Characteristics of the technological architecture of the SISCAT information systems master plan

List of characteristics, quality attributes and non-functional requirements

March, 2022

Generalitat de Catalunya
Departament de Salut

Sistema de
Salut de Catalunya

Direction or coordination: Josep Antoni Mira

Authors or editors: Josep Antoni Mira, Juan Manuel Miguel, David Villacé, Miguel Àngel Lanau i Juan Carlos Cornejo

Edita:

Office of the Director Plan of Information Systems of SISCAT. Catalan Health Service.

1st edition:

Barcelona, March 2022.

Assessorament lingüístic:

Servei de Planificació Lingüística del Departament de Salut (Department of Health Language Planning Service)

Disseny de plantilla accessible 1.05:
Communication Office. Corporate identity.

Sumari

1 Introducció

The purpose of this document is to provide a descriptive framework for the technological architecture that must be shared by the products and services developed within the scope of the Information Systems Master Plan of the Catalan Integrated Public Health System (SISCAT). The definitions, operational definitions and requirements presented in this document must be applied to all components, services and applications that are integrated within the framework of the Master Plan.

Thus, the characteristics described in this document make up the non-functional requirements, also known as quality attributes, that any component of the architecture of the products or services developed within the ecosystem of the SISCAT Information Systems Master Plan must meet.

The document is organised into three broad areas that describe the characteristics to be fulfilled according to the environment to which they refer. Specifically, the three main sections covered here are as follows:

1) Operational characteristics: these are those that describe how the technological architecture must behave once it is deployed and in operation. They mark how it should operate and function.

2) Structural characteristics: these describe what the code serving the applications must be like and what internal quality characteristics it must meet.

3) Transversal characteristics: these are important restrictions and criteria in the definition of the technological architecture that cannot be included in the previous sections.

This document is part of the reference architectures of the Electronic Health Record (HES).

The list of characteristics is based on the RIC20 collection.

2 Characteristics of the architecture tecnològica

2.1 Rules of precedence of the characteristics

In the case of a collision of two or more characteristics, those that are of an operational type take precedence over the others, and those that are structural over those that are transversal. In the event of a conflict between characteristics of the same class, those that are more relevant take precedence according to that indicated in section 2.2.

In any case, the professional architect responsible for the development can establish rules adapted to the context if he or she justifies them adequately.

2.2 Summary of the most relevant characteristics

Table 1. Operational characteristics

Operational characteristic	Llindars
Availability	24x7 with 99,95 % availability
Continuïtat	RTO < 1 h RPO = 0 Recovery of data at one point in time: 48 h
Rendiment	Gestió de la capacitat: ● In PaaS, IaaS, Hosting type installations: ○ RAM: maximum occupancy rate of 65 %. ○ CPU: maximum permissible occupancy of 65 %. ○ disc: maximum allowable occupancy of 65 %. ● In container-based installations: ○ RAM: maximum occupancy rate of 85 %. ○ CPU: maximum permissible occupancy of 85 %. ○ disc: maximum occupancy rate of 85 %. Response time: It is understood that the response time is from the invocation of the business layer (application programming interface (API) or event publication).

Operational characteristic	Llindars
	□ Single recordings: maximum accepted 60 ms (milliseconds), within a 95th percentile. □ Query operations of a specific record per key: maximum accepted of 40 ms, within a 95 percentile.
Recuperabilitat	Less than two hours in systems that are not in high availability multiCPD 30 s in case of systems with high availablity between centres de processaments de dades (CPD)
Reliability	High availability in all components and services
Robustesa	Preferably, communication between systems should be based on the following events
Scalability	All components must be automatically scalable both vertically and horizontally.

Table 2. Structural characteristics

Structural characteristic	Llindars
Extensibilitat	Within the same functional domain by means of code versions. Between functional domains depending on the event and if not possible using a REST API.
Instal-labilitat	Automatically via SIC on all layers.
Internationalisation or localisation	Officials in Catalonia
Portabilitat	Applications and components prepared to be deployed in the infrastructures of the main public networks with little effort. Preference for solutions as a service (SaaS). Includes both business components and data repositories.
Support	All components must ensure 24x7 business support, with a response time of less than one hour in the event of a critical incident.
Facilitat d'actualització / Management of obsolescence	Compatible versions in any case, preferably those that are de suport a llarg termini (LTS). Update of the annual major version in July. Update of stickers at the beginning of the year.

Structural characteristic	Llindars
	Updating of critical or safety stickers when declared by the manufacturer. Updating only by automatic means - scripts, *playbooks*, etc. Manual actions following a written procedure are not allowed. The procedures d'actualització hauran de preveure la marxa enrere en cas que falli.
Testabilitat	All the components must be able to be tested automatically in all the applicable test variants.
Deployability	All components must be able to be deployed automatically, including databases and client components that may be required at workstations. Deployment procedures must provide for a fallback in case of failure.
Transaccionalitat	Use the SAGA protocol for distributed transactions, and ACID when necessary.

Table 3. Cross-cutting characteristics

Transversal characteristic	Llindars
Accessibility	The user fronts must comply with the legal requirements (WCAG-AA).
Monitoratge	The components must be monitored by the standard monitoring tools provided by the control centre of the Centre de Telecomunicacions i Tecnologies de la Informació (CTTI).
Security	Authentication: By means of the established management tool, preferably by GICAR. Authorisation: based on attributes, to which permissions are assigned and allocated to users and components, with preference given to the authorisation system. of GICAR.
Usability	Norman's rules [NOR13] must be applied.

2.3 Characteristics operational

2.3.1 Continuïtat

Refers to the ability to recover from a technological disaster. Security and restoration keys: recovery of data at a point in time.

- Objective recovery time (RTO): must be less than one hour.
- Objective recovery point (ORP). The status of the set of data determined at a specific moment in time that consequently entails the concept of the set of data that cannot be recovered. Using this last concept as a basis for definition, the value must be 0. In addition, it should be noted that as a set of the data considered, the data from any of the repositories must be taken into account, not only the repository directly related to the specific components from which the need for recovery arises.

- It must be possible to retrieve the data at a certain point in time during the following 48 hours. And, most importantly, the following must be taken into account:

    - The Oracle *archive log.*

    - Els *oplog* de MongoDB (mitjançant Ops Manager).

2.3.2 Availability

Defines how much time the system has to be available. It is considered that it is necessary to achieve high availability at all levels, 24x7, 365 days a year. 99.95% availability.

The components of the SISCAT Information Systems Master Plan solutions must be designed to allow more than one instance to run at the same time, concurrently accessing its data model.

2.3.3 Scalability

Capacity of the system to work with the expected performance even if the number of users changes, increases or decreases.

Preferably, business applications will be deployed on container technologies that allow automatic scaling policies to be defined and implemented. Container orchestration systems will have to manage their capacity in the same way as indicated in the performance characteristic, in the sense of acting in compliance with pre-established conditions with a set of boundary values.

The components shall have automatic scaling capabilities, or hot scaling without service height.

In database systems, and others, that do not allow automatic scaling capability, the necessary automatisms must be enabled to be able to carry out scaling actions without operator intervention.

The systems that cannot be automated manage their capacity in the same way as indicated in the yield characteristic, in order to avoid falls due to resource depletion (vegetative scaling, capacity plan).

Note: in the case of horizontal scaling of databases, a distinction can be made between scaling in reads and scaling in writes. The former can be automated and carried out while warm, the latter implies a distribution of resources in a new set of main-secondaries that may require a more extensive maintenance layer.

2.3.4 Reliability

The system must be fail-safe. All components and services must be duplicated in order to have high availability both locally and between two DPCs in remote locations.

2.3.5 Recuperabilitat

This section is related to the continuity of the business. It indicates the threshold values at which the application should be up and running and providing service once again.

We have two options for recovering the service: the disaster recovery plan (PRD) or high availability between CPD.

- If we are in PRD mode, in the event of the failure of a DPC, the application must be operational within two hours in the secondary DPC until there is high availability between the two available DPCs.
- If we are in multiCPD high availability mode, services should be restored in 30 seconds.

In the event of a crash of a component of an application, the clustering and high availability mechanisms must allow the application to continue within 30 seconds.

2.3.6 Rendiment

This characteristic includes performance tests, spike analysis, analysis of the frequency of the functions used, the capacity required and the response time.

- By capacity we mean various aspects required: RAM (GB), HD (GB), CPU (cores, *millicores*), input and output operations per second (IOPS).
- In case of anomalies or failures derived from an absolute use of the associated capacity, it will have to be increased in the most automatic way possible when:
    - In installations of the types Platform as a service (*PaaS*), Infrastructure as a service (*IaaS*), *Hosting*:
        - RAM: maximum occupancy rate of 65 %.
        - CPU: maximum permissible occupancy of 65 %.
        - HD: maximum occupancy rate of 65%.
    - In container-based installations:
        - RAM: maximum occupancy rate of 85 %.
        - CPU: maximum permissible occupancy of 85 %.
        - HD: maximum occupancy rate of 85%.
- Response time: It is understood that the response time is from the invocation of the business layer (API or event publication).
    - Single recordings: maximum accepted 60 ms (milliseconds), within a 95th percentile - 95% of the samples are kept within this limit.
    - Query operations of a specific record per key: maximum accepted of 40 ms, within a 95 percentile.

To calculate the expected duration of an interaction, the composition of different single accesses will give the duration of an interaction from the outermost layer, the front-end of a given solution. As an example, it is possible to calculate the estimated duration of an appointment reservation to a doctor as the aggregation of the services: to close a patient (40 ms) + to close a doctor (40 ms) + to close a consultation office (40 ms) + to register a new appointment (60 ms) + to prepare the presentation as a response to the end user (60 ms) would give a total estimated response time of 240 ms.

2.3.7 Robustesa

The system must be able to handle errors and limit conditions while it is running, even if the error is due to an external cause such as network connectivity, power failure or machine failure.

The components have to be linked to each other in the form of events, in case one of them does not work, it does not affect the functioning.

In the event that they cannot be called for events, the unavailability of the applications will remain in a queue that will be used to retry them. The user must receive a warning that the operation will be dealt with in deferment.

In the event of not being able to use an attempt basket, the applications must have a contingency plan or alternative route to carry out the action.

2.3.8 Observabilitat

The system must be able to respond to any questions about the functioning of applications or resources at any time regardless of the complexity.

The system and applications have to be instrumented in order to be able to collect metrics, data, volumes, etc., and send them to a system capable of computing and analysing this information in order to obtain the complete compression of the system.

Business metrics monitor the strategic *endpoint* of the system in order to obtain input or processing time metrics.

The specific tools currently provided are Prometheus, together with Grafana, which will have to be expanded with more tools to obtain all the information.

All the developments carried out are not considered complete until they have incorporated an analysis and instrumentation that is compatible with their observability.

- Infrastructure monitoring.

- Monitoring of real users.
- Record display.
- Visualisation of the situation of distributed transactions.

2.4 Característiques estructurals

2.4.1 Aprofitament o reutilització

Ability to take advantage of components common to multiple solutions.

All the solutions developed must be designed with the highest possible degree of abstraction in order to facilitate their reuse and to be quickly detected as usable elements.

A reusable component is considered to be transversal and, therefore, it must have a dedicated infrastructure to protect its resilience and that of the components that cradle it.

2.4.2 Configurability

Ability of users to change aspects of the software configuration (by means of usable interfaces).

No such capacities are foreseen in this version.

2.4.3 Deployment automatic

All the fish and components of the architecture must be deployed automatically. Solutions that have definitions based on relational database type components must be managed by suitable DevOps tools (Liquibase, Redgate, Flyway, DBmaestro, ...). In the same way, it must be possible to deploy components in the user's workstation. The automatic deployments will have to have mechanisms to carry out the backward movement automatically as well.

2.4.4 Entorns

The areas of execution of the products and services of the SISCAT Information Systems Master Plan are DES, INT, PRE and PRO.

DES has to be provided by the development supplier, it will have to respect the products and components and their versions, offered by CTTI in the rest of the environments.

INT must be managed by CTTI and determines the products and components and their versions in the PRO environment, although the architecture may be simplified in the duplication of components or other PRO-specific features.

PRE must be managed by CTTI and must replicate exactly the architecture proposed in PRO except for its size, which must be at least 70% of that of PRO.

PRO must be managed by CTTI and all the characteristics mentioned in this document must be applied.

2.4.5 Extensibilitat

Ability to add new items to the application with the minimum impact on the current service.

- Within the same functional domain, new versions of the code are established.
- Between functional domains, preferably via asynchronous events. If this is not possible, by means of REST API or GraphQL API calls, depending on the appropriate interlocution of the functional domains.

In any case, extensibility must be based on direct connections to databases.

2.4.6 Ease of upgrading or managing obsolescence

Facility to update easily and quickly from a previous version to a current version for both server and client programming.

All the elements that make up the architecture of the products and services of the SISCAT Information Systems Master Plan must have the support of the corresponding manufacturer or community.

Long-term product versions (LTS) should preferably be chosen whenever available.

Two annual periods are established for updating the products: in January and July. Bearing in mind that the products follow the semantic version (major, minor, sticky), in July it is updated to the major version, and in July to the most recent minor version within the installed major version. Whenever there are new developments since the last update.

In the event of a high safety risk or malfunction, the version of the sticker recommended by the manufacturer or the community maintaining the product must be applied immediately.

All the fish must have the update scripts that allow them to be automatically updated. In the case of generated source code, it must be deployed exclusively through the Continuous Integration Service (SIC). In the case of products, they must be packaged in automation managers such as Helm, Ansible, Pulumi or whatever is offered by the manufacturer or community. It has to be evaluated if it is appropriate that, in the case of generating the binary, it is packaged in the binary repository service of the CTTI based on Harbor. All update scripts have support for rollback in case of failure.

2.4.7 Instal-labilitat

Ability to install a solution in the simplest possible way. The solutions of the SISCAT Information Systems Master Plan (its components) are deployed by means of the SIC or by means of an automation script, completely unattended execution, to install it as well as its updates. The fact that a solution, development or product cannot be installed automatically is a reason for rejection and it will be necessary to evaluate acceptable alternatives that meet this condition.

2.4.8 Internationalisation

Support for being able to represent literals in different languages within the same application. As a minimum, the official languages in Catalonia are established, although other languages may be enabled according to functional criteria.

2.4.9 Interoperability

Ability of the system to integrate with the rest of the elements of its functional context in an easy and simple way. The criteria are very similar to those of the extensibility feature. The solutions must make it possible to integrate third parts, beyond their own components. In order to make this possible in an easy and simple way, they must expose an interface based on

For REST, the specifications of [API16] should be followed for APIs, and for GraphQL, those of [GRA21]. For both ways, it will be necessary to define the contracts associated with the data structures exchanged in the requests and responses, preferably based on health care standards.

2.4.10 Location

Support for multiple languages in data entry and display screens, in reports and for measuring units or currencies. As a minimum, the official localisation criteria are foreseen, although others may be enabled according to functional criteria.

2.4.11 Maintainability

Capacity to be able to apply changes in the evolution of the system easily. The design of the solution must have the maximum degree of disagreement between its components so that, when applying an evolution or correction to them, it is easily identifiable on which of the components it has to be applied and that it only needs to be applied to one of these components. If it has to be changed to different components, it is an indicator of low cohesion and high cohesion and, therefore, of high technical debt.

2.4.12 Portabilitat

It consists of the ability to implement a solution on different platforms with the same technological base.

Given the current trend in information systems, these must be able to be run on cloud platforms. All products and applications to be installed in the context of HES must be ready to be migrated to the cloud with little effort.

The descriptors that make it possible to deploy and obtain resources within the scope of the cloud platform must be based on the Kubernetes orchestrator.

Databases must be used that have a way of being brought to the cloud with little effort and with mature migration procedures.

2.4.13 Support

Type or level of support required for the solution.

All components must have 24x7 support. As far as possible, support from the manufacturer must be contracted if the component is based on a product with a response time of less than one hour, in the event of critical incidents or loss of operation. If there is no manufacturer and there is support from a free software community behind the product, developers must have permissions to make changes to the code repository or participate as a contributor to the code repository. *Forks* can be made to the original repository only in case of emergency and on a temporary basis. The content of the pasting carried out must revert to the community by means of the corresponding *pull request* to the original repository.

2.4.14 Testabilitat

The components of a solution must be tested automatically: unit tests, functional tests, performance tests, exploratory test records and any other type of test that may arise.

2.4.15 Transaccionalitat distributed

Apply the SAGA [SAG21] pattern when distributed transactions involving different micro-services are considered. If ACID transactions are required, use the databases that have this facility.

2.5 Characteristics transversals

2.5.1 Accessibility

Access to all users, without distinction. By legal requirement, user interfaces have the double A of the WCAG standard.

2.5.2 Arxivament

History-keeping or deletion of inactive data. To maintain operational database systems at maximum efficiency, solutions must be designed to allow the archiving of a subset of their data without compromising the overall functionality of the solution itself. Archiving makes use of long term and low cost storage options.

2.5.3 Compliment legal

The solutions are designed to comply with current legislative restrictions on data protection and others that may apply.

2.5.4 Integrity

Data integrity is the maintenance and assurance of data accuracy and consistency throughout the data lifecycle and is a critical aspect for the design, implementation and use of any system that emulates, processes or retrieves data.

Systems that store data must have mechanisms to prevent corruption both at the time of writing and at the time of retrieval, using redundant machinery and rules for verifying the information retrieved.

2.5.5 Monitoratge

Ability to monitor the status of the components, establish links to integrate alerts and warnings with the current system of the other components. The components are monitored by the standard monitoring tools provided by the CTTI control centre.

2.5.6 Privacitat

Ability to withhold information from any actor who does not have the appropriate authority to access or manage this information. This information includes information related to and involved in the communications and own traces (*logs*) of the components of the solution.

Data related to users must be encrypted when they are stored. Encryption must be done at the logical level (database engine) and at the physical level (disk level).

All communications with and between services must be encrypted. The

encryption keys must be rotated periodically, every 180 days.

2.5.7 Security

Authentication: security requirements to ensure controlled access by any type of actor, such as end users or other external components of the solution.

The architecture defines an authentication management tool that allows authentication with different systems, among them GICAR - see the reference to architecture

Keycloak. Solutions must adopt integration with this management tool and base their particular security policy on it.

This authentication management tool centralises a unique control in order to verify which actor can access or not to interact with the components of the solution.

The security standards to be applied are OpenID Connect and SAML. If other standards are proposed, the proposals must be evaluated with the arguments presented.

Authorisation: security requirements to ensure that the actors have the authorisation to operate with the solution. To simplify it, authorisation is based on attributes, and each actor has to adhere to one or more attributes according to the characteristics of the functions to be performed.

Incident response: the systems concerned must have the capacity to respond to a security incident in less than six hours. This includes the ability to bring any product sticker or application into production automatically.

It must comply with the current security standards published by the Cybersecurity Agency.

2.5.8 Traçabilitat

All changes made to the platform must leave the corresponding trace in the change management system.

Changes to the source code must be reflected in the corresponding *release notes* in the Git project itself.

The changes in the modules keep the traceability with respect to the functional requirement or the characteristic of the architecture that they intend to cover, effectively documented in the life cycle management and quality management plan in force.

2.5.9 Usability

Ease of use.

S'han d'aplicar les deu regles de Norman [NOR13]:

- Visibility of the status of the system
- Relationship between the system and the real world
- Control and freedom of the user
- Consistency and standards
- Error prevention
- Recognise rather than remember
- Flexibility and efficiency of use
- Aesthetic and minimalist design
- Help users to recognise, diagnose and correct errors.
- Availability of aid and documentation

3 Referències bibliogràfiques

[API16] RESTful APIs: best practices [Internet]. [cited on 7 March 2022]. Available at: https://canigo.ctti.gencat.cat/blog/2016/01/api/

[GRA21] GraphQL [Internet]. [cited on 7 March 2022]. Available at: https://spec.graphql.org/October2021/

[NOR13] NORMAN, D. *The design of everyday things*. Edició revisada i ampliada. New York: Basic Books, 2013. 347 p.

[SAG21] Microservices Pattern: Sagas [Internet]. microservices.io. [cited 19 November 2021]. Available at: http://microservices.io/patterns/data/saga.html

[RIC20] RICHARDS, M.; FORD, N. *Fundamentals of software architecture: an engineering approach*. First edition. Beijing Boston Farnham Sebastopol Tokyo: O'Reilly, 2020. 400 p.