions allow the user to connect the product to a data source whether it be a relational database, a spreadsheet a CSV file or an API, expose it to a virtual environment where additional virtual resources with join on coherent data can be done.

If the TEIID product is historically more mature is migrating towards a deployment proposition linked to a specific environment proposed by RedHat that is not immediately compatible with the MICADO deployment strategy and the end user tool for working with data sources is a client based on the eclipse software that is going out of commission.

The second is a relatively new player in the field that could be promising as a web-based option.

Evaluation

Evaluation element	Dremio	TEIID
Functional completeness	2	1
Functional correctness	2	2
Functional appropriateness	2	2
Resource Utilisation	2	2
Co-existence	2	2
Interoperability	2	2
Learnability	2	1
Operability	2	1
Maturity	2	1
Confidentiality	2	2
Integrity	2	2
Modularity	2	2
Analysability	2	2
Replaceability	2	2
Totals	**28**	**24**

Project choice

The project choice is still to be made together with the involved teams after a more in-depth analysis on the two tools.

3.7. Dashboard

Dashboarding software can be divided in two different categories:

- platforms that sit on top of data lake and offer a set of features to transform data, aggregate it and present in diagrams and charts, managing also users and permissions
- single libraries that allow developers to design single charts over data retrieved from a backend; these solutions are mainly client-side solutions targeted to development.

Usage in MICADO

Since MICADO aim is not to develop hard coded diagrams, but to empower the end user to use a user-friendly tool to create his own business insight the category that will be examined are platforms for generating dashboarding.

This solution represents a compromise between the power of custom development and the approach of integration that the project decided to follow.

Technical requirements

Dashboards could be used in different use cases in the three frontend applications of MICADO; this calls for a wide set of integration requirements.

Technical requirements
Dashboarding platforms MUST allow the exposure of a single diagram or chart or dashboard as an API call or embedding, since the final experience that the project wants to offer to the end user, on some frontend, is a unified interface detached from the dashboarding platform.
Dashboarding platforms MUST have a customizable UI
Dashboarding platforms MUST have a custom authentication option to be able to integrate authentication within the overall MICADO system; this SHOULD include a single sign on feature to allow PA users to shift from MICADO application to dashboarding design with a uniform user experience
Dashboarding platforms MUST be able to produce diagrams and chart plotted on maps
Dashboarding platforms MUST be able to generate timelines.

Software state of the art

There are some open source platforms that offer the capability to offer the require features; the most used are Metabase and Superset.

Superset is a project built by AirB&B and donated to the Apache foundation, actually in the incubator state (that means the process of adoption in the community of the Apache ecosystem and not the level of stability of the product).

Metabase is an open source business intelligence tool. It lets you ask questions about your data, and displays answers in formats that make sense, whether that's a bar graph or a detailed table.

Product	Licence	Web site	Release
Superset	Apache License, Version 2.0	https://superset.incubator.apache.org/	
Metabase	GNU Affero General Public Licence V3	https://github.com/gunthercox/ChatterBot	

Evaluation

Both products are equivalent and deeper analysis will have to be conducted by the project team on the topics of embedding and authentication to validate MICADO use cases.

Data Sources	Metabase	Superset
Amazon Redshift	✓	✓
Google BigQuery	✓	✓
Cassandra		
MongoDB	✓	
PostgreSQL	✓	✓
MySQL	✓	✓
Google Analytics	✓	
Snowflake	✓	
Druid	✓	✓
H2	✓	
SQLite	✓	✓
Microsoft SQL Server	✓	✓
CrateDB	✓	
Oracle	✓	✓

Vertica	✓	✓
Presto	✓	

Authentication Tool	Google OAuth	LDAP	OpenID	Database
Metabase	Present	Present	Absent	Absent
Superset	Present	Present	Present	Present

Evaluation element	Superset	Metabase
Functional completeness	2	1
Functional correctness	2	2
Functional appropriateness	2	2
Resource Utilisation	2	2
Co-existence	2	2
Interoperability	2	1
Learnability	1	1
Operability	2	2
Maturity	2	2
Confidentiality	2	2
Integrity	2	2
Modularity	2	2
Analysability	2	2
Replaceability	2	2
Totals	**27**	**26**

Project choice

Also in this area the decision over which of the two tools adopt in the MICADO architecture is not yet consolidated. A further investigation and cross referencing with the result of the co-

creation workshop and the dashboarding needs will be made to identify the best solution for the project.

Swappability

The two softwares examined represent the open source option of a wide range of tools. The integration of these tool has been designed with the goal of embedding dashboards implemented in this software. Therefore, this is a swappable component provided that the alternative offers an embedding solution.