UNIVERSIDAD DE LA REPÚBLICA URUGUAY

FACULTAD DE INGENIERÍA

DINABANG-CDA

Informe de Proyecto de Grado presentado por

Gonzalo Capote, Lucía Sosa y Carolina Trías

en cumplimiento parcial de los requerimientos para la graduación de la carrera de Ingeniería en Computación de Facultad de Ingeniería de la Universidad de la República

Supervisores

Franco Simini
Antonio López Arredondo

Montevideo, 16 de noviembre de 2024

Agradecimientos

Agradecer profundamente a todas las personas e instituciones que hicieron posible el desarrollo de este proyecto. Entre ellos queremos mencionar especialmente a nuestros tutores Franco Simini y Antonio López, así como Darío Santos y todo el equipo del Núcleo de Ingeniería Biomédica por el apoyo continuo, dedicación y compromiso. También mencionar y agradecer a Jorge Domínguez y Rodrigo Barboza de la empresa MOVI por su excelente disposición y compromiso. A Lucia Grundel, Gastón Milano, Juan Tabarez y Sabrina Sellanes por la información aportada de alto valor para que este proyecto alcanzara un mayor nivel de excelencia.

Agradecer también a nuestras familias y amigos, que nos han apoyado y acompañado en este camino desde fuera del aula, haciendo que todo el esfuerzo y dedicación fuera posible.

Resumen

La generalización de la Historia Clínica Electrónica hace que sea necesario conectarla con equipos biomédicos. Los repositorios que centralizan documentos clínicos deben, por lo tanto, incluir también el resultado de estudios, tratamientos o evidencia de las actividades de los pacientes generados por instrumentos médicos en hospitales, consultorios o wearables. En Uruguay, estos repositorios están previstos únicamente para los mayores prestadores de salud, dejando sin conexión a los profesionales de la salud en la libre profesión, centros de rehabilitación y consultorios ajenos a las instituciones grandes.

Para realizar la conexión entre un equipo biomédico cualquiera y la Historia Clínica Electrónica (HCE) se proyecta e implementa DOCLUY. **DOCLUY es una plataforma de interoperabilidad que facilita la conexión de equipos biomédicos y software con repositorios de documentos** clínicos, incluyendo la Historia Clínica Electrónica Nacional de Uruguay (HCEN). DOCLUY permite enviar información clínica en diferentes formatos estándar de informática médica. Una de las ventajas de DOCLUY es que no solo está diseñada para grandes prestadores de salud, sino que también es accesible para profesionales de la salud independientes, centros de rehabilitación y consultorios en los que el único requisito es la disponibilidad de internet. Además, DOCLUY integra múltiples repositorios, lo que significa que cualquier dispositivo biomédico o sistema médico puede enviar información clínica a diferentes repositorios de manera simple y segura, independientemente del formato utilizado.

DOCLUY recibe la información clínica a través de un servicio web, genera y envía un documento clínico en el formato requerido (por ejemplo, CDA: Clinical Document Architecture) al repositorio destino. **Esto hace que los generadores de información clínica no tengan que preocuparse por formatos, estándares ni políticas de integración de cada uno de los repositorios clínicos**. Además, el envío de esta información se hace de forma asíncrona, lo que delega en DOCLUY la responsabilidad de asegurarse de la entrega del documento una vez recibida la información, liberando de ello al equipo biomédico que generó el informe destinado a la HCEN de un paciente en particular. Esta solución fue implementada con tecnología de Amazon Web Services, que permitió la construcción de una **arquitectura sin servidor, basada en colas de mensajes, procesamiento asíncrono, recuperación frente a fallos de conectividad con los repositorios y trazabilidad**.

Habiendo cumplido las etapas de experimentación y testing, **DOCLUY fue**

conectado a dos generadores de información clínica y un repositorio de documentos clínicos. Se logró la integración exitosa del dispositivo clínico DINABANG y de la aplicación Sistema Informático de Enfermería (SISENF), ambos generadores de información clínica, uno como equipo de medida de la potencia de miembros inferiores, el otro como sistema de información en consultorio y hospitalización. Se logró, con el repositorio de documentos clínicos HCEN mediante un servidor local que emula su entorno productivo. Ambas conexiones permitieron validar la implementación de DOCLUY realizada, con el envío exitoso de información clínica en formatos CDA nivel 1, que encapsula PDF y metadatos, desde dispositivos médicos a un repositorio de documentos clínico.

Índice general

1. Introducción **1**

2. Revisión de antecedentes **3**
2.1. Estándares HL7 3
2.1.1. HL7 versión 3 3
2.1.2. Formato de documento CDA R2 4
2.1.3. Estándar de interoperabilidad HL7 FHIR 5
2.2. Historia Clínica Electrónica Nacional 6
2.3. Concepto de informática sin servidor 8
2.4. Ley de protección de datos personales 9
2.5. Proyectos relacionados 10
2.5.1. Medida de la potencia muscular DINABANG 10
2.5.2. Sistema informático de enfermería SISENF 11
2.5.3. Sistema informático de patología oncológica músculo esquelética SIPOME 11
2.5.4. Agenda de vacunación COVID-19 12
2.6. Productos existentes de interoperabilidad 12
2.7. Conclusiones del capítulo 14

3. DOCLUY, solución general que satisface DINABANG-CDA **15**
3.1. Requerimientos 15
3.1.1. Requerimientos funcionales 15
3.1.2. Requerimientos no funcionales 17
3.2. Arquitectura propuesta 18
3.2.1. Decisiones Principales 20
3.2.2. Componentes de DOCLUY 23
3.2.3. Procesos principales 26
3.2.4. Manejo de errores y reintentos 30
3.2.5. Monitoreo y observabilidad 31
3.2.6. Trazabilidad de documentos 31
3.2.7. Seguridad 32
3.2.8. Mutilenguaje 35
3.2.9. Conclusiones 35
3.3. Implementación de DINABANG-CDA 35

4. Pruebas y Simulaciones **37**
4.1. Configuración del entorno de pruebas . . . 37
4.2. Automatización de pruebas . . . 38
4.3. Casos de prueba . . . 39
4.3.1. Pruebas funcionales . . . 39
4.3.2. Pruebas de integración . . . 41
4.3.3. Pruebas de autenticación y autorización . . . 41
4.3.4. Conclusiones . . . 41

5. Conclusiones y trabajo futuro **43**
5.1. Gestión del proyecto . . . 43
5.2. Dificultades encontradas . . . 51
5.3. Autocrítica . . . 53
5.3.1. Definición del ambiente de trabajo . . . 53
5.3.2. Períodos de baja dedicación . . . 53
5.3.3. Intentos con Prestadores . . . 53
5.4. Conclusiones . . . 53
5.5. Futuro del proyecto . . . 54

Referencias **57**

Glosario **63**

Anexos **65**

A. Comunicación entre múltiples sistemas **67**
A.1. Comunicación directa . . . 67
A.2. Middleware . . . 68

B. Tipos de infraestructura **71**
B.1. Infraestructura en la nube . . . 71
B.2. Infraestructura propia . . . 72
B.3. Infraestructura de los repositorios . . . 72

C. Procesamiento asincrónico **75**
C.1. Procesamiento en lote . . . 75
C.2. Colas de mensajes . . . 76
C.3. Eventos . . . 76

D. Proveedores de infraestructura **79**
D.1. Amazon Web Services (AWS) . . . 79
D.2. Microsoft Azure . . . 80
D.3. Google Cloud Platform (GCP) . . . 80

E. Servicios utilizados de AWS **83**
E.1. API Gateway . . . 83
E.2. Lambda . . . 83
E.3. Amazon Simple Storage Service (S3) . . . 83
E.4. Amazon Elastic Compute Cloud (EC2) . . . 83
E.5. CloudFormation . . . 84
E.6. Step Functions . . . 84
E.7. CloudWatch . . . 84
E.8. DynamoDB . . . 84
E.9. Identity and Access Management (IAM) . . . 84

F. Inversión en infraestructura **85**

G. Tipos de Lenguajes **87**
G.1. JavaScript (node.js) . . . 87
G.2. Python . . . 87
G.3. Java . . . 88
G.4. GO . . . 88
G.5. C# (.NET) . . . 89

H. Tipos bases de datos **91**
H.1. Base de datos relacional (SQL) . . . 91
H.2. Base de datos no relacional (NoSQL) . . . 92
H.3. Solución híbrida (SQL + NoSQL) . . . 92
H.4. Tipos de bases de datos NoSQL . . . 94
H.5. MongoDB . . . 94
H.6. Amazon DynamoDB . . . 94

I. Recepción de información para la generación de documentos **97**
I.1. Recibir toda la información desde el generador de información . . . 97
I.2. Mantener datos pre guardados en el middleware y recibir identificadores . . . 98

J. Detalles del envío a repositorios **101**
J.1. Salud digital . . . 101
J.2. Repositorios FHIR . . . 101
J.3. Métodos custom . . . 101

K. Tabla de estados de documentos **103**

L. Definición de API de creación de documentos **105**

M. Ejemplo de CDA y transacción ITI-41 **111**

N. Aplicación desarrollada para pruebas: Demo DOCLUY **135**

Ñ. Ingeniería de muestra 2022 **139**

O. Integración de sistema SISENF con DOCLUY **141**

P. Integración de dispositivo DINABANG con DOCLUY **143**

P.0.1. Secuencia alternativa donde se intenta enviar un documento que ya fue enviado previamente 151

Índice de figuras

2.1. Estructura del *Clinical Document Architecture* (CDA). 5
2.2. Componentes de la Historia Clínica Electrónica Nacional (HCEN). 7
2.3. DINABANG en el consultorio de fisioterapia. 11

3.1. Diagrama de Arquitectura de la solución en AWS 19
3.2. Estados de procesamiento CDA. 20
3.3. Implementación de API Rest mediante Amazon API Gateway . 23
3.4. Colas de mensajes utilizadas en el flujo general. 24
3.5. Máquina de estados utilizada en el flujo general 25
3.6. Diagrama de medios de almacenamiento 26
3.7. Proceso de recepción y almacenamiento de información clínica . 27
3.8. Máquina de estados de la generación de un archivo CDA 28
3.9. Arquitectura del proceso de envío de documentos a repositorios clínicos 29
3.10. Ejemplo de registro de actividad de un documento. 33

4.1. Entorno de pruebas DOCLUY. 38
4.2. Interfaz de Postman. 38

5.1. Esfuerzo total del equipo a lo largo de los meses. 44
5.2. Esfuerzo total del equipo dedicado a cada actividad. 45

F.1. Inversión en infraestructura de AWS 85

N.1. Demo DOCLUY, página inicial de la aplicación. 136
N.2. Demo DOCLUY, lista de documentos CDA registrados en Repositorio XDS local de una persona. 137
N.3. Demo DOCLUY, visualización de un CDA. 138

Ñ.1. Póster presentado en Ingeniería de Muestra 140

O.1. PDF enviado por SISENF a DOCLUY 142

P.1. Secuencia en DINABANG: Paso 1 144
P.2. Secuencia en DINABANG: Paso 2 145
P.3. Secuencia en DINABANG: Paso 3 146

P.4. Secuencia en DINABANG: Paso 4 147
P.5. Secuencia en DINABANG: Paso 5 148
P.6. Secuencia en DINABANG: Paso 6 149
P.7. Secuencia alternativa en DINABANG: Paso 1 151
P.8. Secuencia alternativa en DINABANG: Paso 2 152

Índice de cuadros

3.1. Comparación de alternativas de comunicación entre múltiples sistemas. 21
3.2. Comparativa de tipos de infraestructura consideradas. 21
3.3. Comparativa de alternativas de procesamiento asíncrono. 22

4.1. Casos de prueba principales ejecutados para validar el correcto funcionamiento de DOCLUY. 40
4.2. Casos de prueba de autenticación y autorización. 41

A.1. Comparación de alternativas de comunicación entre múltiples sistemas. 69

B.1. Comparativa de tipos de infraestructura consideradas. 73

C.1. Comparativa de alternativas de procesamiento asíncrono. 77

D.1. Aspectos considerados en la evaluación de proveedores de infraestructura. 81

G.1. Comparativa de lenguajes de programación considerados. 89

H.1. Tabla comparativa de tipos de bases de datos. 93
H.2. Tabla comparativa de bases de datos no relacionales. 95

I.1. Tabla comparativa de alternativas de recepción de información. . 99

K.1. Tabla de estado de documentos 103

L.1. Información requerida en la API de creación de documentos. . . 109

M.1. Especificación de parámetros definidos en la generación del CDA, la transacción ITI-41 y el mensaje MTOM. 133

Capítulo 1

Introducción

La integración de equipos biomédicos y sistemas de información clínica tiene un alto de grado de madurez en grandes prestadores de salud. Sin embargo, cuando consideramos centros de salud más pequeños, como consultorios de dentistas, clínicas de fisioterapia y otros profesionales independientes, la realidad es que no pueden integrar sus datos clínicos con la Historia Clínica Electrónica Nacional (HCEN)(AGESICl, 2023a). Esta limitación impide que la información generada en dichos establecimientos fluya hacia el sistema centralizado de la HCEN, creando un vacío en la trazabilidad y disponibilidad de los datos de salud de los pacientes.

El dispositivo biomédico DINABANG(R.Barboza, J. Dominguez, A. Fernandez, 2018) es un ejemplo de tecnología que puede generar información clínica valiosa, pero que, hasta ahora, no enviaba información clínica automáticamente a los sistemas nacionales de salud. DINABANG, utilizado en la rehabilitación muscular y deportiva, genera reportes detallados del paciente, los cuales necesitan ser integrados de manera eficiente y segura con la HCEN. Este proyecto pretende cerrar esa brecha, permitiendo que los reportes generados por DINABANG sean enviados a la HCEN en el formato clínico estándar CDA(J. Pascual, 2019).

El presente trabajo tiene como objetivo el análisis, diseño e implementación de una solución de software que actúe como intermediario entre dispositivos biomédicos y repositorios de documentos clínicos para resolver el envío de documentos clínicos en diversos formatos. En particular, se desea enviar reportes clínicos desde el dispositivo biomédico DINABANG. Esta integración permitirá la transferencia automática y segura de los datos clínicos generados por el dispositivo a las historias clínicas, minimizando la intervención humana y, por tanto, los errores asociados.

Para lograrlo se definieron los siguientes objetivos específicos:

- Conocer en profundidad el uso del dispositivo médico DINABANG y participar en prácticas médicas en equipos interdisciplinarios.
- Lograr que los actuales reportes que provee DINABANG se generen tam-

bién en formato CDA.

- Lograr que el dispositivo DINBANG tenga un módulo de integración con Salud Digital que permita interoperabilidad con la HCEN.
- Implementar una solución de software que reciba información de múltiples dispositivos y sistemas médicos y genere documentos clínicos en diversos formatos que puedan ser enviados a distintos tipos de repositorios.

Nuestro proyecto se centra en brindar una herramienta tecnológica para facilitar la interoperabilidad entre los pequeños prestadores de servicios de salud y HCEN, asegurando que cualquier clínica o consultorio pueda generar y enviar sus informes en formato CDA a la HCEN.

DOCLUY, la solución implementada, no solo integra dispositivos biomédicos como DINABANG, sino también se integra con sistemas médicos, como el Sistema Informático de Enfermería SISENF (G. Bonilla, M. Rocanova, 2023), permitiendo el envío automatizado de datos clínicos. Además, la arquitectura diseñada es capaz de soportar otros tipos de formatos médicos además del CDA, lo que abre la puerta a futuras integraciones con otros estándares de interoperabilidad, mejorando aún más la flexibilidad y escalabilidad del sistema.

El presente documento tiene como objetivo exponer el desarrollo del proyecto y los logros alcanzados. Se organiza en 5 capítulos de la siguiente manera.

El capítulo 2 revisa los estándares y tecnologías relevantes al ámbito de la interoperabilidad en salud, así como proyectos relacionados que influyeron en el desarrollo de DOCLUY. También se aborda la situación actual de la HCEN.

El capítulo 3 presenta la arquitectura y diseño de la solución propuesta, describiendo los requerimientos funcionales y no funcionales, así como los componentes técnicos de la solución implementada.

El capítulo 4 describe el entorno de pruebas utilizado para validar la solución, los tipos de pruebas realizadas.

Finalmente, en el capítulo 5 se discuten las conclusiones alcanzadas a partir del desarrollo del proyecto, incluyendo los desafíos enfrentados, las lecciones aprendidas y las posibles líneas de trabajo futuro.

Capítulo 2

Revisión de antecedentes

En este capítulo se exponen los resultados de la revisión bibliográfica y comercial, así como conocimientos necesarios para construir la solución de este proyecto. El objetivo es exponer conceptos importantes para la comprensión del documento y presentar proyectos y productos existentes relacionados con el tema de este trabajo. Comenzamos definiendo los conceptos del ámbito de la salud que dan contexto a este proyecto. Luego presentamos proyectos que influyeron en la solución aportada y finalizamos con la investigación realizada de búsqueda de productos en el mercado.

2.1. Estándares HL7

Health Level Seven International (HL7), fundada en 1987, es una organización sin fines de lucro, acreditada por American National Standards Institute (ANSI) (ANSI, 2023). Se dedica a proporcionar un marco integral y estándares relacionados para el intercambio, la integración y la recuperación de información de salud electrónica que respalda la práctica clínica y la gestión, entrega y evaluación de servicios de salud. (HL7 Internacional, 2023) A continuación describiremos brevemente los estándares y especificaciones de HL7 relevantes para este proyecto.

2.1.1. HL7 versión 3

La versión 3 de HL7 (HL7 V3) es un conjunto de especificaciones basadas en el Modelo de Información de Referencia (HL7, 2024), RIM por sus siglas en inglés (Reference Information Model). (HL7 International, 2023)

HL7 V3 provee un conjunto completo de mensajes, tipos de datos y terminologías necesarias para construir una implementación completa de interoperabilidad de información clínica. Incluye estándares para comunicaciones que documentan y gestionan la atención y tratamiento de pacientes en una amplia variedad de entornos de atención sanitaria. Esto lo convierte en una parte fun-

damental de las tecnologías necesarias para integrar la información sanitaria en distintas áreas de la salud.

2.1.2. Formato de documento CDA R2

HL7 Versión 3 Clinical Document Architecture Release 2 (HL7 V3 CDA R2) es un estándar que define la estructura y la semántica de documentos clínicos, como informes de alta, radiológicos o patológicos, con el propósito de intercambiarlos entre proveedores de atención médica y pacientes. (HL7, 2023b)

Estos documentos pueden incluir texto, imágenes y otros tipos de datos multimedia. Se define como documento clínico aquel que posee las siguientes seis características: (Salud Digital, 2019)

1. Persistencia: Permanecer inalterado y en uso durante un largo periodo de tiempo y de forma independiente a cambios externos.

2. Custodia: Un documento clínico debe ser mantenido por una organización confiable, como un hospital.

3. Capacidad de autenticación: Capacidad de certificación legal de que la información clínica es veraz y auténtica.

4. Contexto: El documento debe definir un contexto predeterminado para el registro y participantes, como la identidad del paciente, quién creó el documento, acto clínico, etc.

5. Completitud: El documento en sí es una unidad de información, de manera que el documento completo, y no solo partes de él, se puede autenticar.

6. Legibilidad humana: Aunque está diseñado para ser procesado por sistemas automáticos, el documento debe ser legible por una persona en un navegador.

El documento CDA se compone de dos partes principales:

CDA Header: Contiene los metadatos del documento, como los datos de los participantes (paciente y médicos), sistema que ha creado el documento, información de seguimiento y auditoría, etc.

CDA Body: Contiene la información que constituye el contenido del documento, y debe contener una parte legible por un ser humano.

Esta información puede estar estructurada de distintas formas, atendiendo a su nivel de estructura. El estándar HL7 CDA clasifica los documentos en tres niveles, según el grado de estructuración y semántica. Los mismos se ilustran en la figura 2.1

En el Nivel 1 el documento es compatible solamente a nivel de cabecera. Un archivo CDA nivel 1 puede contener un documento no estructurado, en cualquier formato propio, como por ejemplo, un documento PDF embebido.

El Nivel 2 contiene datos parcialmente estructurados. Con cabecera estructurada y contenido estructurado en secciones. Estas secciones pueden contener

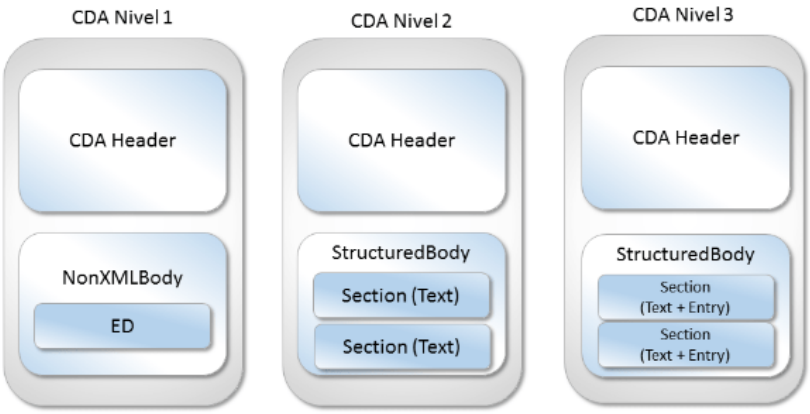


Figura 2.1: Estructura del *Clinical Document Architecture* (CDA). El nivel 1 incluye documentos o imágenes, mientras que el nivel 3 incluye datos estructurados para su procesamiento e interoperabilidad semántica. Tomado de (J. Pascual, 2019)

un título legible y un texto libre sin estructurar, pero la semántica debe estar codificada en un estándar, como LOINC (Regenstrief Institute, 2024) o SNOMED (SNOMED, 2024).

En el Nivel 3 de estructura, el contenido está completamente estructurado y codificado. La cabecera del documento es estructurada, al igual que las secciones, conforme a las estructuras HL7 RIM. Aunque puede haber texto narrativo para garantizar la legibilidad, las entradas deben estar codificadas.

El único nivel que garantiza una interoperabilidad semántica completa es el Nivel 3. (J. Pascual, 2019)

2.1.3. Estándar de interoperabilidad HL7 FHIR

Fast Healthcare Interoperability Resource (FHIR) es un estándar de interoperabilidad desarrollado por HL7 para permitir el intercambio electrónico de datos de atención médica entre sistemas médicos. (TIBCO, 2023)

FHIR utiliza interfaces de programación de aplicaciones (API) para permitir que aplicaciones se "conecten" a un sistema operativo básico, alimentando toda la información relevante en el flujo de trabajo del proveedor. FHIR admite el intercambio de información en una variedad de formatos, incluidos documentos, mensajes, servicios e interfaces RESTful. FHIR está diseñado para adaptarse a la creciente complejidad de los datos de atención médica, las expectativas de los usuarios y el requisito de un enfoque moderno y basado en Internet para

comunicarse entre diferentes componentes.

El objetivo principal de FHIR es abordar las crecientes necesidades de digitalización en la industria de la salud y simplificar el intercambio de datos sin comprometer la integridad de la información. Pretende que los registros de salud electrónicos estén disponibles, sean detectables y fácilmente comprensibles para las partes interesadas a medida que los pacientes se mueven dentro del ecosistema de atención médica.

FHIR es una evolución de CDA, ya que mientras CDA se enfoca en la creación de documentos clínicos estructurados y ricos en contenido, FHIR ofrece un enfoque más ágil y granular mediante el uso de recursos modulares que permiten intercambiar datos de manera más flexible y en tiempo real, adaptándose mejor a las necesidades actuales de interoperabilidad en el ámbito de la salud.

2.2. Historia Clínica Electrónica Nacional

La Historia Clínica Electrónica Nacional (HCEN) es una plataforma que permite acceder a la historia clínica digital de los usuarios del Sistema Nacional Integrado de Salud (SNIS) del Uruguay. Posibilita el registro de cualquier evento médico, independientemente del lugar geográfico y prestador de salud en donde se ofrezca la asistencia. La HCEN permite el intercambio de información clínica con fines asistenciales entre prestadores de salud, con el fin de asegurar la continuidad asistencial del usuario del SNIS.(AGESICl, 2023a)

La HCEN contiene el registro de todas las personas y de todos sus eventos clínicos que hayan sido registrados en formato electrónico. Conecta a todas las organizaciones de la salud a través de una red privada de datos llamada Red Salud. Cada vez que una persona se atiende en un prestador de salud, se genera un documento clínico en formato digital que es almacenado en el Repositorio del propio prestador. A la vez que se genera el documento clínico, también se genera un registro en la plataforma de HCEN que indica únicamente en qué repositorio del prestador se encuentra el documento, sin contener ninguna información clínica.

Si la misma persona se atiende posteriormente en otro prestador de salud, el especialista que realiza el acto asistencial puede consultar la historia clínica completa de la persona a HCEN sin importar en cuántos otros prestadores de salud se haya atendido previamente. Cada vez que de un prestador solicita un evento clínico de un paciente, se consulta el Registro de eventos clínicos de HCEN que tiene la información de en qué repositorio de la organización se encuentra el documento clínico. De esta forma, la organización a la que pertenece el especialista solicita el documento clínico a la organización que lo generó y lo custodia. (HCEN, 2024)

La plataforma HCEN pudo ser establecida gracias a los estándares y lineamientos de informática médica definidos por el programa Salud.uy, hoy denominado Salud Digital (AGESICl, 2023c). Alguna de las definiciones realizadas por el programa Salud Digital con respecto a la HCEN son el uso del estándar HL7 V3 CDA R2 para la definición de los documentos clínicos a ser intercambia-

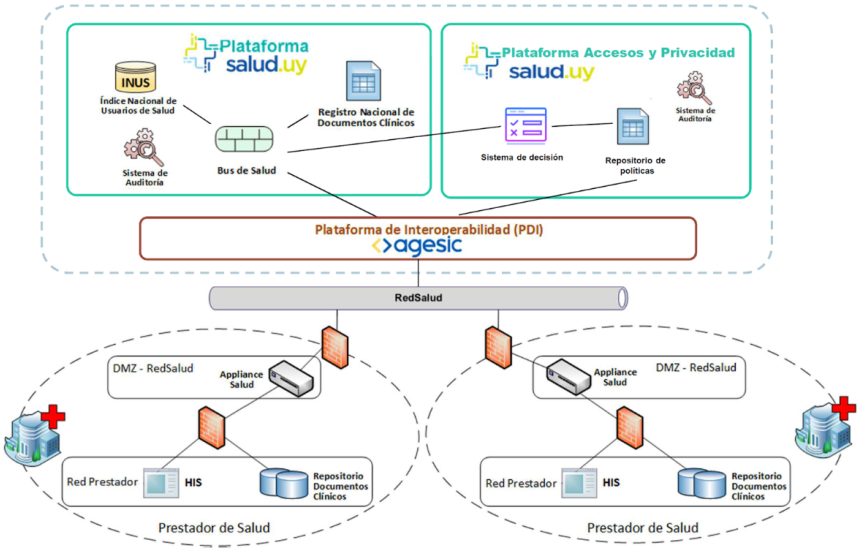


Figura 2.2: Componentes de la Historia Clínica Electrónica Nacional (HCEN). Se ve cómo interactúan los diferentes componentes de la Plataforma Salud Digital. El Repositorio de Documentos Clínicos de cada Prestador de Salud interactúa con el Registro Nacional de Documentos Clínicos a través del Appliance Salud. Tomado de (Salud Digital, 2024)

dos (Salud Digital, 2019) y el perfil Cross-Enterprise Document Sharing (XDS.b) de Integrating the Healthcare Enterprise (IHE) (IHE, 2023) para el intercambio de documentos clínicos (Salud Digital, 2022).

Salud Digital también provee componentes de infraestructura que brindan soporte al funcionamiento de los Servicios de HCEN así como a otros servicios y aplicaciones. Los componentes más importantes de la infraestructura de la Plataforma Salud Digital, para la comprensión de este proyecto, son el Índice Nacional de Usuarios de Salud (INUS), el Registro Nacional de Documentos Clínicos, el Bus de Salud y el Appliance Salud. La infraestructura definida por Salud Digital se puede ver en la figura 2.2

A continuación se describen algunos de sus componentes:

Índice Nacional de Usuarios de Salud (INUS): es un *Enterprise Master Patient Index* (EMPI) (TechTarget, 2017) que gestiona los datos patronímicos de los pacientes y tiene como finalidad identificarlos unívocamente dentro de la plataforma. El INUS utiliza un identificador global de paciente y gestiona además su correspondencia con los identificadores locales que maneja cada actor.

Registro Nacional de Documentos Clínicos: también llamado Registro XDS Nacional, es el índice de los documentos clínicos electrónicos generados y almacenados en los distintos prestadores integrados a la Plataforma Salud Digital.

Este registro permite conocer en qué prestador se encuentra cada documento clínico que conforma la historia clínica de un Usuario de Salud. El Registro XDS Nacional está basado en el perfil XDS.b de la IHE.

Bus de Salud: es un componente de tipo Enterprise Service Bus (ESB)(D. Chappell, 2004) y tiene la responsabilidad de ser el único punto de acceso a la Plataforma Salud Digital. Este componente centraliza la comunicación hacia y desde los demás componentes que forman parte de la plataforma.

Appliance Salud: El Appliance Salud es un componente de hardware y software que se aloja en cada Prestador de Salud y expone un conjunto de servicios web que ofician de intermediarios entre los sistemas del prestador y los servicios de la Plataforma Salud Digital. Los servicios que el Appliance Salud expone hacia los prestadores resuelven temas de integración con la Plataforma asociados a seguridad y disponibilidad, entre otros. Por otro lado, el Appliance Salud también expone servicios que pueden ser invocados por la Plataforma Salud Digital al momento de que otra institución de salud quiera, por ejemplo, recuperar un documento clínico custodiado por el prestador. Los servicios provistos por el Appliance Salud son un subconjunto de la interfaz de servicios XDS.b de IHE que hacen posible que los distintos Prestadores de Salud puedan registrar y consultar documentos clínicos.(AGESICl, 2023b)

Opcionalmente, Salud Digital también provee a los Prestadores de Salud un Repositorio XDS, Registro XDS y XDSBus locales para que puedan registrar y almacenar sus documentos clínicos e integrarse paulatinamente a la plataforma.(Equipo HCEN, 2020)

El XDSBus es responsable de exponer los webs services para la consulta y escritura del XDS local de cada institución y en el Registro XDS Nacional, a través del Appliance Salud.

El Repositorio XDS local expone los webs services requeridos para guardar y recuperar CDAs, mientras que el Registro XDS local expone los servicios necesarios para la consulta y escritura de los metadatos de los documentos clínicos.

2.3. Concepto de informática sin servidor

La computación sin servidor es un modelo de desarrollo de aplicaciones en el que se puede crear e implementar aplicaciones en una infraestructura de servidores administrados por terceros. Todas las aplicaciones requieren servidores para ejecutarse. Sin embargo, en el modelo sin servidor, un proveedor de servicios en la nube se encarga del trabajo rutinario; aprovisiona, escala y mantiene la infraestructura subyacente. El proveedor de servicios en la nube realiza varias tareas, como la administración del sistema operativo, la aplicación de parches de seguridad, la administración del sistema de archivos y la capacidad, el equilibrio de carga, la supervisión y el registro. Como resultado, los desarrolladores pueden centrarse en el diseño de las aplicaciones y seguir recibiendo los beneficios de una infraestructura de servidores rentable, eficiente y escalable de forma masiva.

¿Por qué es importante la computación sin servidor?

En los primeros días de Internet, cualquier persona que quisiera ejecutar una aplicación web tenía que comprar y mantener servidores físicos. Por lo general, las empresas almacenaban sus dispositivos de servidor físico en centros de datos locales. Esto podía resultar caro, ya que la mayoría de las aplicaciones solo utilizaban una fracción muy pequeña de los recursos del equipo del servidor.

El modelo de computación en la nube resolvió inicialmente este problema al permitir a los clientes crear servidores o instancias virtuales en el equipo del proveedor de servicios en la nube. Sin embargo, los clientes aún tenían que aprovisionar, configurar, actualizar y escalar sus servidores virtuales.

En respuesta a estos problemas, los proveedores de servicios en la nube comenzaron a ofrecer tecnologías sin servidor para aumentar la agilidad y optimizar aún más los costes. Con la computación sin servidor, sus desarrolladores pueden ejecutar código, administrar datos e integrar aplicaciones sin preocuparse por las tareas de administración de la infraestructura.

La adopción de la tecnología sin servidor tiene varios beneficios, que describimos a continuación.

Aumento de la productividad de los desarrolladores

Los equipos de desarrollo pueden centrarse en crear aplicaciones en lugar de configurarlas. La reducción de los gastos operativos significa que las aplicaciones llegan al mercado más rápido. Los desarrolladores pueden responder a los comentarios de los clientes y publicar cambios frecuentes en el código de las aplicaciones.

Escalabilidad eficiente

Los proveedores de servicios en la nube proporcionan una característica de escalamiento automático en los entornos sin servidores. Las aplicaciones sin servidor se escalan automáticamente de cero a picos de demanda. Los desarrolladores no tienen que pensar en el uso al escribir código.

Costos más bajos

Solo se paga por la CPU, la memoria y otros recursos informáticos necesarios cuando se ejecuta el código. No se paga nada por los recursos inactivos. Este modelo de facturación de pago por valor garantiza una utilización óptima de los recursos y no se desperdicia debido al exceso de aprovisionamiento. (AMAZON, 2023)

2.4. Ley de protección de datos personales

El dispositivo clínico DINABANG aloja su infraestructura tecnológica en centros de datos del exterior, lo que determina que nuestra solución deberá

considerar la normativa legal respecto a la transferencia internacional de datos personales.

El Artículo 23 de la Ley 18331(Ley 18331, 2008) de protección de datos personales establece las pautas y requisitos para la transferencia de datos personales fuera del territorio uruguayo.

Las resoluciones N.º 19/022 (URSEC, 2022a) y N.º 42/022 (URSEC, 2022b) de la Unidad Reguladora de Servicios de Comunicaciones (URSEC) reglamentan el cumplimiento del mencionado artículo de la ley 18331 para el proveedor de servicios en la nube de AMAZON.

2.5. Proyectos relacionados

En las siguientes secciones hacemos breves referencias a los proyectos del Núcleo de Ingeniería Biomédica (NIB) que han influenciado en el desarrollo del presente trabajo, como DINABANG, proyecto que origina el nacimiento de DOCLUY y SISENF, sistema con el que se logró una integración exitosa. También mencionaremos casos de éxito como la *Agenda de Vacunación COVID* que almacena información clínica en Uruguay y utiliza AWS.

2.5.1. Medida de la potencia muscular DINABANG

DINABANG es el proyecto base que origina el presente trabajo. Es un equipo biomédico de exportación de la empresa uruguaya "MOVI Technology for Life" patentado por la Universidad de la República. Fue desarrollado para medir la velocidad, la fuerza y la potencia de miembros inferiores durante la rehabilitación muscular. DINABANG es usado durante el entrenamiento de deportistas de élite o futbolistas profesionales. Es el resultado de la colaboración interdisciplinaria de clínicos e ingenieros en el ámbito de la Unidad Académica de Fisioterapia de la Facultad de Medicina y Núcleo de Ingeniería Biomédica de las Facultades de Medicinas e Ingeniería, ambos ubicados en el Hospital de Clínicas.

DINABANG incluye un elemento sensor de fuerza ("strain gauge") junto a una unidad inercial (IMU "inertial measurement unit") para medir fuerza, velocidad angular y aceleración del miembro inferior de la persona atada por una tobillera a las manos del fisioterapeuta mediante una cinta de goma elástica. En un terminal móvil (tablet o celular), el especialista ve en tiempo real qué fuerza hace el paciente o deportista y en qué momento y, por lo tanto, gradúa y ajusta su propia fuerza muscular para obtener el resultado óptimo. En la Tablet se almacenan informes y gráficos con los resultados obtenidos en las pruebas.

En la figura 2.3 se muestra el uso de DINABANG por parte de un especialista para medir fuerza a lo largo de un movimiento de flexo-extensión de miembro inferior en el proceso de rehabilitación muscular.

Es muy importante remarcar que este proyecto es un ejemplo de como un proyecto de fin de carrera se transforma en un producto y luego una empresa y, por lo tanto, en una fuente de innovación y creación de tecnología en nuestro país. (R.Barboza, J. Dominguez, A. Fernandez, 2018)

Figura 2.3: DINABANG en el consultorio de fisioterapia.
Notar la cinta elástica, el sensor de fuerza y microprocesador (de color blanco) atado a la tobillera y a la cinta elástica, la unidad inercial colocada en la parte posterior de la tobillera y la tablet con la medida en tiempo real para realimentación del paciente que la mira.(MOVI, 2023)

2.5.2. Sistema informático de enfermería SISENF

SISENF (G. Bonilla, M. Rocanova, 2023) es un sistema informático diseñado para apoyar la asistencia y docencia de enfermería, generando una historia clínica de enfermería que facilita el registro en la historia clínica del paciente. SISENF favorece el registro de datos del proceso de enfermería de una consulta general, tanto virtual como presencial, permitiendo a los profesionales de enfermería documentar sus procedimientos y obtener indicadores para evaluar su productividad, la calidad de la atención y la satisfacción del paciente.

Al igual que DINABANG y los dispositivos médicos utilizados para tomar mediciones de pacientes, la información generada por SISENF debe registrarse en la HCEN. Por este motivo, SISENF es otro ejemplo, de sistema clínico que puede utilizar la solución DOCLUY que se desarrolla en el presente proyecto.

2.5.3. Sistema informático de patología oncológica músculo esquelética SIPOME

El Sistema Informático de Patología Oncológica Músculo Esquelética (SIPOME) (N. Hourcade, V. Coggan, E. Della Mea, 2020) es un sistema informático innovador que tiene como objetivo mejorar la prestación integral de salud. SIPOME mejora el registro y acceso a la información clínica de los pacientes sa-

tisfaciendo las necesidades del Instituto Nacional de Ortopedia y Traumatología (INOT) y generando información en formato CDA que puede ser almacenada en la HCEN.

2.5.4. Agenda de vacunación COVID-19

En el contexto de la pandemia del año 2020 el Ministerio de Salud Pública (MSP) de Uruguay junto con la Agencia de Gobierno Electrónico y Sociedad de la Información y del Conocimiento (AGESIC), el programa Salud Digital y otros proveedores de tecnología, desarrollaron un sistema de solicitud de agenda para la campaña de vacunación contra el COVID-19 utilizando servicios en la nube de AWS. El sistema fue puesto en producción en dos días y permitió que toda la población adulta pueda solicitar agenda y obtener día y hora para vacunarse.

El Chief Technology Officer (CTO) de GeneXus Gaston Milano, fundamentó que la alternativa de mantener todo el sistema on-premises y hacer actualizaciones en los data center habría resultado más caro y hubiera tomado mucho más tiempo. "Una nube elástica como la que tiene AWS permite a los clientes agregar capacidad de servidores rápidamente para manejar un fuerte aumento en demanda. Después, dejan de usar esa capacidad adicional y aseguran mayor eficiencia y una mejor relación costo-beneficio", dijo el CTO. El ejecutivo agregó que el sistema se puso en operación en un tiempo récord de dos días.

El Director de Infraestructura y Operaciones de AGESIC Gabriel Hernandez, asegura que el mayor cambio que tuvieron que implementar fue pasar de un esquema sincrónico, como el utilizado por el sistema anterior, a un esquema asíncrono. La nueva infraestructura funciona como una capa de protección para que el sistema antiguo de agenda de horas no se vea saturado y deje de responder a quienes desean solicitar agenda. "La solución fue crear una capa protectora con Amazon Web Services que soportara los picos de demanda y pudiera entregar información, de forma que el sistema de agenda pudiera consumirlo a una mayor velocidad, sin atender directamente al público", explica Hernández.

El sistema de vacunación utiliza Amazon Simple Storage Service (Amazon S3) para almacenamiento, Amazon API Gateway, AWS Lambda para la protección de datos y Amazon DynamoDB para la parte de consultas de base de datos. Todo fue desarrollado en un modelo sin servidor, con funciones como servicios. Hernández indica que tener el Software Development Kit (SDK) de AWS para hacer la programación aún más simple fue fundamental. (AGESIC, 2021)

2.6. Productos existentes de interoperabilidad

En esta sección, exploramos aplicaciones médicas y plataformas de interoperabilidad, centrándonos específicamente en aquellas que operan con estándares HL7.

La primera parte de la investigación se centró en la búsqueda de soluciones que conecten dispositivos médicos portátiles con software de historia clínica.

Durante la investigación, identificamos un conjunto de aplicaciones que facilitan una conexión fluida y segura entre diversos dispositivos y sistemas de atención médica, las cuales describiremos a continuación.

Los protocolos HL7, reconocidos por su papel fundamental en la estandarización de la comunicación en salud, sirven como columna vertebral para la interoperabilidad, permitiendo el intercambio de información crítica de manera coherente y comprensible.

A continuación mencionamos algunos de los productos.

Aidbox C-CDA/FHIR Converter

Aidbox C-CDA/FHIR Converter (Aidbox, 2017) es una herramienta diseñada para facilitar la conversión de datos clínicos entre los estándares Consolidated Clinical Document Architecture (C-CDA) y Fast Healthcare Interoperability Resources (FHIR).

La conversión entre C-CDA y FHIR permite a los profesionales de la salud y a las instituciones utilizar diferentes sistemas y formatos de datos sin perder la integridad o la comprensión de la información clínica. Aidbox C-CDA/FHIR Converter actúa como un puente, facilitando la transición y la compatibilidad entre estos dos estándares.

AVRO FHIR

AVRO FHIR es una tecnología que combina el sistema de serialización de datos AVRO con el estándar FHIR y el manejo de datos en el ámbito de la salud.

AVRO es un sistema de serialización de datos desarrollado por Apache, diseñado para permitir la transmisión de datos en formatos binarios o JSON con alta eficiencia y compatibilidad entre diferentes lenguajes de programación.

La combinación de AVRO con FHIR permite:

Eficiencia en el Manejo de Datos: AVRO ofrece una forma eficiente de serializar y deserializar datos, lo que es crucial para el manejo de grandes volúmenes de datos de salud.

Interoperabilidad Mejorada: Al integrar AVRO con el estándar FHIR, se mejora la interoperabilidad entre sistemas de salud, facilitando la integración y el intercambio de datos de manera más efectiva.

Reducción de la Complejidad: La serialización con AVRO simplifica la estructura de datos, reduciendo la complejidad en el almacenamiento y transmisión de información de salud basada en FHIR.

Entonces AVRO FHIR busca optimizar la gestión y el intercambio de datos de salud al aprovechar las ventajas de ambos estándares, promoviendo una interoperabilidad más fluida y eficiente en el sector de la atención médica.(C. Altamirano, 2023)

Open mHealth

Open mHealth es una iniciativa que busca mejorar la interoperabilidad en el ámbito de la salud móvil mediante el desarrollo de herramientas que faciliten el intercambio de datos entre aplicaciones de salud y dispositivos. Su objetivo principal es permitir que diferentes tecnologías y plataformas trabajen juntas de manera fluida, promoviendo una integración más eficaz de los datos de salud recolectados de diversas fuentes. Esto se logra mediante la creación de especificaciones abiertas y un enfoque basado en estándares que aseguran que los datos puedan ser compartidos y utilizados de manera consistente y segura.

Uno de los estándares desarrollados por Open mHealth es el Open mHealth Data Model, el cual define una estructura común para la representación de datos de salud. Este modelo proporciona un marco para estandarizar cómo se describen y organizan los datos de salud, permitiendo que las aplicaciones y servicios de salud se comuniquen de manera efectiva y precisa. Al adoptar este estándar, los desarrolladores pueden crear soluciones que se integren fácilmente con otras plataformas y sistemas, mejorando así la interoperabilidad y la calidad del cuidado de salud.

Open mHealth y HL7 pueden ser vistos como complementarios. Mientras que HL7 se enfoca en la interoperabilidad a nivel de sistemas de salud en general, Open mHealth se centra en la integración de datos provenientes de aplicaciones móviles y dispositivos de salud. Los estándares de HL7, especialmente FHIR, pueden ser utilizados para mejorar la interoperabilidad de los datos definidos por los modelos de Open mHealth.(OpenmHealth, 2024)

2.7. Conclusiones del capítulo

La revisión de antecedentes abordó los conceptos, estándares, plataformas y productos que forman la base del próximo capítulo, en el que se detallarán los requisitos y la solución propuesta para cumplir con los objetivos establecidos del proyecto.

Capítulo 3

DOCLUY, solución general que satisface DINABANG-CDA

En este capítulo se presentan los requerimientos identificados para el proyecto DINABANG-CDA, así como la arquitectura diseñada para la solución que los cumple y una descripción general de DOCLUY, la solución implementada.

3.1. Requerimientos

El objetivo principal de este proyecto es desarrollar una solución de software que permita enviar información clínica desde distintos dispositivos y sistemas hacia repositorios de documentos clínicos. Esto permitirá que la información clínica generada se encuentre disponible en repositorios como, por ejemplo, la Historia Clínica Electrónica Nacional de los pacientes. Como objetivo específico se busca integrar y automatizar el envío de reportes generados por DINABANG a HCEN. A partir de estos objetivos y del análisis previo de HCEN y DINABANG, se desprenden los siguientes requerimientos, agrupados en funcionales y no funcionales.

3.1.1. Requerimientos funcionales

Envío de documentos clínicos a HCEN

El envío de información clínica a HCEN se realiza a través de documentos en formato CDA. Para enviar un CDA a la HCEN, se debe realizar lo siguiente:

1. Generar un documento CDA (Salud Digital, 2019) según el estándar HL7 V3 CDA R2, con la información clínica que se desea enviar a HCEN.

2. Firmar (Salud Digital, 2013) el CDA de forma electrónica.

3. Implementar la transacción ITI-41 ProvideAndRegisterDocumentSet que provee el Appliance Salud vía web service SOAP. (Salud Digital, 2022)

4. Implementar un mensaje del tipo Message Transmission Optimization (MTOM)(W3C, 2006) que incluya la transacción ITI-41 y el documento CDA firmado. (Salud Digital, 2022)

5. Enviar el mensaje MTOM al Appliance Salud (Salud Digital, 2022) a través de un POST HTTP.

Interfaz para envío de documentos CDA a HCEN

DOCLUY debe proveer una interfaz que permita a sistemas generadores de información clínica enviar la información a HCEN. Esta información debe ser procesada y transformada en un documento CDA nivel 1 para ser enviado a HCEN.

Envío de reportes de DINABANG a HCEN

Los reportes PDF generados por DINABANG deben poder ser enviados a HCEN utilizando la interfaz mencionada anteriormente. Por lo tanto, DINABANG debe poder proveer toda la información necesaria para consumir dicha interfaz, incluida información del paciente, del profesional de salud que toma las mediciones con el dispositivo, información del evento asistencial y el PDF que será incluido en el CDA nivel 1.

Integración con múltiples sistemas

La solución debe permitir que se integren múltiples sistemas externos tanto para recibir información clínica, como para enviar documentos clínicos.

Consulta de estado de documentos

Se debe proveer una interfaz que permita consultar el estado de un documento en proceso.

Autenticación y autorización

Un sistema externo puede acceder a la información del estado de un documento solo si se autentica como el sistema que originalmente envió la información del documento.

Notificar resultado de operación

Luego de procesar la información recibida por un sistema externo, se debe poder informar a dicho sistema del resultado del procesamiento del documento a través de un webhook. Se debe informar tanto si el procesamiento fue exitoso y el documento generado, como si se produjo algún error al generar el documento. El

envío del resultado se realiza consumiendo una API que debe proveer el sistema externo.

La solución no debe depender de la disponibilidad de un webhook por parte del sistema externo para continuar su funcionamiento.

Procesamiento asíncrono

La integración de un sistema externo con la solución no debe afectar su rendimiento ni causar demoras. Por lo tanto, la solución debe procesar la información recibida de forma asíncrona.

3.1.2. Requerimientos no funcionales

Escalabilidad de repositorios de documentos clínicos

Los repositorios de documentos clínicos almacenan y gestionan documentos clínicos en sus propias infraestructuras. Los documentos almacenados pueden seguir diferentes estándares tanto de almacenamiento como de comunicación. La solución diseñada debe poder escalar para permitir que se integren distintos repositorios que almacenen documentos en diversos formatos.

Compatibilidad con funcionalidades existentes

Los usuarios de DINABANG que no hagan envíos a HCEN no deben ser afectados por las nuevas funcionalidades. Esto implica que el envío a HCEN no sea un paso obligatorio en la creación y culminación de un reporte. De esta manera, sé mantiene una experiencia de usuario consistente para todos los usuarios.

Cumplir con normativa de almacenamiento de datos clínicos

Se debe cumplir lo establecido por la ley 18.331 de Protección de Datos Personales y en particular el artículo 23 respecto a la transferencia de datos personales fuera del territorio nacional.

Optimización de recursos en nube

La solución no debe generar costos extras por uso innecesario. Se debe optimizar los recursos actuales.

Compatibilidad multilenguaje

Los documentos CDA deben poder generarse en más de un idioma; al menos español e inglés.

Observabilidad

El sistema debe registrar trazas y logs que permitan identificar errores en el procesamiento y diagnosticar problemas de manera efectiva.

3.2. Arquitectura propuesta

En esta sección exponemos la arquitectura propuesta para el desarrollo de la solución que satisface los requerimientos vistos en la sección 3.1.

La arquitectura de DOCLUY está compuesta por diferentes componentes como se muestra en la figura 3.1. Cada componente tiene un rol específico en el proceso de generación de un documento clínico. Se recibe la información clínica a través de un endpoint (Servicio API Gateway de AWS). Este servicio realiza los chequeos de autenticación y consumo del generador o fuente de documentos (Ejemplo: DINABANG, SISENF). La información recibida es tomada por una función AWS Lambda. El servicio AWS Lambda permite la creación de múltiples funciones Lambda, las cuales se encargan de ejecutar diferentes porciones de código que, en conjunto y junto a otros servicios, conforman el sistema completo. Esta primera función AWS Lambda (Receptor o Document Receiver), toma la información clínica recibida desde el generador, verifica su unicidad y la deposita en una cola de mensajes (Cola de Entrada o Input Queue). La cola de entrada se utiliza para encolar toda la información clínica recibida que esté pendiente de ser procesada. Esta cola es implementada con el servicio Amazon Single Queue Service, y permite el procesamiento asincrónico de la información clínica, haciendo que la API Rest que recibe la información pueda dar una respuesta inmediata a las aplicaciones que la consumen sin detener o entorpecer su funcionamiento. Los mensajes encolados en la cola de entrada son tomados por una función lambda encargada de iniciar el procesamiento (Document Processor). Esta lambda toma la información encolada y dependiendo del tipo de documento destino, inicia la ejecución de la lambda encargada de procesar la información (FHIR Starter, Custom Starter, CDA Starter). En el caso de documentos clínicos dirigidos a repositorios CDA, la lambda "CDA Starter" inicia un AWS Step Function (AWS, 2024m), que da como resultado el documento CDA correspondiente a la información recibida y el repositorio destino. Una vez terminado el procesamiento, la lambda encargada de generar el documento, deposita el documento en una cola de salida (Cola de Salida u Output Queue) listo para ser enviado. La inclusión de esta segunda cola, permite reintentar el envío del documento en caso de que no haya conectividad con el repositorio. La lambda "Enviador" (Document Sender) toma el documento de la cola de salida, e intenta enviarlo al repositorio. En caso de lograrlo, marca el documento como enviado y el procesamiento se da por finalizado (todos los estados por los que puede pasar un documento en el flujo de procesamiento se muestran en la figura 3.2). En caso de error, se envía el documento a la cola de reintentos, la cual tiene un retraso incremental, lo que permite volver a intentar el envío en un tiempo futuro.

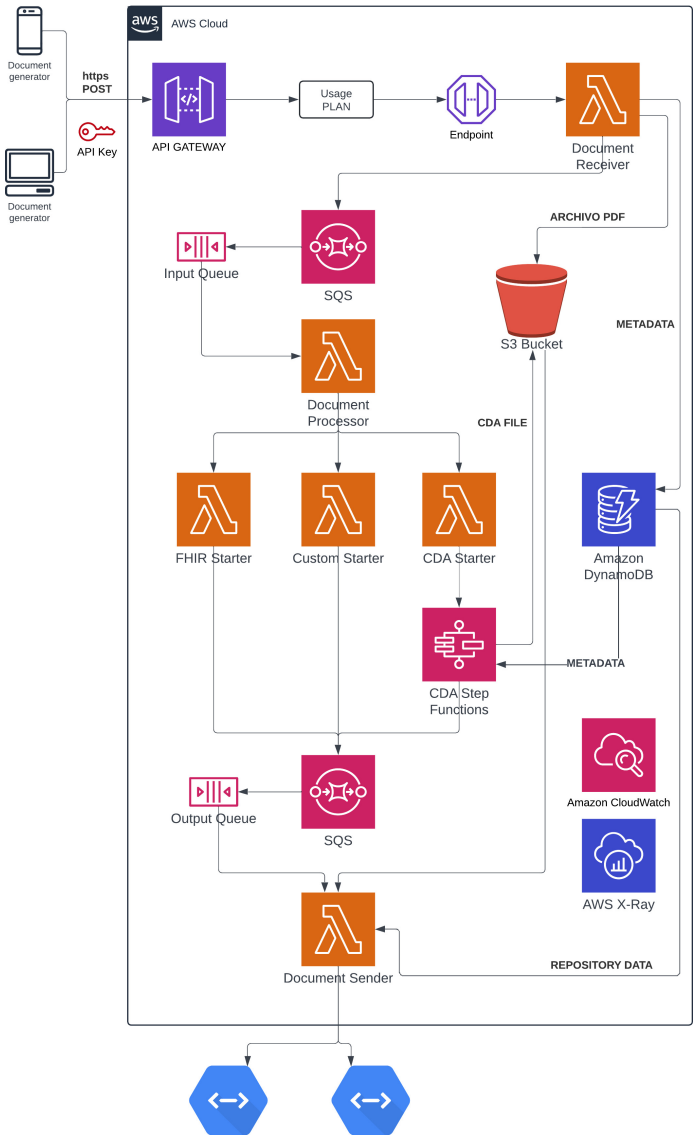


Figura 3.1: Diagrama de Arquitectura de la solución en AWS
Se representa la API Gateway que es punto de entrada de las solicitudes realizadas por los Generadores de Documentos, los Step Functions que orquestan procesamiento, las funciones Lambda que ejecutan código, las colas de mensajes SQS que permiten el envío de información entre componentes, Amazon CloudWatch que registra logs y AWS X-Ray que registra trazas.

En las siguientes secciones se presenta en detalle cada punto de este flujo, y

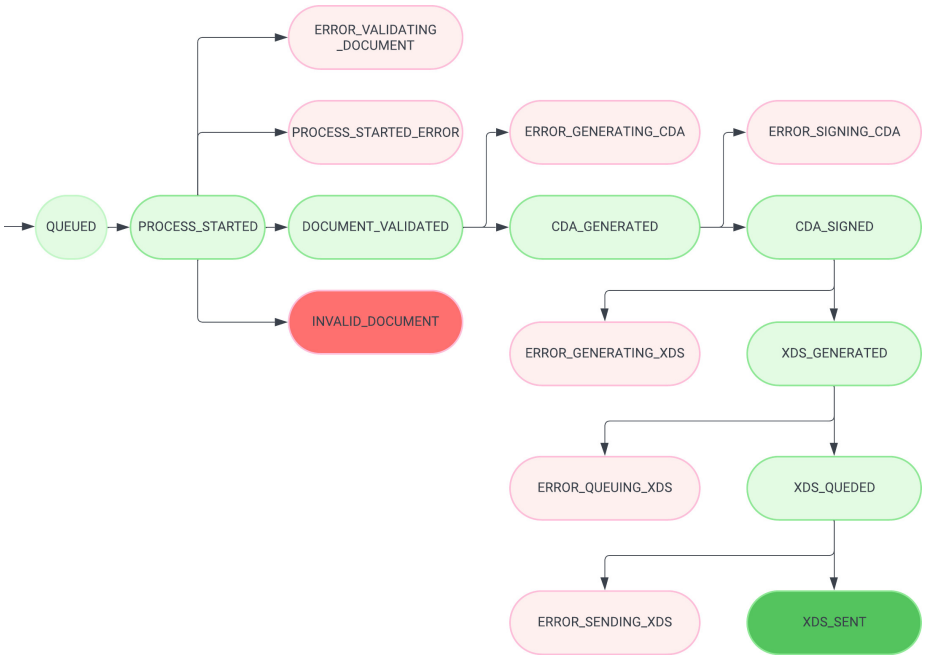


Figura 3.2: Estados de procesamiento CDA.
La generación de un CDA pasa por diferentes etapas, en las cuales el proceso puede estar en progreso o en estado de error. En rojo se ven los estados de error (internos y externos), en verde los estados en flujo normal.

las componentes diseñadas para llevarlo a cabo.

3.2.1. Decisiones Principales

En esta sección exponemos brevemente las decisiones más importantes que fueron tomadas.

Como nuestro principal objetivo es comunicar diferentes sistemas para realizar el envío de información clínica y generar documentos clínicos, en primera instancia, evaluamos diferentes formas comunicación entre múltiples sistemas. Consideramos tanto la comunicación directa entre generadores de información y repositorios, como la implementación de un **middleware** (AWS, 2024i). Debido a que es una prioridad para nosotros diseñar una solución que tenga la capacidad de integrar nuevos generadores de información y repositorios clínicos con el menor esfuerzo posible, optamos por desarrollar un middleware. Esta decisión nos permite definir una arquitectura flexible que reduce la complejidad de futuras integraciones y mantenimiento, en contraste a la alternativa de comunicación directa. La evaluación completa de estas alternativas se encuentra en el Anexo A. La tabla de decisión 3.1 muestra los aspectos evaluados.

Criterio	Comunicación Directa	Middleware
Facilidad de Implementación	Alta	Media
Latencia	Baja	Media
Escalabilidad	Baja	Alta
Mantenibilidad	Baja	Alta
Desacoplamiento	Baja	Alta
Complejidad Inicial	Baja	Media
Redundancia en Transformaciones	Alta	Baja

Tabla 3.1: Comparación de alternativas de comunicación entre múltiples sistemas.

Una vez definida la implementación de un middleware, evaluamos el tipo de infraestructura a utilizar. Consideramos utilizar una infraestructura propia, implementar la solución en la infraestructura de los repositorios clínicos, o emplear servicios en la nube. Decidimos utilizar infraestructura en la nube; una elección basada en los beneficios de escalabilidad, alta disponibilidad y los reducidos costos iniciales. La evaluación completa realizada se encuentra en el Anexo B. La tabla de decisión 3.2 muestra una comparación entre las distintas alternativas.

Criterio	Infraestructura en la Nube	Infraestructura Propia	Infraestructura de Repositorios
Escalabilidad	Alta	Media	Baja
Disponibilidad y Tolerancia a Fallos	Alta	Media	Baja
Costos Iniciales	Bajos	Altos	Bajos
Costos Recurrentes	Medios	Bajos	Medios
Mantenimiento y Actualizaciones	Gestión por Proveedor	Autogestionado	Dificultoso
Control sobre la Infraestructura	Medio	Alto	Bajo
Latencia	Media	Baja	Alta
Seguridad y Cumplimiento	Alta	Alta	Variable

Tabla 3.2: Comparativa de tipos de infraestructura consideradas.

Para determinar de qué forma implementaríamos el procesamiento asíncrono de información, evaluamos el uso de microservicios y eventos, procesamiento en lotes y colas de mensajes. Decidimos utilizar colas de mensajes, ya que nos permiten desarrollar un sistema desacoplado, escalable (especialmente durante picos de carga) y con una complejidad de implementación y mantenimiento

media que estábamos dispuestos a enfrentar. La evaluación completa de alternativas de procesamiento asíncrono se encuentra en el Anexo C. La tabla de decisión 3.3 muestra las características consideradas.

Criterio	Procesamiento en Lote	Colas de Mensajes	Microservicios y Eventos
Facilidad de Implementación	Alta	Media	Baja
Latencia	Alta	Baja	Baja
Escalabilidad	Media	Alta	Alta
Mantenibilidad	Alta	Media	Baja
Desacoplamiento	Baja	Alta	Alta
Complejidad Inicial	Baja	Media	Alta
Adecuado para Procesamiento en Tiempo Real	No	Sí	Sí
Gestión de Picos de Carga	Baja	Alta	Alta

Tabla 3.3: Comparativa de alternativas de procesamiento asíncrono.

Luego de definir el uso de infraestructura en la nube, decidimos utilizar Amazon Web Services (AWS) como proveedor de infraestructura, una elección basada en su escalabilidad, alta disponibilidad y cumplimiento normativo con la legislación uruguaya sobre protección de datos. En el Anexo D se encuentra la evaluación completa de las alternativas evaluadas.

Se puede ver una breve descripción técnica de todos los servicios de AWS utilizados en Anexo E y la inversión realizada en etapa de validación técnica, desarrollo y testing en el Anexo F.

En el diagrama 3.1, se presenta la solución propuesta, que aprovecha las potentes capacidades de AWS. Dentro de los servicios de AWS utilizados se destacan: AWS Lambda como servicio de cómputo para ejecutar código; Amazon DynamoDB para almacenar información en bases de datos NoSQL; Amazon Single Queue Service para colas de mensajes; y Amazon API Gateway para las comunicaciones y control de uso.

Esta solución está diseñada para recibir información clínica desde distintos generadores de información (dispositivos o sistemas), crear documentos clínicos y enviarlo al repositorio correspondiente. Esta arquitectura implementa mecanismos para **proteger el uso de la API REST** (mediante APIKey), asegurar la **trazabilidad del estado de los documentos** y **responder ante posibles errores ocasionados por conexión con terceros**. También se realiza un **monitoreo del sistema** con el fin de garantizar su correcta operación y eficiencia.

Durante este capítulo explicaremos el funcionamiento general del middleware y el detalle de cada una de las componentes que lo conforman.

3.2.2. Componentes de DOCLUY

En esta sección se exponen en detalle cada uno de los componentes principales que conforman el flujo de generación de documentos clínicos de DOCLUY.

API Rest

Como se muestra en la figura 3.3 la API REST es el punto de entrada para que los generadores de documentos envíen la información al middleware, y puedan realizar consultas de forma controlada y segura. Se utiliza Amazon API Gateway para gestionar la creación, mantenimiento y seguridad de la API.

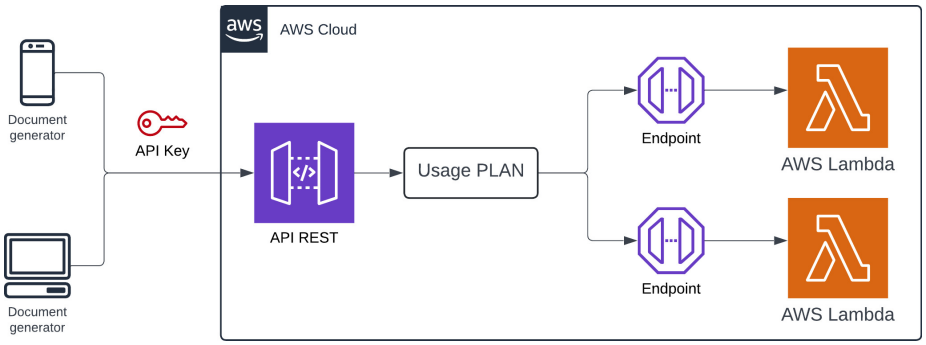


Figura 3.3: Implementación de API Rest mediante Amazon API Gateway

Amazon API Gateway proporciona una interfaz HTTP segura para la API REST, gestionando la autenticación, autorización y control de uso mediante API Keys.

Cada generador de documentos autorizado recibe una API Key que se debe incluir en las solicitudes HTTP para autenticar y autorizar la comunicación con el sistema. Esta API Key funciona como un identificador único, con la cual se puede autenticar (saber qué generador está realizando la solicitud) y autorizar (controlar que tenga los permisos necesarios para la operación que está intentando realizar). El uso de API Key también permite aplicar configuraciones específicas para controlar el uso y evitar abusos de servicio. Cada generador de documentos, a través de su API Key tiene asociado un límite de llamados a la API, el cual permite a DOCLUY controlar la estabilidad y evitar el abuso de servicio que se puede dar por diferentes factores. El uso de API Key aumenta la seguridad de DOCLUY, ya que garantiza que únicamente los generadores de documentos autorizados puedan enviar documentos a través DOCLUY.

Colas de mensajes

Para garantizar el procesamiento asíncrono se incluyen dos colas de mensajes principales, y una cola secundaria para orquestar los reintentos en caso de falla.

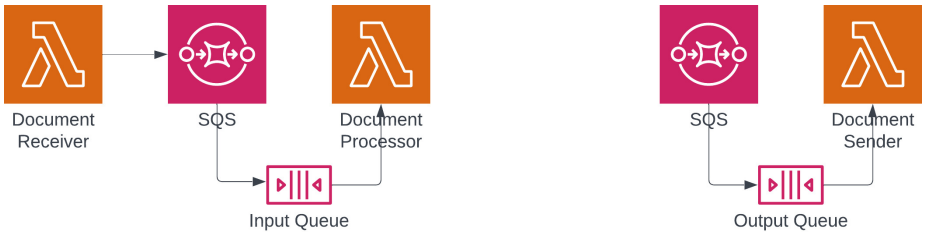


Figura 3.4: Colas de mensajes utilizadas en el flujo general.
La cola de entrada recibe la información clínica a ser procesada. La cola de salida gestiona el envío de los documentos generados a los repositorios.

Las colas de mensajes nos permiten desacoplar la recepción de información de la generación del documento, mejorando la escalabilidad y robustez del sistema.

En el diagrama 3.4 se muestran las dos colas de mensajes implementadas en el flujo general del procesamiento de un documento.

Se crea una cola de entrada, la cual tiene la función de recibir información entrante que necesita ser procesada para generar el documento clínico. El uso de esta cola permite que DOCLUY reciba la información, almacene los datos necesarios y de una respuesta inmediata. De esta manera, el generador de información clínica no quedará a la espera del procesamiento para continuar con su flujo y podrá visualizar su información como entregada. Para DOCLUY, esta información quedará almacenada, el documento quedará en estado “PENDING” con su correspondiente identificador en la cola de entrada.

En cuanto la lambda de procesamiento quede disponible, tomará el siguiente documento de la cola de entrada y lo enviará a procesar. La información clínica es procesada, y como resultado de este proceso se genera el documento clínico a ser enviado. Una vez generado este documento, se coloca en la cola de salida. El uso de la cola de salida permite que DOCLUY pueda procesar tantos documentos como le sea posible sin depender de la conectividad de receptor para culminar la generación del documento. Una vez en la cola de salida, es tomado por la lambda encargada de hacer el envío al repositorio. Esta lambda toma documentos de la cola de salida, y en caso de no poder hacer el envío, lo envía a una cola secundaria con un retraso de un minuto. Una lambda encargada de los reintentos, pasado el tiempo de retraso, toma el documento y lo vuelve a colocar en la cola de salida. Este proceso se repite con un máximo de cinco reintentos, e incrementando el tiempo de retraso.

Las capacidades, tanto de las colas como de procesamiento, dependen del presupuesto invertido y recursos contratados en AWS. Si bien AWS permite la auto-escalabilidad, lo que implica que los recursos se incrementen automáticamente en caso de ser necesario, DOCLUY tiene incorporado un límite de llamados a la API. Este límite es por fuente de información clínica (ej.: DINABANG o SISENF) y controla la carga de trabajo que cada fuente puede demandar a DOCLUY. Del mismo modo, también es limitada la cantidad de

reintentos, frente a una falla.

Cabe aclarar que los límites implementados en este trabajo fueron en contexto de desarrollo, para ser ajustados en entornos productivos se deben hacer un análisis mediante pruebas de estrés, el cual tiene un costo.

Step functions

Con Amazon Step Functions(AWS, 2024c) coordinamos y gestionamos el flujo de trabajo para la creación del documento. En la figura 3.5 se puede ver de qué manera la máquina de estados interactúa con las colas de entrada y de salida, siendo esta máquina de estados la encargada de producir el documento clínico.

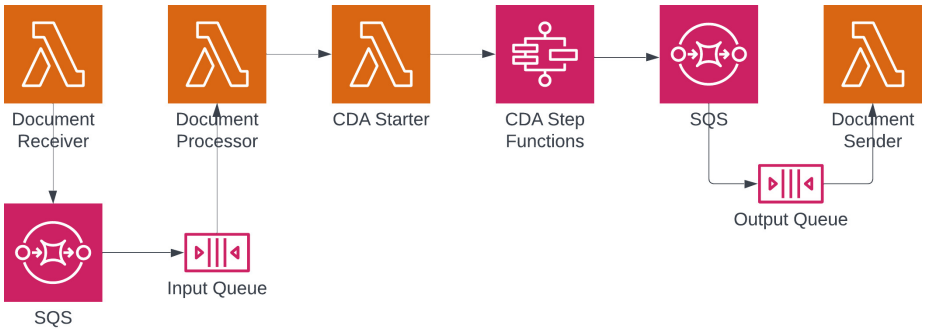


Figura 3.5: Máquina de estados utilizada en el flujo general

La máquina de estados describe las secuencia de pasos necesarios para la generación del documento según el formato. Cada tarea en el flujo de trabajo es manejada por una función Lambda. Si ocurre un error, Step Functions gestiona los reintentos y la lógica de error según se describe en el apartado de manejo de errores.

Todas las funciones Lambda de la solución utilizan Node.js como lenguaje de programación. En el Anexo G se presenta el análisis de otras alternativas consideradas.

DynamoDB y S3 para almacenamiento

La información clínica se conforma tanto por archivos, como por en metadatos. Se utiliza Amazon DynamoDB para almacenar los metadatos de los documentos y Amazon S3 para almacenar los archivos PDF recibidos y documentos generados. En la figura 3.6 se pueden ver las comunicaciones con estos dos medios de almacenamiento a lo largo del procesamiento del documento.

No solo se almacenan los archivos PDF recibidos como parte de la información clínica, sino que también se almacenan en S3 los diferentes archivos

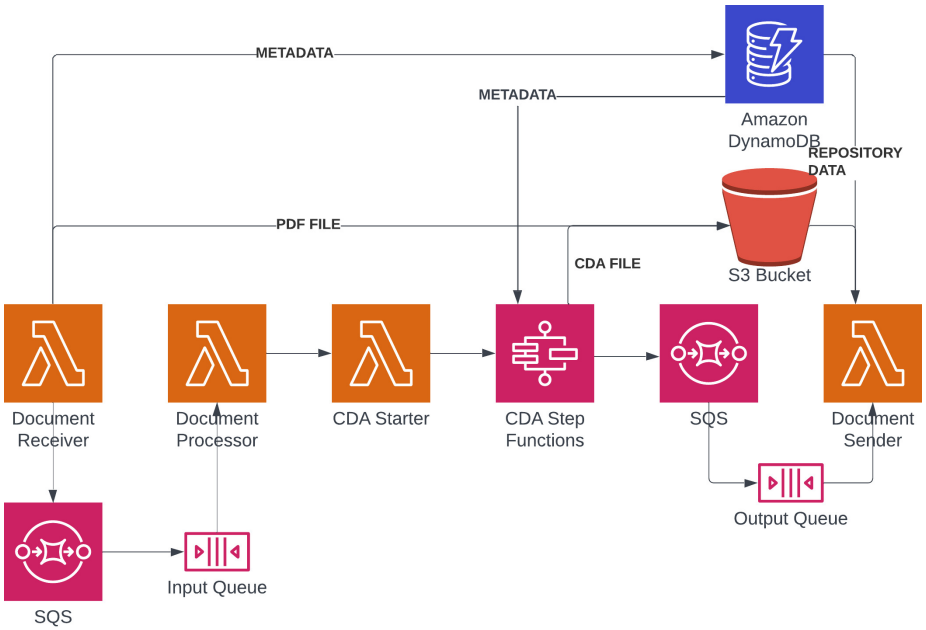


Figura 3.6: Diagrama de medios de almacenamiento

intermedios generados durante el procesamiento, necesarios para la conformación del CDA. Los documentos son enlazados a sus correspondientes archivos en S3 a través de su identificador único asignado.

En el Anexo H se puede ver el análisis comparativo de las alternativas evaluadas para el almacenamiento de información.

Para asegurar la correcta protección de información, se propone utilizar cifrado de datos en reposo tanto en DynamoDB como en S3, y políticas de acceso restringido a los buckets (contenedor de objetos para almacenar datos) de S3.

3.2.3. Procesos principales

La operación principal que realiza el middleware es el envío de documentos clínicos. Podemos visualizar esta operación **en tres subprocesos principales**:

1. Recepción de Información Clínica
2. Generación de Documento Clínico
3. Envío de Documento Clínico a repositorio

Esta separación permite un **procesamiento asíncrono**, ya que la generación del documento y el posterior envío son absolutamente independientes de la recepción de la información. De esta manera se asegura la disponibilidad del

sistema frente a posibles fallos en la generación de documentos y comunicación con repositorios.

Recepción de información clínica

Esta componente se encarga de recibir la información clínica, almacenarla en la base de datos y encolarla para que pase a la siguiente etapa del procesamiento.

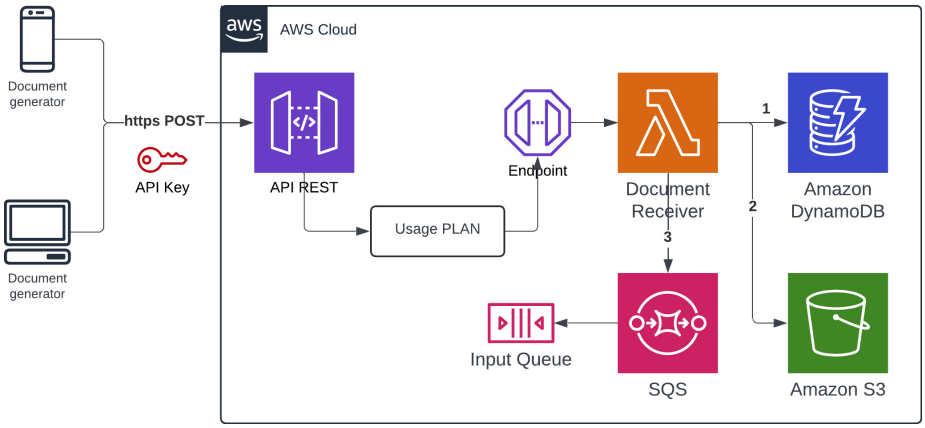


Figura 3.7: Proceso de recepción y almacenamiento de información clínica

Como se ve en la figura 3.7, para recibir la información, se expone un endpoint mediante Amazon API Gateway. Este endpoint recibe los metadatos necesarios, junto al archivo correspondiente. En el Anexo I se encuentra el análisis realizado para determinar de qué forma recibir la información de los generadores.

Una vez que API Gateway recibe la solicitud HTTP, verifica la API Key contra las claves registradas para asegurarse de que la fuente esté autorizada. Si la API Key es válida, la solicitud se pasa a la función Lambda encargada de recibir el documento.

La función Lambda **verifica en DynamoDB si la información para la creación del documento ha sido recibida anteriormente o no** (identificada por su identificador externo, versión y repositorio). Si ya fue recibida, devuelve un error indicando que el documento ya fue enviado para su procesamiento. En caso contrario, se le asigna un identificador global único y se procede al almacenamiento de la información.

Los **metadatos del documento** (información del autor, paciente, estado del procesamiento, etc.) **se almacenan en DynamoDB. El archivo PDF del documento se guarda en un Bucket de Amazon S3.**

Una vez almacenada la información necesaria, **es colocada en la cola de entrada** en Amazon SQS, lo que indica que la información está lista para ser procesada.

Generación de documentos clínicos

Una función Lambda se activa cada vez que **se recibe información en la cola de mensajes SQS**. Una vez detectado un documento en la cola de entrada, la función Lambda actualiza el estado del documento a **IN_PROGRESS** (Ver tabla de estados en Anexo K) e inicia el step function correspondiente al tipo de documento que se quiere generar.

En la figura 3.8 se muestra la máquina de estados encargada de generar un documento en formato CDA.

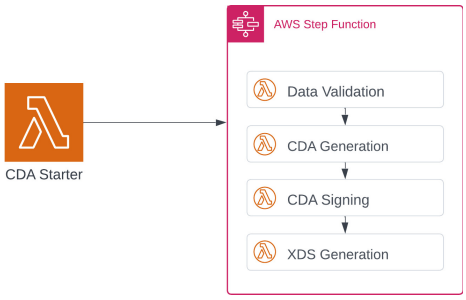


Figura 3.8: Máquina de estados de la generación de un archivo CDA

La primera tarea en el Step Function es **validar la información recibida**. Esto asegura que la información es completa y que respeta el formato esperado. Si la validación falla, el proceso entra en el flujo de manejo de errores. En caso contrario, se pasa a la siguiente etapa del step function, la cual **crea un documento en formato CDA nivel 1**, conforme a los estándares clínicos correspondientes.

El siguiente paso se encarga de firmar el documento (según las indicaciones del repositorio destino), garantizando su autenticidad e integridad. El paso final genera el mensaje que contiene el documento CDA firmado en el formato requerido por el repositorio. Este mensaje es luego colocado en la cola de envío (SQS), listo para ser enviado al repositorio.

Si alguno de los pasos falla, **se activa el flujo de manejo de errores** que intenta la recuperación (en caso de ser posible), notifica los errores y actualiza el estado del documento en DynamoDB. Lo veremos en detalle más adelante.

Este flujo de procesamiento asegura la integridad y autenticidad de los documentos generados antes de su envío a los repositorios correspondientes. El uso de AWS Step Functions parte el procesamiento en pasos, esto mejora notoriamente el proceso de identificar errores, ya que este servicio permite monitorear cada paso del proceso (tanto con logs de CloudWatch ,como en la consola de AWS Step Functions). Esto facilita al equipo de desarrollo la tarea de identificación de errores, aumentando la productividad y permitiendo dar una respuesta más rápida ante fallos inesperados.

Envío de documentos a repositorios

Una vez creado el documento y enviado a la cola de salida, el proceso de envío se realiza de manera automatizada mediante funciones Lambda. Esta función se activa cuando un documento llega a la cola, como se muestra en el diagrama 3.9.

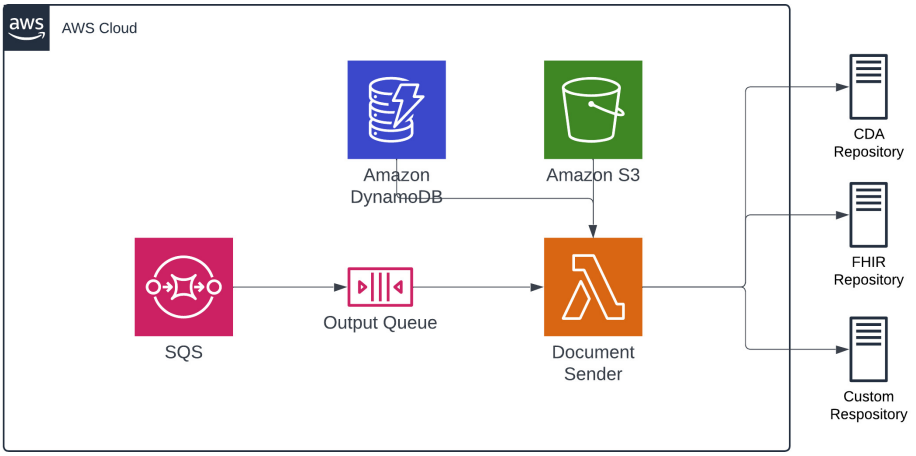


Figura 3.9: Arquitectura del proceso de envío de documentos a repositorios clínicos

La función Lambda implementa el **método de comunicación asociado con el repositorio de destino**. Por ejemplo, para enviar documentos a HCEN se utiliza un servicio web que recibe documentos en formato XML; para enviar documentos a un repositorio FHIR se utiliza una API REST que recibe documentos en formato JSON. Los detalles de la comunicación, como las credenciales y endpoints, se obtienen de la configuración almacenada en DynamoDB asociada al repositorio correspondiente. Se puede ver más detalle del envío a los repositorios en el Anexo J.

Una vez enviado el documento, se actualiza el estado a **SENT** (ver tabla de estados en Anexo K.

Si por algún motivo el envío falla, el documento entra en flujo de manejo de errores. El flujo de manejo de errores incluye un mecanismo de reintento, que intenta enviar el documento nuevamente después de un tiempo de espera. Adicionalmente, se registran los detalles del error y se actualiza el estado del documento en DynamoDB.

Este proceso asegura que, dentro del correcto funcionamiento de AWS, los documentos clínicos se envíen de manera segura y eficiente, ya que **optimiza la continuidad del servicio** debido al la recuperación del sistema frente a errores, con sus correspondientes reintentos.

3.2.4. Manejo de errores y reintentos

Dado que DOCLUY depende de terceros para completar sus tareas, es importante mantener una buena gestión de los errores y políticas de reintento. Durante cada uno de los sub-procesos del sistema, se identifican, registran y notifican los errores que ocurren.

Durante cualquier etapa del flujo (recepción, procesamiento, envío), si ocurre un error, **este es identificado y clasificado**. Los errores pueden ser causados por problemas de validación, fallos en la comunicación con repositorios (corte prolongado de internet), errores de transformación, etc.

Una vez detectado un error, **se registra en DynamoDB** con detalles específicos, incluyendo el tipo de error, el paso en el que ocurrió y un mensaje descriptivo. A continuación se muestra un ejemplo de posible error registrado:

```
{
  "document_id": "123456",
  "status": "ERROR_PARTIAL",
  "error_details": {
    "type": "CommunicationError",
    "step": "SendToRepository",
    "message": "Failed to connect to Salud Digital web service"
  },
  "timestamp": "2024-06-02T12:00:00Z"
}
```

Notificación de error

Se envía una notificación a un canal de Slack (Slack, 2024) designado para el administrador del sistema. La notificación incluye los detalles del error y el estado del documento.

Decisión de reintento

Durante el procesamiento y envío de los docuemntos, DOCLUY puede detectar diferentes tipos de errores. Estos errores se pueden clasificar entre recuperables y no recuperables. Un error que se da por inconsistencia en los datos recibidos, es un error frente al que DOCLUY no se puede recuperar, debe ser resuelto por generador de la inforamción. En este caso, el documento queda en estado de ERROR, y el generador es notificado del error mediante el uso de webhooks. Por el contrario, si es el error puede ser de carácter circunstancial (el servidor del repositorio destino está caído), DOCLUY pone este documento en el proceso de reintento.

De esta manera, errores no recuperables resultarán en la terminación del procesamiento con la notificación correspondiente, mientras que los recuperables serán ingresados al proceso de reintentos, el cual se describe a continuación.

Proceso de reintento

Cuando DOCLUY se encuentra frente a un proceso de reintento, el documento es enviado a una cola de espera con un tiempo incremental según el número de reintento. El primer reintento se realiza al minuto entrar en flujo de reintento. El segundo intento se realiza a los 5 minutos, el siguiente a los 20 minutos y el último a los 60 minutos. Para esto se configura una cola con delay, con la cual el documento queda en espera hasta ser ingresado nuevamente en la cola de procesamiento.

Finalización con error

Si todos los reintentos fallan, el documento se marca con un estado de "ERROR" en DynamoDB. Una notificación final se envía al administrador del sistema con el resultado del fallo definitivo.

Estos mecanismos aseguran que los errores son manejados de manera efectiva y transparente, proporcionando la capacidad de reintentar automáticamente los errores temporales y notificando adecuadamente al administrador del sistema sobre los problemas que requieren intervención manual. Esto mejora la robustez y fiabilidad del sistema, asegurando la continuidad del procesamiento de documentos clínicos.

3.2.5. Monitoreo y observabilidad

Para monitorear el estado del sistema utilizamos Amazon CloudWatch y AWS X-Ray.(AWS, 2024a)

Con Amazon CloudWatch se definen métricas para monitorear el rendimiento y la salud del sistema, como el tiempo de procesamiento de documentos, la tasa de errores, etc. También se activa CloudWatch Logs para registrar y centralizar los logs generados por los distintos componentes de la solución. Son configuradas alertas para notificar sobre problemas potenciales, como altas tasas de error, volumen de requests inusuales, etc.

Con AWS X-Ray registramos y consultamos trazas de las solicitudes procesadas a través de toda la solución.

3.2.6. Trazabilidad de documentos

La trazabilidad de documentos es esencial para monitorear y auditar el estado de los documentos durante su ciclo de vida en el sistema.

Para implementar la trazabilidad, se proporciona un endpoint que permite consultar el estado de un documento específico. Durante todo el proceso, se guarda un registro de actividad detallado que permite seguir el estado del documento paso a paso.

Este mecanismo asegura una trazabilidad completa y transparente de los documentos, permitiendo a los administradores y sistemas externos monitorear el estado y progresión de los documentos.

Registro de actividad

Cada documento tiene un historial de actividad que documenta cada paso del proceso de procesamiento y envío. Cada registro almacena el código del resultado de la ejecución de la actividad y la fecha y hora en la que finalizó en formato *Tiempo Universal Coordinado* (UTC) (IBM, 2021). En la figura 3.10 se muestra un ejemplo de registro de actividad.

Los estados de un documento se pueden consultar en el Anexo K

Endpoint para consultar el estado de un documento

Se proporciona un endpoint en la API Gateway para consultar el estado actual de un documento específico. Este endpoint permite a los sistemas generadores de documentos y otros sistemas autorizados obtener el estado y el historial de actividad del documento.

$GET/document/status/\{document\_id\}$

3.2.7. Seguridad

Debido a la naturaleza sensible de los datos, la seguridad es un aspecto importante en este sistema. A continuación, se detallan las estrategias y medidas de seguridad definidas en el middleware para garantizar la protección de esta información.

API gateway y API keys

Se utiliza el servicio de API Gateway para manejar todas las solicitudes entrantes. La autenticación de las solicitudes se hacen mediante API Keys, asegurando que solo los generadores de documentos autorizados puedan acceder al servicio. Además, se asignan roles y permisos específicos a cada API Key para limitar el acceso a las funcionalidades necesarias.

IAM (Identity and Access Managment)

IAM (Identity and Access Management) es el servicio de AWS que permite gestionar de forma segura el acceso a los recursos de la nube. Como parte de esta arquitectura, se realiza una correcta configuración aprovechando el potencial de este servicio. Esto implica que cada uno de los servicios que se visualizan en el diagrama de arquitectura (figura 3.1), tiene permisos de acceso limitados únicamente a los servicios necesarios para llevar a cabo su tarea.

De este modo, solo los servicios y componentes que realmente lo necesiten pueden acceder a la base de datos, al servicio de almacenamiento de archivos o a las colas de mensajes.

Cifrado en tránsito

Esta arquitectura propone el uso de HTTPS para todas las comunicaciones entre los generadores de documentos y la API Gateway. A sí mismo, todas las

```
{
  "activity_history": [
        {
            "date": "2024-06-02T20:03:28.913Z",
            "code": "QUEUED"
        },
        {
            "date": "2024-06-02T20:03:31.044Z",
            "code": "PROCESS_STARTED"
        },
        {
            "date": "2024-06-02T20:03:35.707Z",
            "code": "VALIDATED"
        },
        {
            "date": "2024-06-02T20:03:38.661Z",
            "code": "CDA_GENERATED"
        },
        {
            "date": "2024-06-02T20:03:42.047Z",
            "code": "CDA_SIGNED"
        },
        {
            "date": "2024-06-02T20:03:45.276Z",
            "code": "XDS_GENERATED"
        },
        {
            "date": "2024-06-02T20:03:45.874Z",
            "code": "XDS_QUEDED"
        },
        {
            "date": "2024-06-02T20:03:49.060Z",
            "code": "XDS_SENT"
        }
    ]
}
```


Figura 3.10: Ejemplo de registro de actividad de un documento. Cada registro almacena el resultado de la actividad y la fecha en la que finalizó en formato UTC

comunicaciones internas entre servicios de AWS también utilizan conexiones seguras.

De esta manera se establece una conexión segura entre el cliente y el servidor. Durante este proceso, se intercambian claves de cifrado que aseguran que los datos transmitidos no puedan ser leídos ni alterados por terceros mientras viajan a través de la red. HTTPS protege la confidencialidad e integridad de la información, garantizando que únicamente el cliente y el servidor tengan acceso a los datos intercambiados.

Cifrado en reposo

Para proteger la información en reposo, se propone el cifrado de datos en DynamoDB y S3 usando las claves de AWS KMS (Key Management Service).

En DynamoDB, el cifrado en reposo utiliza el servicio de AWS Key Management Service (KMS) para administrar las claves de cifrado, protegiendo automáticamente todos los datos almacenados en las tablas. De manera similar, S3 ofrece el cifrado en reposo a través de KMS, garantizando que los archivos almacenados (PDF recibidos, CDAs generados, etc.) estén cifrados antes de ser escritos en disco. En ambos casos, el cifrado protege la información de accesos no autorizados.

Si bien esta especificación está fuera del alcance del proyecto, se describe la forma óptima de implementación y se recomienda como punto importante para trabajo futuro.

AWS cloudTrail

Esta arquitectura también propone la habilitación de AWS CloudTrail para registrar todas las acciones realizadas en los recursos de AWS. Monitoreo de actividades sospechosas o no autorizadas, y generación de reportes de auditoría.

Gestión de secretos

La arquitectura propone AWS Secrets Manager para almacenar y gestionar de forma segura las claves de firma de documentos y otras claves sensibles que los repositorios puedan llegar a tener.

Manejo de información del repositorio

Los datos sensibles (como claves para firmar documentos) son separadas del resto de la información, lo cual permite tener un acceso restringido y controlado de estos datos.

Esto incrementa la seguridad, ya que mejora la confidencialidad y disponibilidad de la información clínica manejada por el sistema, dando mayores garantías a los usuarios y repositorios.

3.2.8. Mutilenguaje

El middleware soporta la generación de documentos clínicos con leyendas en español, inglés y portugués. DOCLUY se puede extender para soportar lenguajes adicionales.

3.2.9. Conclusiones

La arquitectura propuesta para la solución DOCLUY se destaca tanto por su capacidad de integración entre múltiples sistemas y generadores de información clínica, como su capacidad de procesamiento asíncrono. Al adoptar un enfoque basado en middleware, se garantiza la escalabilidad y flexibilidad necesarias para gestionar las comunicaciones entre dispositivos biomédicos, sistemas médicos y repositorios de información. Además, gracias al uso de colas de mensajes, este sistema brinda una rápida respuesta a las aplicaciones externas, siendo capaz de realizar tareas en forma asincrónica, y recuperarse a posibles errores de comunicación con los repositorios. Este enfoque no solo simplifica la integración inicial, sino que también facilita futuras extensiones y nuevas conexiones con otros dispositivos y plataformas.

El uso de infraestructura en la nube, particularmente con los servicios de Amazon Web Services (AWS), la consideramos una opción funcional tanto en términos de escalabilidad como de servicios. Los servicios utilizados por DOCLUY, como Amazon API Gateway, Lambda y DynamoDB, no solo permiten una gestión eficiente de los recursos, sino que también aseguran la trazabilidad, seguridad y disponibilidad de los datos clínicos generados y almacenados.

Adicionalmente, se ha diseñado una arquitectura modular que soporta diferentes formatos clínicos, permitiendo la flexibilidad necesaria para, en el futuro, enviar otros tipos de documentos clínicos además del formato CDA. Este enfoque modular, adaptable y asincrónico, posiciona a DOCLUY como una solución escalable, robusta y segura que puede satisfacer las crecientes demandas de interoperabilidad en el ámbito de la salud.

3.3. Implementación de DINABANG-CDA

La implementación de la solución diseñada se realizó de forma incremental, partiendo de un prototipo que contenía los componentes mínimos necesarios que permitían recibir información, procesarla y almacenarla. Este prototipo consistía en recibir información a través de una API, almacenarla en DynamoDB, enviar información a través de una cola SQS y pasar por distintas etapas de procesamiento realizadas por funciones Lambda orquestadas por un Step Function.

Con base en este prototipo, se definieron las tareas necesarias para completar la solución diseñada en la sección 3.2. El desarrollo fue creciendo en forma iterativa hasta llegar a una solución que cumple con los objetivos planteados.

Se desarrolló la solución a la que llamamos DOCLUY, que se integra con múltiples sistemas generadores de información, no solo con DINABANG. Recibe

información a través de una API, genera un documento CDA, lo firma electrónicamente, genera la transacción ITI-41 necesaria para enviar el CDA a HCEN y envía el mensaje MTOM SOAP para almacenar el CDA en el Repositorio XDS local y registrarlo en el Registro XDS local. La API gestiona autenticación y autorización a través de API Keys.

En el Anexo L se define la estructura de la API y en el Anexo M se encuentra un ejemplo de CDA y transacción ITI-41 generados por la solución.

La solución es asíncrona y escalable, permitiendo que se pueda integrar fácilmente nuevos repositorios clínicos para almacenar otro tipo de documentos distintos a CDA. Por ejemplo, se podría agregar a la solución el procesamiento necesario para que se genere un documento FHIR y se envíe a un FHIR Server sin afectar la solución actual de procesamiento de CDA y envío a HCEN.

DOCLUY además, registra y maneja errores notificando al generador de información cuando ocurre un error a través de un webhook (es una forma de comunicación entre aplicaciones en la que un servicio envía datos automáticamente a otro en respuesta a un evento específico) y a los administradores de DOCLUY a través de una notificación de la aplicación de mensajería Slack. Adicionalmente, DOCLUY provee una API para consultar el estado de un documento que está siendo o fue procesado por la solución. También registra logs y trazas durante todo el procesamiento, permitiendo observabilidad e identificar errores de forma precisa.

DOCLUY también implementa los requerimientos no funcionales definidos en la primera parte de este capítulo.

Capítulo 4

Pruebas y Simulaciones

Para garantizar que la implementación efectivamente realizada está acorde con los objetivos y requerimientos y del proyecto; definimos, implementamos y ejecutamos un completo conjunto de casos de pruebas. En este capítulo veremos los casos de pruebas creados y la metodología de ejecución.

4.1. Configuración del entorno de pruebas

Uno de los desafíos claves del proyecto es la integración con HCEN. Para validar la correcta implementación de esta conexión, se creó una réplica de un servidor XDS en AWS. Así mismo, se creó en nuestra infraestructura de AWS un ambiente de testing, el cual consta de una instancia completa de la solución aislada de las instancias de desarrollo y de producción. Dado que nuestra solución es accesible mediante API Rest (no tiene una interfaz de usuario), la mayoría de los casos de prueba se realizaron a través de la herramienta Postman (POSTMAN, 2024). Adicionalmente, para facilitar la validación de la creación de documentos clínicos con el servidor XDS, construimos una aplicación web que se conecta al servidor XDS y provee un entorno visual donde se puede listar y visualizar los documentos recibidos. En el Anexo N se muestra la aplicación desarrollada.

Adicionalmente, se realizaron pruebas de integración a través de una versión de la aplicación de DINABANG conectada a nuestro ambiente de testing.

En la figura 4.1 se puede ver un resumen de la arquitectura, donde se muestran las componentes y conexiones relevantes para el entorno de pruebas.

La instancia del servidor XDS fue instalada a partir de software provisto por AGESIC. Este servidor es parte de los elementos que provee Salud Digital a los prestadores de salud para que se integren con HCEN a través del Appliance Salud. El servidor XDS contiene un XDSBus, un Repositorio XDS local y un Registro XDS local. De esta manera pudimos emular correctamente la integración con el Appliance Salud a través del servidor XDS.

El entorno de testing creado asegura que cualquier aplicación o dispositivo

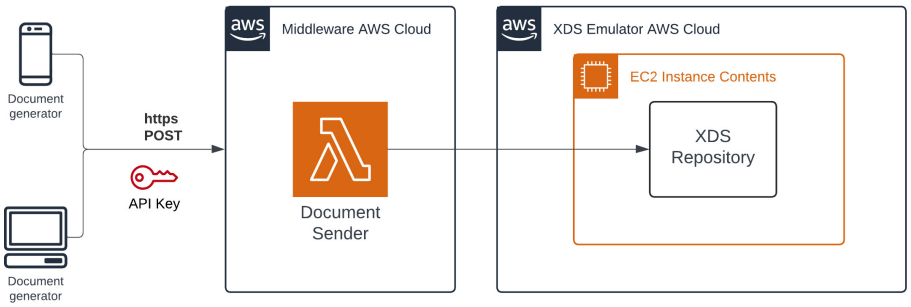


Figura 4.1: Entorno de pruebas DOCLUY.
Componentes y conexiones relevantes para el entorno de pruebas, las cuales incluyen un repositorio XDS propio

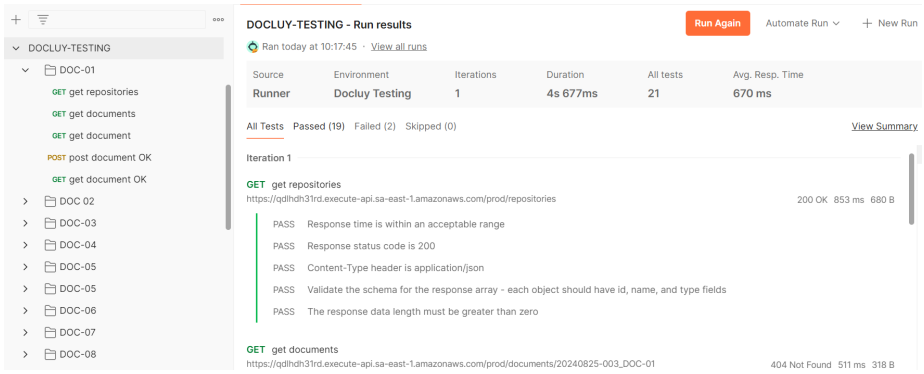


Figura 4.2: Interfaz de Postman.
Ejemplo de configuración y ejecución de casos de prueba

que consuma la API de DOCLUY se integre adecuadamente, minimizando los ajustes necesarios para el correcto funcionamiento en entornos productivos.

4.2. Automatización de pruebas

Los casos de pruebas diseñados, fueron ejecutados con la herramienta Postman. De esta manera pudimos correrlos de forma tanto manual como automática, y obtener un claro y completo reporte de errores. En la figura 4.2 se muestra un ejemplo de configuración y ejecución de casos de prueba.

4.3. Casos de prueba

4.3.1. Pruebas funcionales

Realizamos pruebas sobre la correcta funcionalidad del sistema. Estas pruebas abarcan tanto las operaciones básicas, como casos más complejos: envío de documentos y recuperación ante fallos de conectividad.

En la tabla 4.1 se muestran los casos de pruebas diseñados para probar el alcance de este proyecto. Cabe destacar que se construyeron diferentes repositorios y fuentes para llevar a cabo las pruebas.

Código	Descripción	Detalle
DOC-01	DOC OK	Se envía un documento correctamente al repositorio "Prestador de Salud A_OK" con la fuente "TESTING_1" y se consulta su estado. Se espera ver el documento correctamente enviado al repositorio indicado en formado CDA.
DOC-02	DOC Duplicado	Se envía un documento correctamente al repositorio "Prestador de Salud A_OK" con la fuente "TESTING_1" y luego se intenta enviar la misma información: igual source, igual id, igual versión. Se espera que el documento sea entregado al repositorio correspondiente una única vez. Se espera que el segundo intento de envío no efectué ninguna operación.
DOC-03	DOC OK A 2 REPOS	Se envía un documento correctamente al repositorio "Prestador de Salud A_OK" con la fuente "TESTING_1". Se envía el mismo documento correctamente al repositorio "Prestador de Salud B_OK" con la fuente "TESTING_1". Se espera que lleguen el documento a ambos repositorios.
DOC-04	DOC OK, REPO SIN CONEXIÓN	Se envía un documento correctamente al repositorio "Prestador de Salud C_NO" con la fuente "TESTING_1" y se consulta su estado. Se espera que el documento refleje el error correspondiente en el estado, entre en flujo de reintento y finalice con error de conectividad. Se espera recibir notificación en Slack.

Código	Descripción	Detalle
DOC-05	DOC OK, REPO SIN FIRMA	Se envía un documento correctamente al repositorio “Prestador de Salud D_NO” con la fuente “TESTING_1” y se consulta su estado. Se espera que el documento refleje el error correspondiente en el estado, entre en flujo de reintento y finalice con error de conectividad. Se espera recibir notificación en Slack.
INVALID-DOC-01	Autor Inválido	Se envía información de un reporte con un campo de autor inválido al repositorio “Prestador de Salud A_OK” con la fuente “TESTING_1”. Se consulta su estado. Se espera que el documento refleje el estado de error correspondiente.
INVALID-DOC-02	Paciente Inválido	Se envía información de un reporte con información del paciente no válida al repositorio “Prestador de Salud A_OK” con la fuente “TESTING_1” y se consulta su estado. Se espera que el documento refleje el estado de error correspondiente.
INVALID-DOC-03	Archivo Inválido	Enviar información de un reporte sin archivo .pdf al repositorio “Prestador de Salud A_OK” con la fuente “TESTING_1” y se consulta su estado. Se espera que el documento refleje el estado de error correspondiente.
INVALID-DOC-04	Metadatos inválidos	Enviar información de un reporte con información del documento no válida al repositorio “Prestador de Salud A_OK” con la fuente “TESTING_1” y se consulta su estado. Se espera que el documento refleje el estado de error correspondiente.
INVALID-DOC-06	Ejes inválidos	Enviar información de un reporte con valores de ejes no válidos al repositorio “Prestador de Salud A_OK” con la fuente “TESTING_1” y se consulta su estado. Se espera que el documento refleje el estado de error correspondiente.

Tabla 4.1: Casos de prueba principales ejecutados para validar el correcto funcionamiento de DOCLUY.

4.3.2. Pruebas de integración

Se hicieron dos tipos de pruebas de integración. Por un lado, se integraron dos dispositivos (DINABANG Y SISENF) y, por otro lado, se integró con el emulador XDS. Para facilitar la validación de la integración con el servidor XDS, se utilizó la aplicación web desarrollada que se conecta al servidor XDS y provee un entorno visual donde se puede listar y visualizar los documentos recibidos. En el Anexo N se muestra esta aplicación.

Los casos de prueba funcionales, que involucran un único generador de documentos, fueron ejecutados desde la aplicación de DINABANG, y los que resultan en un envío correcto de documento, fueron validados con la aplicación de visualización de XDS. El resto se validó con el endpoint de trazabilidad, en el que se muestra el estado resultante de la operación solicitada.

4.3.3. Pruebas de autenticación y autorización

Nos aseguramos de que los mecanismos de autenticación y autorización estén configurados correctamente. En la tabla 4.2 se muestran las pruebas realizadas para verificar que los sistemas autenticados tengan acceso adecuado y que las restricciones de autorización se aplican correctamente.

Código	Descripción	Detalle
INVALID-SOURCE-01	Source NO Existe	Enviar documento al "Prestador de Salud A_OK" con la fuente "TESTING_INVALID". Se espera que devuelva 401, error de operación no autorizada.
INVALID-SOURCE-02	Source "A" a repo de "B"	Enviar documento al "Prestador de Salud A_OK" de fuente "TESTING_1" con la fuente "TESTING_2". Se espera que devuelva 401, error de operación no autorizada.

Tabla 4.2: Casos de prueba de autenticación y autorización.

4.3.4. Conclusiones

El proceso de pruebas fue fundamental para asegurar que la implementación cumpliera con los objetivos y requerimientos del proyecto. Se definió y ejecutó un conjunto completo de casos de prueba en Postman, abarcando tanto las funcionalidades clave como los escenarios críticos. La creación de un entorno de pruebas permitió realizar una validación de integración técnica de DOCLUY, lo que derivó en resultados positivos en la mayoría de los casos, y detectar errores de implementación que fueron corregidos. Este proceso de testing fue clave para identificar y corregir errores en etapas tempranas, permitiendo ajustes continuos que perfeccionaron el funcionamiento final de DOCLUY.

Capítulo 5

Conclusiones y trabajo futuro

En este capítulo se resumen los resultados alcanzados y dificultades encontradas durante el desarrollo del proyecto. Se reflexiona sobre los objetivos cumplidos y se sugieren mejoras y posibles extensiones de la solución. Se realiza una autocrítica de los logros alcanzados y se incluye información de la gestión del proyecto. El objetivo es proporcionar una visión integral del trabajo realizado a lo largo del proyecto y señalar posibles direcciones para continuarlo.

5.1. Gestión del proyecto

Para organizar y coordinar el trabajo de manera eficiente, utilizamos tableros Kanban en la herramienta Trello (Trello, 2024). Por cada objetivo planteado, construimos un backlog con todas las tareas identificadas, las cuales fueron priorizadas en reuniones semanales, asegurando que siempre estuviéramos enfocados en las actividades más críticas para el avance del proyecto.

Durante la etapa de desarrollo, adoptamos una estrategia basada en sprints de una a dos semanas de duración. Esta metodología nos permitió mantener un enfoque iterativo y adaptable, asegurando que los ajustes necesarios pudieran hacerse a medida que avanzaba el desarrollo.

La priorización y organización de las tareas se llevaban a cabo en sesiones dedicadas exclusivamente a la gestión y seguimiento del proyecto, realizadas semanal o quincenalmente. Estas sesiones eran esenciales para evaluar el estado de las tareas y reasignarlas si era necesario.

El equipo está compuesto por tres personas, cada una de las cuales desempeñó distintas tareas. Todos los miembros participaron en actividades de investigación, diseño de arquitectura y documentación, pero en la fase de implementación hubo una separación de tareas diferenciada entre los integrantes para permitir una mayor especialización y eficiencia en la ejecución de tareas.

Esfuerzo total realizado 2382 horas

Para ilustrar cómo se distribuyó el tiempo dedicado al proyecto, se presenta en la figura 5.1 la distribución del esfuerzo a lo largo de los meses; y en la figura 5.2 se presenta la cantidad de horas dedicadas a cada tipo de actividad, como investigación, diseño, desarrollo, documentación, entre otras. Estas imágenes permiten ver cómo se organizó y ejecutó el trabajo a lo largo del proyecto.

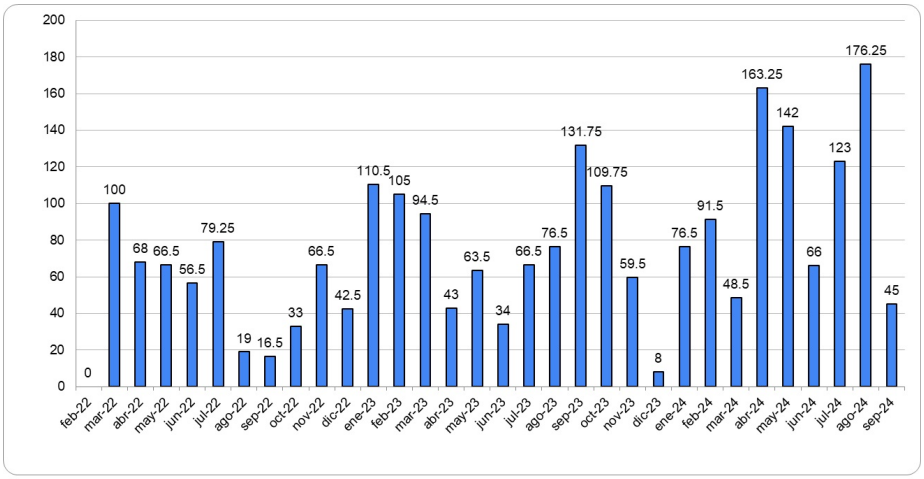


Figura 5.1: Esfuerzo total del equipo a lo largo de los meses.

A continuación, se detalla el desarrollo del proyecto a través de los hitos más importantes, los cuales marcan su evolución en el tiempo.

Comienzo de investigación (Marzo 2022)

Comenzamos con la investigación de documentación de HCEN generada por Salud Digital y búsqueda de información de proyectos y productos similares al objetivo inicial del proyecto.

Capacitación inicial DINABANG (Marzo 2022)

Participamos de una instancia de capacitación del dispositivo clínico DINABANG con motivo de una práctica de estudiantes a cargo del profesor Darío Santos. Se realizaron distintos test de fuerza en miembros inferiores. Nos permitió tener una primera aproximación a DINABANG desde el punto de vista práctico.

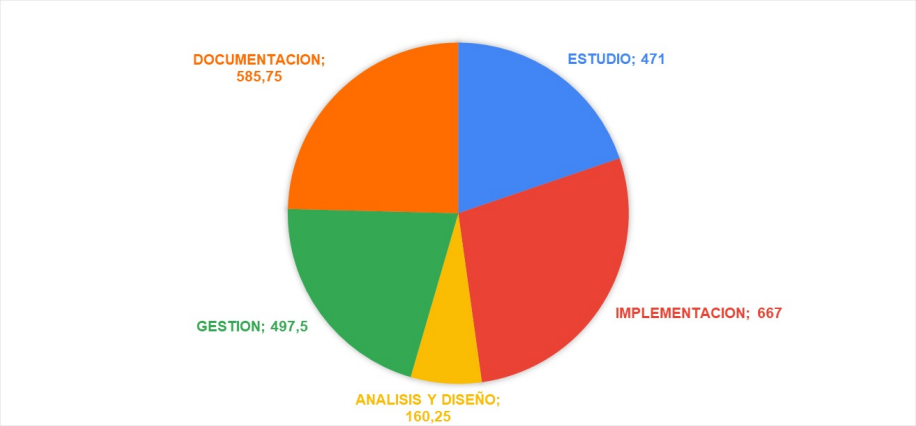


Figura 5.2: Esfuerzo total del equipo dedicado a cada actividad.

Curso "de rodilla" (Marzo 2022)

Participamos en el curso *Análisis 3D del movimiento de la rodilla para rehabilitación, evaluación perioperatoria y medicina deportiva*. Tuvimos la oportunidad de conocer en profundidad el uso del dispositivo clínico DINABANG y su aplicación práctica.

Reuniones para conocer uso detallado de DINABANG (Abril 2022)

Comienzo de reuniones con tutores e integrantes de la empresa MOVI para conocer en más detalle el uso de DINABANG y de qué forma nuestro proyecto podía agregar valor.

Propuesta de colaboración con Hospital Británico (Abril 2022)

Dado que existía la posibilidad de integrar nuestra solución con HCEN a través del prestador de salud Hospital Británico, iniciamos una propuesta de colaboración. Sin embargo, los resultados no fueron positivos y no pudimos asegurar la cooperación con esta institución. De nuestra parte, se realizaron todos los esfuerzos posibles, pero ante la falta de éxito, nos vimos obligados a explorar otras alternativas para cumplir con los objetivos establecidos.

Decisión de enfoque del proyecto (Junio 2022)

Hasta este momento, el proyecto tenía dos objetivos principales. El primero, generar un complemento de DINABANG que generara reportes de evolución y seguimiento con la información tomada por el dispositivo en sesiones anteriores. El segundo, era generar, tanto con los reportes de sesiones individuales como con los reportes de evolución, documentos CDA que luego se enviaran a la HCEN.

Luego de unos meses dedicados a investigación de HCEN, los aspectos técnicos de DINABANG y otros proyectos desarrollados por el NIB (NIB, 2024), observamos lo siguiente.

En primer lugar, vimos que un requisito recurrente en los proyectos del NIB era que se solicitara la generación documentos CDA con información de los sistemas que se pedían desarrollar y que luego se integraran a HCEN. Estudiando los documentos de esos proyectos, vimos que en algunos casos se generaban los documentos CDA, pero la integración con HCEN se registraba como trabajo futuro.

En segundo lugar, luego de realizar varias reuniones con los integrantes de la empresa MOVI, entendimos que la generación de reportes de seguimiento consistía en recopilar información de sesiones previas y crear una nueva forma de presentarla.

Es por esto, que decidimos enfocarnos primero en la generación de un documento CDA y su envío a HCEN, antes que en la generación de reportes de seguimiento, ya que el primer objetivo representaba un mayor desafío y mayor tiempo de investigación, desarrollo y pruebas.

Nuestro objetivo pasó a ser la implementación de una solución que permita generar documentos CDA y los envíe a HCEN no solo con información de DINABANG, sino que permitiera la integración de múltiples sistemas y dispositivos. De esta forma, no solo resolvíamos la generación de CDA e integración con HCEN para de DINABANG, sino que brindábamos una solución que pudieran utilizar otros proyectos del NIB para resolver su integración con HCEN. De esta forma, los futuros proyectos podrían centrarse en otros aspectos a resolver y no invertir tiempo en investigación de HCEN y generación de CDA, tiempo que ya estábamos invirtiendo nosotros.

En este momento, también decidimos definir una arquitectura que se adapte fácilmente a la incorporación de generación de documentos en distintos formatos clínicos. De esta forma, la solución permitiría agregar fácilmente la generación de documentos en nuevos formatos sin afectar los componentes existentes.

Propuesta de colaboración con CRAMI (Julio 2022)

Continuando con el objetivo de establecer un entorno de pruebas real para evaluar el funcionamiento de nuestro producto, realizamos gestiones con el prestador de salud CRAMI.

A pesar de nuestros esfuerzos, no fue posible lograr una colaboración efectiva con CRAMI. Si bien hubo una amplia disposición por el departamento técnico de CRAMI, no pudimos elevar la propuesta a FEPREMI, ya que todo su sistema de historia clínica está integrado y administrado por otra empresa de software. Esta estructura operativa, y la madurez de nuestro proyecto en este punto, impidió la integración con CRAMI. Aunque la colaboración no pudo materializarse debido a estas limitaciones, hemos utilizado esta experiencia para reflexionar sobre nuestros enfoques y buscar alternativas que nos permitan avanzar en la validación y mejora de nuestro producto.

Propuesta de colaboración con Hospital de Clínica (Julio 2022)

El Hospital de Clínicas fue considerado como alternativa para llevar a cabo las pruebas de integración con un prestador de Salud. Sin embargo, no pudimos asegurar la participación de esta institución en nuestro proyecto. De nuestra parte, se realizaron todos los esfuerzos posibles para lograrlo, pero ante la falta de éxito, nos vimos obligados a buscar alternativas para cumplir con los objetivos establecidos.

Reunión con Salud Digital (Agosto 2022)

Se realizaron gestiones formales con la dirección de Salud Digital para lograr un entorno de pruebas. Dicha solicitud fue evaluada y denegada.

Ingeniería de muestra (Octubre 2022)

Participamos en la edición 2022 de Ingeniería de Muestra. Durante este evento, presentamos nuestro proyecto. Esta experiencia no solo nos permitió compartir nuestras ideas y soluciones con otros profesionales del campo, sino que también nos brindó la oportunidad de recibir retroalimentación directa de expertos y colegas. A lo largo de la jornada, pudimos conectar con personas afines a nuestra área de estudio y ampliar nuestra perspectiva y enfoque.

El póster presentado en el evento se puede ver en el Anexo Ñ.

Descartar colaboración de prestador de salud (Noviembre 2022)

Tras descartar las opciones del Hospital Británico, CRAMI y el Hospital de Clínicas, decidimos no insistir en la búsqueda de un prestador de salud y optamos por enfocarnos en crear un entorno de pruebas utilizando el componente XDS proporcionado por AGESIC. Este componente es el que utilizan los prestadores de salud en entornos productivos. Dicha pieza de software está disponible en el sitio web de AGESIC de manera pública.

Presentación FHIR (Noviembre 2022)

El equipo de proyecto participo en una presentación de FHIR a cargo de Lucia Grundel, entendemos que fue una instancia muy provechosa que nos permitió afirmar muchos conceptos, por ejemplo que nuestra arquitectura contemplara la ampliación a formatos de salida diferentes a CDA. Nuestra arquitectura debía ser flexible y escalable.

Propuesta de arquitectura (Diciembre 2022)

Se realizó una presentación de la primera versión de arquitectura ante el tutor Antonio López. En esta instancia obtuvimos una retroalimentación muy positiva y la consideramos clave en el desarrollo de nuestro producto, puesto que Antonio nos dio su punto de vista sobre la arquitectura y nos señaló oportunidades para mejorar la asincronía de la solución.

Reunión Pablo Orefice y Lucía Grundel(Diciembre 2022)

Participamos de una instancia interesante donde Lucia y Pablo nos dieron sus visiones respecto al contexto tecnológico en el que se desarrollaba nuestro proyecto.

Reunión Gastón Milano sobre arquitectura en la nube (Enero 2023)

Gastón Milano participó en la implementación del sistema de agenda de vacunación COVID, y fue el responsable de su reformulación, ya que este sistema tenía un enfoque sincrónico y había colapsado por el volumen de peticiones. Obtuvimos información de primera mano sobre el caso de éxito que fue llevar ese sistema a un enfoque asíncrono con AWS. En esta instancia, Gastón nos dio una perspectiva detallada del enfoque elegido, información que se incluye como antecedentes en este proyecto.

Integración local con XDS (Creación, registro y consulta de CDA en XDS) (Enero 2023)

Se logró consumir exitosamente las operaciones brindadas por el XDS. Este hito fue clave en la validación técnica y nos permitió confirmar la posibilidad de conectarnos a un entorno de producción. De esta manera logramos garantizar que los CDA generados son perfectamente recibidos y alojados en la solución de software que brinda AGESIC a los prestadores de salud.

Ajuste de alcance del proyecto (Enero 2023)

Con una primera versión de la arquitectura y una estimación del trabajo a realizar, se decidió reevaluar el alcance del proyecto. Se tuvo en cuenta que se mejoraron los requerimientos iniciales en cuanto a dispositivos y repositorios soportados, lo cual implicaba una mayor carga de trabajo.

Se inició la conversación con los tutores del proyecto y con MOVI sobre la alternativa de ajustar el alcance. Finalmente se decidió remover la generación de reportes de evolución de DINABANG, para enfocar los esfuerzos en la integración con HCEN y en el desarrollo de una solución escalable que, a futuro, facilite la generación de otros tipos de documentos clínicos. Esta decisión se tomó porque entendemos que podemos aportar un mayor valor al concentrar nuestros esfuerzos en el desarrollo de esta solución escalable en lugar de en la creación de los reportes de seguimiento.

De esta forma, los reportes de evolución los desarrollará MOVI.

Definición de arquitectura (Enero 2023)

Presentamos a los tutores la segunda versión de arquitectura, la cual se expone en este proyecto y que optimiza los recursos y beneficios de los servicios sin servidor.

Capacitación en AWS (Enero 2023)

Comienzo de capacitación de desarrollo en AWS por parte del equipo.

Contacto con equipo Técnico de Salud Digital (Febrero 2023)

Logramos un contacto fluido con Juan Tabarez y Sabrina Sellanes, integrantes del equipo técnico de Salud Digital. Gracias a estas comunicaciones obtuvimos la información faltante que nos permitió la correcta instalación del XDS en nuestra infraestructura.

XDS en infraestructura en AWS (Febrero 2023)

Conseguimos implementar una instancia del XDS versión 5.0, la cual es exactamente igual a la utilizada por los prestadores de salud en su integración con Salud Digital. Si bien lo habíamos logrado en entornos locales, en este punto logramos implementarlo sobre AWS. Este logro garantiza la posibilidad de la integración de nuestro sistema con HCEN a través de la red de Salud.

Validación técnica en AWS (Febrero 2023)

Se consiguió realizar validaciones técnicas de los componentes que considerábamos clave para garantizar el funcionamiento de nuestra solución.

Creación de prototipo con componentes AWS (Marzo 2023)

Se crearon y conectaron en AWS todas las componentes mínimas necesarias para completar una tarea de punta a punta, es decir, desde el punto en el que se recibe información clínica, hasta enviar el documento clínico correspondiente a nuestro emulador XDS. Este prototipo es la base sobre la cual construimos la solución. En los siguientes meses nos centramos en agregar componentes y funcionalidades para llegar a la solución diseñada en el capítulo anterior.

Inicio de contacto con SISENF (Julio 2023)

Primer contacto con el equipo del proyecto de grado SISENF para que puedan realizar el envío de un CDA a HCEN a través de nuestra solución.

Pruebas de integración con MOVI (Setiembre 2023)

Elaboramos y compartimos con MOVI documentación para realizar la integración de envío de reportes de la aplicación a HCEN con DOCLUY. En esta etapa comenzamos las pruebas de envíos de reportes desde la aplicación DINABANG para generar CDA y enviarlos a HCEN.

Integración con SISENF (Noviembre 2023)

Se consiguió una integración exitosa con SISENF. Para ello, mantuvimos reuniones técnicas con el equipo del proyecto SISENF y se les proporcionó documentación de la API. Fue la segunda integración exitosa de información clínica transformada en CDA y enviada a nuestro emulador XDS. Ver anexo O

Presentación de solución a tutores (Abril 2024)

Se realizó una presentación a los tutores del proyecto para mostrar avances y definir dónde cerrar el ciclo de desarrollo de funcionalidades de la solución.

Pruebas de integración con MOVI (Mayo 2024)

Se consiguió una integración exitosa con DINABANG, ver anexo P. Para ello, la empresa MOVI realizó desarrollos de integración necesarios desde su aplicación en una versión beta. Invocando la API de DOCLUY, se logra que un estudio clínico en PDF se transforme en un documento clínico con el formato CDA versión 1 y quede alojado en el componente de software XDS.

Desarrollo de interfaz de usuario para pruebas y demo (Mayo 2024)

Se desarrolló una interfaz de usuario web simple para poder visualizar los CDA generados por la solución de la misma forma que se muestran en HCEN. Se implementan las operaciones del XDS de consulta de metadata de documentos de un paciente y consulta de un CDA.

Demo a tutores (Mayo 2024)

Se realizó por primera vez una demostración completa del sistema funcionando en tiempo real. Se mostró la creación de un reporte en la aplicación de DINABANG, el envío del mismo a nuestro sistema, la generación del CDA y la recepción en nuestro emulador XDS.

Comienzo de documentación (Junio 2024)

Repasamos el avance del proyecto revisando qué se implementó hasta el momento y qué quedaba pendiente y se definió, junto con el tutor, finalizar el desarrollo en el estado actual. Los aspectos que se habían diseñado, pero aún no implementado, se documentarían en la memoria del proyecto como sugerencia para la continuación del trabajo.

Unificación de documentación (Junio 2024)

Durante el transcurso del proyecto fuimos documentando los puntos y diagramas más importantes en documentos parciales. En esta fecha reunimos la documentación en un documento único, dándole el formato requerido para la

entrega del mismo. La realización de este documento se convirtió en el foco principal.

Implementación de observabilidad transversal a la solución (Julio 2024)

Se incorpora a todos los componentes de la solución el registro de logs y trazas para identificar errores ocurridos en el procesamiento de la solución.

Integración de manejo de errores (Julio 2024)

Se completa la integración de flujo de manejo de errores en la totalidad de la solución.

Liberación final (Agosto 2024)

Se liberó la última versión del sistema, que agrega controles de unicidad, manejo de errores y firma de documentos CDA.

Primera ejecución completa y exitosa de casos de prueba (Agosto 2024)

Se ejecutaron todos los casos de prueba desarrollador para testear la solución, habiendo corrido cada uno de ellos en forma exitosa por primera vez.

5.2. Dificultades encontradas

Durante el desarrollo de este proyecto nos encontramos con diferentes desafíos, tanto técnicos como no técnicos. En esta sección hacemos mención a los más relevantes.

Poca disponibilidad de prestadores de salud

El prestador de salud, inicialmente designado para realizar las pruebas de integración, no estuvo disponible a pesar de los esfuerzos realizados. Esto retrasó parte del análisis, ya que tuvimos que contactar otros prestadores, de los que igualmente no tuvimos mejor cooperación. Esto dio como resultado una inversión de horas no productivas, un retraso en el armado de la arquitectura y provocó una inversión de horas adicional en la construcción de un entorno de pruebas propio.

Aunque la falta de respuesta por parte de la mutualista inicial fue una dificultad inesperada, la rápida adaptación y búsqueda de alternativas como CRAMI permitieron continuar con el desarrollo del proyecto. Es importante destacar que entendemos las limitaciones de recursos que enfrentan estas instituciones para participar en iniciativas de este tipo.

En el caso de las mutualistas afiliadas a FEPREMI, se identificó que todas ellas unifican con salud digital en una sola plataforma. Esta integración plantea

un escenario diferente en términos de implementación y prueba de software, lo cual se tuvo en cuenta al evaluar las opciones alternativas.

Esta experiencia resalta la importancia de tener planes de contingencia y flexibilidad en proyectos que involucren la colaboración con instituciones externas. Además, subraya la necesidad de establecer comunicaciones claras y acuerdos específicos desde las etapas iniciales del proyecto.

Integración con Salud Digital

Durante el período de inicio de nuestro proyecto, Salud Digital atravesaba un proceso de cambios en la dirección de la entidad, lo que dificultó la canalización efectiva de nuestras solicitudes de acceso o creación de un ambiente de pruebas.

Dada la falta de avances con Salud Digital y colaboración con un prestador de salud, nuestra única alternativa era instalar un servidor XDS en nuestra propia infraestructura, emulando así la infraestructura que tendría un prestador de salud, pero sin la posibilidad de que los documentos generados llegaran al entorno de testing de HCEN.

Errores levantando ambientes locales

Tuvimos varias dificultades para poder levantar en entornos virtuales el software XDS que AGESIC brinda en su página. La versión de libre descarga, que se obtiene siguiendo los pasos del manual de instalación, no funcionaba correctamente, ya que estaba desactualizada.

Para superar estos obstáculos, contactamos un equipo técnico de AGESIC, con el cual logramos establecer un canal de comunicación fluido. Este canal resultó ser de gran ayuda, permitiéndonos obtener acceso a la pieza de software XDS en su última versión, alojada en la plataforma de Amazon. La implementación de este canal técnico facilitó la comunicación directa y eficiente, permitiéndonos avanzar en nuestro proyecto.

Sin acceso a INUS

Dentro de las restricciones encontradas en la integración con Salud Digital, estuvo la imposibilidad de acceder al Índice Nacional de Usuarios. No pudimos tener acceso a un INUS de testing de HCEN ni a una alternativa para emular la integración.

Capacitación en tecnologías

El cien por ciento del equipo tuvo que capacitarse en AWS para poder diseñar y desarrollar la arquitectura que se presenta en este documento.

Además, parte del equipo tuvo que capacitarse en node.js y bases de datos no relacionales.

5.3. Autocrítica

Este proyecto presentó un gran espacio de aprendizaje. En esta sección exponemos algunas cuestiones y métodos que entendemos podríamos haber resuelto de una mejor manera, las cuales tomamos como oportunidades de aprendizaje que tendremos presentes en la ejecución de trabajos futuros.

5.3.1. Definición del ambiente de trabajo

Conociendo todas las dificultades e inversión de tiempo a la hora de intentar conectar nuestra solución con un prestador de salud, entendemos que habría sido mejor optar desde un principio por levantar un ambiente propio para el XDS. También entendemos que, podríamos haber iniciado el desarrollo de esta manera, y haber realizado las conversaciones con los prestadores de salud en una etapa más madura del proyecto, esto quizás nos habría dado una mayor credibilidad sobre el potencial de nuestra solución y despertado mayor interés en los prestadores.

5.3.2. Períodos de baja dedicación

Por diferentes contratiempos laborales y personales que se fueron dando a lo largo del proyecto, no pudimos aportar las horas semanales y/o mensuales que habíamos estipulado inicialmente. Esto dio como resultado una extensión en el tiempo total de realización del proyecto.

5.3.3. Intentos con Prestadores

Dedicamos mucho tiempo en lograr un prestador de salud para desarrollar nuestro proyecto. Hemos discutido en el equipo que quizás luego de no lograr trabajar con el Hospital Británico ya haber intentado la simulación. Los tres estamos de acuerdo en que fue un aprendizaje que debiamos pasar, nunca hubo un no explicito y eso alargo los tiempos. Luego el trato con CRAMI nos aportó información de que otra manera no la habríamos logrado.

5.4. Conclusiones

DOCLUY, además de resolver el problema principal, proporciona una solución técnica que permite a pequeñas entidades integrar la información generada con la HCEN y otros repositorios futuros. Esto representa un avance significativo hacia la disponibilización de la información clínica, brindando a pequeñas entidades como SISENF y dispositivos de medición como DINABANG la oportunidad de contribuir a la mejora de la atención integral del paciente. De esta forma, se promueve una mayor interoperabilidad y accesibilidad de los datos clínicos, facilitando una mejor asistencia global.

La solución propuesta en este documento cumple con todos los objetivos definidos en el alcance final del proyecto. El prototipo DOCLUY, es una implementación funcional del diseño propuesto que cumple con los objetivos definidos por el proyecto.

Ampliamos el valor del proyecto con la decisión de remover del alcance inicial la generación de reportes de evolución de DINABANG, enfocándonos en el diseño y desarrollo de una solución que pueda integrarse con múltiples dispositivos y generar documentos clínicos según estándares requeridos por Salud Digital. No solo implementamos una solución que se integra con HCEN, sino que puede extenderse sin mayor complejidad para implementar otros estándares de documentación clínica. Entendemos que, de esta forma, agregamos más valor al producto final que si nos hubiésemos enfocado en los reportes de evolución.

Gracias a la metodología iterativa incremental que utilizamos para el desarrollo de DOCLUY, logramos implementar las funcionalidades y aspectos que agregan más valor al producto obtenido o representan un mayor desafío técnico.

Además de construir la solución, se logró alcanzar dos casos de éxitos con la conexión efectivamente DINABANG y SISENF. Esto muestra y confirma el potencial de la solución, además de su correcta implementación.

5.5. Futuro del proyecto

El primer paso para continuar con este proyecto en el aspecto de integración con HCEN es poder acceder a un ambiente de pruebas de un prestador de salud o de Salud Digital. Esto permitiría poder consumir el servicio del Appliance Salud que procesa la transacción ProvideAndRegisterDocumentSet (ITI-41) en lugar de consumir el endpoint del Repositorio XDS local de pruebas provisto por AGESIC. De esta forma se podría detectar si se deben realizar ajustes en la implementación de la transacción o la generación del CDA.

Durante el proceso de ejecución del proyecto, hemos observado una clara tendencia a nivel internacional hacia la adopción de FHIR. Es el estándar emergente para el intercambio de documentos clínicos, marcando un cambio significativo en comparación con el uso previo del formato CDA. En respuesta a esta evolución, nuestra plataforma de interoperabilidad se ha diseñado para admitir diversos formatos de salida, con especial énfasis en FHIR, considerándolo la implementación de conexiones a repositorios FHIR como el próximo paso en la evolución de este sistema.

De igual manera, se propone anexar, como método de envío, el envío de los reportes por medio de email. Este módulo permite una integración con receptores de menor porte, incluyendo el propio paciente.

Adicionalmente, para garantizar la disponibilidad y robustez del sistema y optimizar todo lo que se ha desarrollado, se recomienda, como futuros pasos, la implementación de los siguientes aspectos diseñados de la solución que son importantes y no fueron implementados:

- Políticas de acceso restringido a los buckets de S3.

- Gestión de versiones y eliminación segura en S3.
- Definición de métricas en Amazon CloudWatch para monitorear el rendimiento del sistema y alertas en base a esas métricas.
- Asignación de roles y permisos específicos para los API Keys para limitar el acceso a funcionalidades.
- Cifrado de datos en DynamoDB y S3, como se describe en la sección 3.2.7.
- Uso de claves de AWS Key Management Service (AWS KMS).
- Uso de AWS CloudTrail para registrar todas las acciones realizadas en los recursos de AWS.
- Uso de AWS Secrets Manager para almacenar y gestionar las claves API y otras credenciales de manera segura.
- Rotación automática de claves y credenciales para minimizar el riesgo de exposición.
- Almacenamiento separado de datos sensibles, (como claves para firmar documentos), como se describe anteriormente en la sección de seguridad de la solución.

DOCLUY está preparado para escalar tanto en infraestructura como en integración, con una alta capacidad de escalabilidad que permite no solo construir sobre lo que se plantea como trabajo futuro, sino también agregar nuevas funcionalidades y adaptarse a futuros desafíos. En un entorno tecnológico en constante evolución, DOCLUY está diseñado para seguir creciendo y respondiendo de manera ágil a las demandas cambiantes, asegurando que siga siendo una solución robusta y flexible a largo plazo.

Referencias

A. Stross-Radschinski. (2015). A programming language changes the world [Manual de software informático]. Montevideo, Uruguay.

AGESIC. (2021). *Con aws, uruguay pone sistema de solicitud de agenda de vacunación en operación en tiempo récord.* https://aws.amazon.com/es/solutions/case-studies/agesic-msp/.

AGESIC. (2023a). *Género v1.* https://centrodeconocimiento.agesic.gub.uy/documents/207224/425690/genero_v1.xml/ca38859d-e4e2-9f8a-4468-ce6e9e657304. (Accessed: 2023-09-22)

AGESIC. (2023b). *Unaoid.* https://unaoid.gub.uy. (Accessed: 2023-09-22)

AGESICl. (2023a). *Hcen.* https://www.gub.uy/ministerio-salud-publica/tramites-y-servicios/servicios/historia-clinica-electronica-nacional.

AGESICl. (2023b). *Hcen.* https://centroderecursos.agesic.gub.uy/web/arquitectura-salud.uy/inicio/-/wiki/Arquitectura+para+Salud/Arquitectura+Tecnolgica.

AGESICl. (2023c). *Salud digital.* https://www.gub.uy/agencia-gobierno-electronico-sociedad-informacion-conocimiento/politicas-y-gestion/programas/es-saluduy.

Aidbox. (2017). *Aidbox fhir platform.* https://docs.aidbox.app.

AMAZON. (2023). *Tecnología sin servidor.* https://aws.amazon.com/es/what-is/serverless-computing/.

AMAZON. (2024a). *Amazon s3.* https://aws.amazon.com/s3/.

AMAZON. (2024b). *Api gateway.* https://aws.amazon.com/api-gateway.

AMAZON. (2024c). *Aws lambda.* https://aws.amazon.com/lambda/.

AMAZON. (2024d). *Dynamodb.* https://aws.amazon.com/dynamodb/.

ANSI. (2023). *American national standards institute.* https://www.ansi.org/. (Accessed: 2024-08-26)

AWS. (2024a). *Aws native observability.* https://catalog.workshops.aws/observability/en-US/aws-native.

AWS. (2024b). *Aws sdk para go.* https://aws.amazon.com/es/sdk-for-go/.

AWS. (2024c). *Aws step functions.* https://aws.amazon.com/step-functions/.

AWS. (2024d). *Boto3 documentation.* https://boto3.amazonaws.com/v1/documentation/api/latest/index.html.

AWS. (2024e). *Colas de mensajes.* https://aws.amazon.com/es/message-queue/#:~:text=A%20message%20queue%20is%20a,once%2C%20by%20a%20single%20consumer.

AWS. (2024f). *Crear aplicaciones .net con aws.* https://aws.amazon.com/es/developer/language/net/solutions/.

AWS. (2024g). *.net en aws.* https://aws.amazon.com/es/developer/language/net/.

AWS. (2024h). *Servicio de bases de datos no relacionales: Amazon dynamodb.* https://aws.amazon.com/es/pm/dynamodb/.

AWS. (2024i). *¿qué es el middleware?* https://aws.amazon.com/es/what-is/middleware/#:~:text=El%20middleware%20act%C3%BAa%20como%20un,servicio%20unificado%20a%20sus%20usuarios. (Accessed: 2024-09-12)

AWS. (2024j). *¿qué es el procesamiento por lotes?* https://aws.amazon.com/es/what-is/batch-processing/#:~:text=El%20procesamiento%20por%20lotes%20es,repetitivos%20y%20de%20gran%20volumen.

AWS. (2024k). *¿qué es java?* https://aws.amazon.com/es/what-is/java/.

AWS. (2024l). *¿qué es python?* https://aws.amazon.com/es/what-is/python/.

AWS. (2024m). *¿qué es step functions?* https://docs.aws.amazon.com/es_es/step-functions/latest/dg/welcome.html. (Accessed: 2024-09-12)

C. Altamirano. (2023). *Integración fhir.* https://oa.upm.es/75029/1/tfg_carolina_altamirano_deakova.pdf.

D. Chappell. (2004). *Enterprise service bus.* O'Reilly.

Data Ingeneering. (2024). *¿cómo se decide cuándo utilizar el procesamiento por lotes o en tiempo real?* https://www.linkedin.com/advice/1/how-do-you-decide-when-use-batch-real-time-processing-h2rle#:~:text=Volume%20of%20Data%3A%20Batch%20Processing,without%20waiting%20for%20batch%20intervals.

Equipo HCEN. (2020). Guia de configuracion xds [Manual de software informático]. Montevideo, Uruguay.

G. Bonilla, M. Rocanova. (2023). Sistema informático enfermería [Manual de software informático]. Montevideo, Uruguay.

HCEN. (2024). *Snis.* https://www.youtube.com/watch?v=ewc9W3Vp63M.

HL7. (2023a). *External code systems.* https://terminology.hl7.org/external_terminologies.html. (Accessed: 2023-09-22)

HL7. (2023b). *Hl7standar.* https://www.hl7.org/implement/standards/product_brief.cfm?product_id=7.

HL7. (2024). *Hl7 reference information model.* https://www.hl7.org/implement/standards/rim.cfm. (Accessed: 2024-08-26)

HL7 Internacional. (2023). *Hl7.* https://www.hl7.org/about/index.cfm?ref=nav.

HL7 International. (2023). *Hl7products.* https://www.hl7.org/implement/standards/product_brief.cfm?product_id=186.

IBM. (2021). *Hora universal coordinada.* Horauniversalcoordinada. (Accessed: 2024-09-06)
IBM. (2024a). *¿qué es el teorema cap?* https://www.ibm.com/mx-es/topics/cap-theorem.
IBM. (2024b). *¿¿qué es una base de datos nosql?* https://www.ibm.com/es-es/topics/nosql-databases.
IHE. (2023). *Xds.* https://profiles.ihe.net/ITI/TF/Volume1/ch-10.html.
International Organization for Standardization. (2023). *Iso 8601:2004.* https://www.iso.org/standard/40874.html. (Accessed: 2023-09-22)
IT Procedure Template. (2020). *Batch processing vs. real-time processing.* https://www.it-procedure-template.com/batch-processing-vs-real-time-processing/.
J. Pascual. (2019). *Hl7 cda.* https://www.disrupciontecnologica.com/hl7-cda/.
Ley 18331. (2008). *Ley-18331.* https://www.impo.com.uy/bases/leyes/18331-2008.
Library of Congress. (2017). *Iso 639-2: Codes for the representation of names of language.* https://www.loc.gov/standards/iso639-2/php/code_list.php. (Accessed: 2023-09-23)
litslink. (2024). *Deciding between c# and python: Selecting the ideal language for your project.* https://litslink.com/blog/csharp-vs-python-choosing-right-language-for-your-project.
M. Villalba. (2023). *Qué arquitectura es mejor para mi aplicación? eventos o máquinas de estado?* https://www.youtube.com/watch?v=3UwgnYByOk8&ab_channel=Marciaylanube.
M. Wagner. (2023). *Point to point (direct) vs middleware connections.* https://docs.celigo.com/hc/en-us/articles/20085283685915-Point-to-point-direct-vs-middleware-connections.
Microsoft. (2024). *Learn to code c#.* https://dotnet.microsoft.com/en-us/learntocode.
MongoDB. (2024a). *Database. deploy a multi-cloud database.* https://www.mongodb.com/products/platform/atlas-database.
MongoDB. (2024b). *Sharding.* https://www.mongodb.com/docs/manual/sharding/.
MongoDB. (2024c). *¿qué es mongodb?* https://www.mongodb.com/es/company/what-is-mongodb.
MOVI. (2023). *Movi.* https://movitfl.com/home/.
MOZILLA. (2024). *Javascript.* https://developer.mozilla.org/en-US/docs/Web/JavaScript.
N. Freed, N. Borenstein. (2021). *Mime part one: Format of internet message bodies.* https://www.ietf.org/rfc/rfc2045.txt. (Accessed: 2023-09-23)
N. Hourcade, V. Coggan, E. Della Mea. (2020). *Sipome.* https://www.colibri.udelar.edu.uy/jspui/bitstream/20.500.12008/24357/1/HCD20.pdf.

NIB. (2024). *"proyectos de fin de carrera" (ingeniería eléctrica), "proyectos de grado" (ingeniería en computación), "monografÍas" (eutm y ciclo métodos cuantitativos, medicina).* http://www.nib.fmed.edu.uy/sitio_nib/proyectos/Proyectos_fin.html. (Accessed: 2024-08-30)

OpenmHealth. (2024). *Openmhealth.* https://www.openmhealth.org/.

ORACLE. (2024). *¿qué es una base de datos relacional (sistema de gestión de bases de datos relacionales)?* https://www.oracle.com/es/database/what-is-a-relational-database/.

POSTMAN. (2024). *Postman.* https://www.postman.com/.

R.Barboza, J. Dominguez, A. Fernandez. (2018). *Dinabang.* http://www.nib.fmed.edu.uy/sitio_nib/BibliotecaNIB/PublNIB238.pdf.

Regenstrief Institute. (2023). *Logical observation identifier names and codes (loinc): Oid 2.16.840.1.113883.6.1.* https://loinc.org/oids/2.16.840.1.113883.6.1/. (Accessed: 2023-09-22)

Regenstrief Institute. (2024). *Loinc.* https://loinc.org/. (Accessed: 2024-08-26)

Romain. (2023). *Why asynchronous event-driven frameworks are critical for real-time data processing.* https://medium.com/@romtref/why-asynchronous-event-driven-frameworks-are-critical-for-real-time-data-processing-8b3a3c77de5d.

Salud Digital. (2013). *Guía de implementación firma electrónica de documentos clínicos.* Montevideo, Uruguay.

Salud Digital. (2017a). *Guía para la gestión de oid.* Montevideo, Uruguay.

Salud Digital. (2017b). *Tabla de identificadores mínimos de usuario y de autor.* Montevideo, Uruguay.

Salud Digital. (2019). *Guía de implementación cda mínimo hl7 v3 cda-r2.* Montevideo, Uruguay.

Salud Digital. (2021a). *Guía técnica servicios receta digital nacional prestadores de servicios de salud.* Montevideo, Uruguay.

Salud Digital. (2021b). *Hoja de estilo para visualización de cda - version 3.0.* https://centrodeconocimiento.agesic.gub.uy/documents/207224/0/Hoja+de+estilo+CDA+-+Salud.xsl/8cf20caf-4bdd-b364-c935-d5730694401e. (Accessed: 2022-07-14)

Salud Digital. (2022). *Guía técnica de metadatos xds.* Montevideo, Uruguay.

Salud Digital. (2024). *Arquitectura de referencia hcen.* https://centroderecursos.agesic.gub.uy/web/arquitectura-salud.uy/inicio/-/wiki/Arquitectura+para+Salud/Arquitectura+Tecnol%C3%B3gica.

Slack. (2024). *Slack.* https://slack.com/intl/es-uy/help/articles/115004071768-%C2%BFQu%C3%A9-es-Slack.

SNOMED. (2024). *What is snomed ct?* https://www.snomed.org/what-is-snomed-ct. (Accessed: 2024-08-26)

Systems Design. (2024). *How can you use asynchronous processing to improve system scalability?* https://www.linkedin.com/advice/0/how-can-you-use-asynchronous-processing-improve-abkte?lang=en&originalSubdomain=es.

TechTarget. (2017). *Enterprise master patient index (empi).* https://www.techtarget.com/searchhealthit/definition/master-patient-index-MPI. (Accessed: 2024-09-02)

The Salesforce First. (2023). *Point-to-point vs middleware integration.* https://www.linkedin.com/pulse/point-to-point-vs-middleware-integration/.

TIBCO. (2023). *Que es hl7 fhir.* https://www.tibco.com/es/reference-center/what-is-hl7-fhir.

Trello. (2024). https://trello.com.

URSEC. (2022a). *Ursec 19/022.* https://www.gub.uy/unidad-reguladora-control-datos-personales/institucional/normativa/resolucion-n-19022-0.

URSEC. (2022b). *Ursec 42/022.* https://www.gub.uy/unidad-reguladora-control-datos-personales/institucional/normativa/resolucion-n-42022.

W3C. (2006). *Mtom.* https://www.w3.org/submissions/soap11mtom10/.

Glosario

AGESIC Agencia de Gobierno Electrónico y Sociedad de la Información y del Conocimiento. 12, 37, 47, 48, 52, 54

Amazon API Gateway Es un servicio totalmente gestionado que los desarrolladores pueden utilizar para crear, publicar, mantener, supervisar y proteger APIs a cualquier escala.. 12, 22, 23, 27

Amazon DynamoDB Servicio SQL sin base de datos totalmente gestionado que proporciona un rendimiento rápido y predecible con una escalabilidad perfecta.. 12, 22, 25

Amazon S3 Es un servicio de almacenamiento para Internet. Se puede usar para almacenar y recuperar cualquier cantidad de datos en cualquier momento y desde cualquier parte de la Web.. 12, 25

Amazon Single Queue Service Es una cola hospedada de confianza y escalable diseñada para almacenar mensajes mientras viajan de un equipo a otro.. 18, 22

ANSI American National Standards Institute. 3

API Application Programming Interface. 17

AWS Amazon Web Services. 10, 12, 81, 83, 85, 87

AWS Lambda Servicio web que se utiliza para ejecutar código sin provisionar ni administrar servidores. 12, 18, 22

AWS Step Function Step Functions se basa en máquinas y tareas de estados. Las máquinas de estados se denominan flujos de trabajo, que son una serie de pasos basados en eventos. Cada paso de un flujo de trabajo se denomina estado. (AWS, 2024m). 18

Bucket Un bucket es un contenedor para objetos almacenados en Amazon S3.. 27

C-CDA Consolidated Clinical Document Architecture. 13

CDA Clinical Document Architecture. xi, 1, 2, 4, 6, 12, 13, 15, 17, 20, 26, 28, 36

CTO Chief Technology Officer. 12

FHIR Fast Healthcare Interoperability Resource. 5, 13, 29, 36

HCEN Historio Clinica Electronica Nacional. 2, 6, 12, 15, 17, 29, 36

HL7 Health Level Seven. 3–6, 12, 13, 15

HTTP Hypertext Transfer Protocol. 23, 27

IHE Integrating the Healthcare Enterprise. 8

INOT Instituto Nacional de Ortopedia y Traumatologıa. 12

INUS Indice Nacional de Usuarios de Salud. 7

JSON JavaScript Object Notation. 13, 29

MTOM Message Transmission Optimization Mechanism. xiii, 16, 36, 111, 114, 133

NIB Núcleo de Ingeniería Biomédica. 10

PDF Portable Document Format. 25

RIM Reference Information Model. 3, 5

SDK Software Development Kit. 87, 88

SIPOME Sistema Informático de Patología Oncológica Músculo Esquelética. 11

SISENF Sistema Informatico Enfermeria. 2, 10, 67

Slack Aplicación de mensajería emrpesarial que prmite envío de mensajes a personas dentro y fuera de la organización. Las personas pueden trabajar en espacios dedicados a un tema en especial denominados canales.(Slack, 2024). 30, 36, 39, 40

URSEC Unidad Reguladora de Servicios de Comunicaciones. 10, 79, 80

UTC Coordinated Universal Time. 32, 33

XDS Registro Nacional de Documentos Clinicos. 7, 8

XDS.b Cross-Enterprise Document Sharing. 8

Anexos

Anexo A

Comunicación entre múltiples sistemas

Existen dos tipos de sistemas externos que debemos comunicar entre sí: los diferentes repositorios de documentos clínicos y los diferentes generadores de información.

Los repositorios de datos clínicos almacenan los documentos clínicos. Estos documentos pueden tener diferentes formatos y diferentes formas de comunicación al momento de recibir documentos. Tanto el formato como la forma de comunicación son relevantes en esta arquitectura, ya que tiene como cometido generar documentos clínicos en el formato específico de cada repositorio para hacer el posterior envío.

Los generadores de información son dispositivos clínicos (ej.: DINABANG), sistemas médicos de gestión (ej.: SISENF), así como clínicas y profesionales independientes (Doctores en Odontología, Licenciados en Nutrición, etc.). Estos tienen la capacidad de generar información clínica que deben ser enviados a los repositorios clínicos con el formato aceptado por cada uno.

A continuación, analizaremos las dos formas consideradas a la hora de resolver la comunicación entre múltiples sistemas: comunicación directa y middleware.

A.1. Comunicación directa

Esta forma de comunicación consiste en comunicar de manera directa cada generador de información con los correspondientes repositorios clínicos. Los generadores serán los encargados de transformar la información recabada al formato clínico requerido.

- Pros
    - Arquitectura simple.

- Menor latencia debido a la comunicación directa sin intermediarios.
- Menor cantidad de componentes a administrar.

- Contras
  - Escalabilidad costosa: cada nuevo repositorio o generador requiere nuevos desarrollos.
  - Mantenimiento complejo, ya que cambios en un repositorio pueden requerir actualizaciones en todos los generadores que se comunican con él.
  - Podría haber redundancia en la lógica de transformación, comunicación y seguridad.

A.2. Middleware

En esta opción, se utiliza un componente intermedio (middleware) que actúa como mediador entre los generadores de información y los repositorios. Este middleware recibe la información de los generadores, la transforma al formato requerido por los repositorios y maneja la lógica de comunicación.

- Pros
  - Centraliza la lógica de transformación, comunicación y seguridad.
  - Facilidad para agregar nuevos repositorios y generadores de información.
  - Cambios en la lógica de una transformación o repositorio solo requieren modificaciones en el middleware.
  - Evita redundancia en lógica de transformación y comunicación.

- Contras
  - Mayor complejidad inicial en la implementación.
  - Posible aumento en la latencia debido al pasaje adicional a través del middleware.(The Salesforce First, 2023)(M. Wagner, 2023)

Luego de evaluar las características que se muestran en la tabla A.1 de las dos alternativas, la mejor opción para este sistema es la implementación de un middleware para la comunicación entre los generadores de información y los repositorios clínicos. De esta manera se centraliza la lógica de transformación, comunicación, seguridad y se facilita el mantenimiento y la integración de nuevos sistemas.

Criterio	Comunicación Directa	Middleware
Facilidad de Implementación	Alta	Media
Latencia	Baja	Media
Escalabilidad	Baja	Alta
Mantenibilidad	Baja	Alta
Desacoplamiento	Baja	Alta
Complejidad Inicial	Baja	Media
Redundancia en Transformaciones	Alta	Baja

Tabla A.1: Comparación de alternativas de comunicación entre múltiples sistemas.

Esta tabla fue mostrada en sección 3.2.1 como la tabla 3.1, en dicha sección se explica en detalle el diseño de la arquitectura

Anexo B

Tipos de infraestructura

Para implementar el middleware, es necesario decidir si utilizar una infraestructura en la nube, una infraestructura propia, o aprovechar la infraestructura de los repositorios de documentos clínicos. Cada opción tiene sus ventajas y desventajas en términos de costos, mantenimiento, escalabilidad y seguridad.

B.1. Infraestructura en la nube

Esta opción se refiere a utilizar servicios en la nube proporcionados por proveedores como Amazon Web Services (AWS), Microsoft Azure, o Google Cloud.

- Pros
  - Escalabilidad: Fácilmente ajustable para manejar aumentos en la carga.
  - Alta disponibilidad y tolerancia a fallos: Servicios en la nube ofrecen redundancia y recuperación ante desastres.
  - Reducción de costos iniciales: No es necesario invertir en hardware y mantenimiento.
  - Actualizaciones y mantenimiento gestionados: El proveedor se encarga de actualizaciones de software y hardware.
  - Seguridad: Proveedores de nube ofrecen medidas de seguridad avanzadas y cumplimiento de normativas.
- Contras
  - Costos recurrentes: Pagos continuos basados en el uso.
  - Dependencia del proveedor: Dependencia de la infraestructura y políticas del proveedor.
  - Latencia: Posible latencia adicional dependiendo de la ubicación geográfica de los centros de datos.

B.2. Infraestructura propia

Implementar y gestionar el middleware en una infraestructura propia, dentro de las instalaciones del proyecto.

- Pros
  - Control completo sobre la infraestructura, configuraciones y políticas de seguridad.
  - Costos controlados.
  - Podría tener menor latencia por ubicación de infraestructura.
- Contras
  - Costos elevados para adquirir hardware y software necesarios.
  - Responsabilidad de gestionar actualizaciones y mantenimiento.
  - Es más complejo y costoso escalar la infraestructura.
  - Necesidad de implementar y gestionar mecanismos de redundancia y recuperación ante desastres.

B.3. Infraestructura de los repositorios

Utilizar la infraestructura existente de los repositorios de documentos clínicos para implementar y gestionar el middleware.

- Pros
  - Aprovechar la infraestructura existente podría reducir costos.
  - Podría facilitar la integración con los repositorios.
- Contras
  - Dependencia de múltiples repositorios con diferentes configuraciones y políticas.
  - Mayor dificultad para mantener y actualizar la infraestructura debido a la falta de homogeneidad.
  - La infraestructura de repositorios pueden no estar diseñadas para soportar la carga adicional.
  - Desafíos en la implementación de medidas de seguridad consistentes y el cumplimiento de normativas.

Tras evaluar las opciones, utilizaremos una infraestructura en la nube para implementar el middleware. Esta opción ofrece un buen equilibrio entre escalabilidad, disponibilidad, costos y seguridad. Las otras dos opciones pueden ser el mejor escenario en casos de integración con un único prestador de salud, o ante requisitos que la solución en la nube no pueda garantizar.

En la tabla B.1 se encuentra un resumen de los aspectos evaluados para decidir qué tipo de infraestructura utilizar.

Criterio	Infraestructura en la Nube	Infraestructura Propia	Infraestructura de Repositorios
Escalabilidad	Alta	Media	Baja
Disponibilidad y Tolerancia a Fallos	Alta	Media	Baja
Costos Iniciales	Bajos	Altos	Bajos
Costos Recurrentes	Medios	Bajos	Medios
Mantenimiento y Actualizaciones	Gestión por Proveedor	Autogestionado	Dificultoso
Control sobre la Infraestructura	Medio	Alto	Bajo
Latencia	Media	Baja	Alta
Seguridad y Cumplimiento	Alta	Alta	Variable

Tabla B.1: Comparativa de tipos de infraestructura consideradas.
Esta tabla fue mostrada en sección 3.2.1 como la tabla 3.2, en dicha sección se explica en detalle el diseño de la arquitectura

Anexo C

Procesamiento asincrónico

Una de las características más importantes de la solución, es permitir que los documentos sean procesados y almacenados en los repositorios, sin que los generadores de información tengan que esperar a que se complete el procesamiento. Esto es importante para manejar eficientemente los recursos y tiempos de espera, así como desacoplar la generación de documentos de su procesamiento y almacenamiento, mejorando la escalabilidad y la capacidad de manejar picos de carga.(Systems Design, 2024)

A continuación, presentaremos tres alternativas evaluadas para el procesamiento de información: procesamiento por lote, colas de mensajes y eventos.

C.1. Procesamiento en lote

Utilizar procesamiento por lotes permite que la información recibida por el middleware sea acumulada y se procese en lotes de forma periódica. De esta manera, los generadores de información pueden continuar realizando otras tareas sin necesidad de esperar una respuesta.(AWS, 2024j)

- Pros
  - Implementación simple.
  - Ideal para grandes volúmenes de datos que no requieren procesamiento en tiempo real.
- Contras
  - Latencia en el procesamiento.
  - No es adecuado para casos que requieren procesamiento inmediato.
  - La información no siempre está actualizada.
  - Posible acumulación de datos que puede llevar a picos de carga.(Data Ingeneering, 2024)(IT Procedure Template, 2020)

C.2. Colas de mensajes

En un sistema de colas de mensajes, la información es enviada a una cola y posteriormente consumida por el encargado de procesarla. Una vez procesados, los mensajes se eliminan de la cola. Las colas de mensajes se utilizan para desacoplar procesos, para proporcionar un búfer y suavizar picos de cargas de trabajo.

- Pros
  - Alta disponibilidad y tolerancia a fallos.
  - Desacoplamiento entre productores y consumidores de mensajes.
  - Capacidad para manejar grandes volúmenes de datos y picos de carga.
  - Escalabilidad.
- Contras
  - Necesidad de monitorear y asegurar la entrega de mensajes.
  - Latencia en el envío de mensajes.(AWS, 2024e)

C.3. Eventos

En este tipo de arquitectura, la información entre los distintos componentes se intercambia mediante eventos. Estos componentes generan y consumen los eventos de manera asíncrona.

- Pros
  - Desacoplamiento entre los componentes del sistema.
  - Escalabilidad.
  - Trazabilidad completa de los eventos y su procesamiento.
- Contras
  - Alta complejidad en la implementación y gestión.
  - Necesidad de garantizar la consistencia eventual.
  - Requiere una infraestructura robusta para el manejo de eventos.(Romain, 2023)(M. Villalba, 2023)

Tras evaluar las distintas opciones presentadas en la tabla C.1, definimos la implementación de colas de mensajes como mejor opción para el procesamiento asíncrono. Esta solución desacopla los componentes que participan en el procesamiento, permitiendo manejar eficientemente los picos de carga y optimizando la disponibilidad del sistema.

Criterio	Procesamiento en Lote	Colas de Mensajes	Microservicios y Eventos
Facilidad de Implementación	Alta	Media	Baja
Latencia	Alta	Baja	Baja
Escalabilidad	Media	Alta	Alta
Mantenibilidad	Alta	Media	Baja
Desacoplamiento	Baja	Alta	Alta
Complejidad Inicial	Baja	Media	Alta
Adecuado para Procesamiento en Tiempo Real	No	Sí	Sí
Gestión de Picos de Carga	Baja	Alta	Alta

Tabla C.1: Comparativa de alternativas de procesamiento asíncrono.
Esta tabla fue mostrada en sección 3.2.1 como la tabla 3.3, en dicha sección se explica en detalle el diseño de la arquitectura

Anexo D

Proveedores de infraestructura

Para seleccionar el proveedor de nube más adecuado para implementar el middleware, consideramos diversos factores, incluyendo el cumplimiento de normativas de protección de datos en Uruguay, costos, servicios ofrecidos y soporte técnico. A continuación se presenta una evaluación de los principales proveedores: AWS, Azure y Google Cloud.

D.1. Amazon Web Services (AWS)

- Cumplimiento Normativo:
  - AWS se ajusta a la legislación nacional sobre protección de datos personales según las resoluciones Nro. 19/022 y 42/022 de la URSEC.
  - AWS proporciona herramientas para cumplir con otras normativas internacionales relevantes como HIPAA, GDPR y más
- Servicios Ofrecidos:
  - Amplia gama de servicios gestionados. (Ej.: SQS, Step function)
  - Servicios avanzados de seguridad y encriptación.
- Costos:
  - Modelo de precios de pago por consumo de recursos.
  - Ofrece una capa gratuita para muchos servicios durante el primer año, lo que permite realizar pruebas y desarrollos iniciales con costos reducidos.
- Soporte Técnico:
  - Amplia documentación y recursos de soporte.

- Ofrece varios planes variando en niveles de servicios, asesoramiento, administración, etc.

D.2. Microsoft Azure

- Cumplimiento Normativo
  - No se encontraron sobre protección de datos personales respecto a Microsoft Azure. Se inició ante URSEC la consulta formal Nro. 2024-71-2671-005082
  - Ofrece herramientas para asegurar el cumplimiento de normativas internacionales como HIPAA, GDPR y más.
- Servicios Ofrecidos:
  - Servicios robustos como Azure Queue Storage para colas de mensajes, Azure Virtual Machines para computación, y Azure SQL Database para bases de datos.
  - Seguridad avanzada y servicios de gestión de claves como Azure Key Vault.
- Costos:
  - Modelo de precios de pago por consumo de recursos.
  - Ofrece créditos gratuitos iniciales y pruebas gratuitas de servicios.
- Soporte Técnico:
  - Extensa documentación y recursos de soporte.
  - Varios niveles de soporte técnico disponibles: Básico, Desarrollo, Standard y Profesional.

D.3. Google Cloud Platform (GCP)

- Cumplimiento Normativo:
  - No se encontraron sobre protección de datos personales respecto a Google Cloud. Se inició ante URSEC la consulta formal Nro. 2024-71-2671-005082
  - Proporciona herramientas y certificaciones para cumplir con normativas internacionales como HIPAA, GDPR y más.
- Servicios Ofrecidos:
  - Servicios como Google Cloud Pub/Sub para colas de mensajes, Google Compute Engine para computación, y Google Cloud SQL para bases de datos.

    - Servicios de seguridad y encriptación avanzados como Google Cloud KMS.

- Costos:
    - Modelo de precios de pago por consumo de recursos
    - Ofrece créditos gratuitos y pruebas gratuitas de muchos servicios.

- Soporte Técnico:
    - Amplia documentación y recursos de soporte.
    - Varias opciones de soporte técnico disponibles (Basic, Development, Production, Enterprise).

Basado en la evaluación y considerando que AWS cumple con la normativa de protección de información clínica de Uruguay, utilizaremos AWS para implementar el middleware. AWS ofrece una amplia gama de servicios gestionados, alta disponibilidad, y escalabilidad, además de cumplir con las normativas necesarias para el manejo de datos clínicos.

En la tabla D.1 presentamos aspectos adicionales tenidos en cuenta en la evaluación de las alternativas.

Criterio	AWS	Azure	Google Cloud
Cumplimiento Normativo (Uruguay)	Cumple con normativa reciente	Verificación necesaria	Verificación necesaria
Servicios de Colas de Mensajes	Amazon SQS	Azure Queue Storage	Google Cloud Pub/Sub
Instancias de Computación	Amazon EC2	Azure Virtual Machines	Google Compute Engine
Servicios de Bases de Datos	Amazon RDS	Azure SQL Database	Google Cloud SQL
Seguridad y Encriptación	AWS KMS	Azure Key Vault	Google Cloud KMS
Costos	Competitivo, pago por uso	Competitivo, pago por uso	Competitivo, facturación por segundo
Soporte Técnico	Varias opciones disponibles	Varias opciones disponibles	Varias opciones disponibles

Tabla D.1: Aspectos considerados en la evaluación de proveedores de infraestructura.

Anexo E

Servicios utilizados de AWS

En conjunto, estas funcionalidades de AWS nos brindan la flexibilidad, la escalabilidad y la seguridad necesarias para implementar y operar de manera exitosa nuestra infraestructura de proyecto en la nube. Nos permite centrarnos en el desarrollo de soluciones innovadoras en lugar de preocuparnos por la infraestructura subyacente.

E.1. API Gateway

Utilizamos API Gateway para crear, publicar y administrar API RESTful. Esto nos permite exponer nuestros servicios de manera segura a través de API públicas o privadas.(AMAZON, 2024b)

E.2. Lambda

Hemos implementado funciones de AWS Lambda para ejecutar código sin servidor. Esto nos permite ejecutar lógica personalizada sin la necesidad de administrar servidores, lo que mejora la escalabilidad y la eficiencia de nuestros servicios.(AMAZON, 2024c)

E.3. Amazon Simple Storage Service (S3)

S3 actúa como nuestro sistema de almacenamiento en la nube, permitiéndonos almacenar y gestionar datos, archivos y objetos de manera segura y altamente disponible.(AMAZON, 2024a)

E.4. Amazon Elastic Compute Cloud (EC2)

Utilizamos instancias de EC2 para ejecutar aplicaciones y cargas de trabajo que requieren un mayor control y personalización a nivel de servidores virtuales.

E.5. CloudFormation

CloudFormation nos ayuda a definir y aprovisionar recursos de infraestructura como código. Esto nos permite crear y gestionar de manera eficiente la infraestructura de nuestro proyecto de manera repetible y automatizada.

E.6. Step Functions

Hemos implementado Step Functions para orquestar y coordinar la ejecución de múltiples servicios y procesos de manera escalable y controlada.

E.7. CloudWatch

CloudWatch es esencial para monitorear y gestionar los recursos de AWS, proporcionando información sobre el rendimiento y la salud de nuestras aplicaciones y servicios.

E.8. DynamoDB

Para el almacenamiento de datos altamente escalable y de baja latencia, utilizamos Amazon DynamoDB(AMAZON, 2024d), una base de datos NoSQL totalmente administrada.

E.9. Identity and Access Management (IAM)

Utilizamos IAM para gestionar y controlar de forma segura el acceso a nuestros recursos y servicios, garantizando que los permisos se otorguen de manera adecuada y segura.

Anexo F

Inversión en infraestructura

Nuestro proyecto fue desplegado enteramente en la nube de Amazon AWS. En la figura F.1 se detalla el costo total invertido, el valor promedio mensual y un detalle de los servicios más utilizados en los últimos meses.

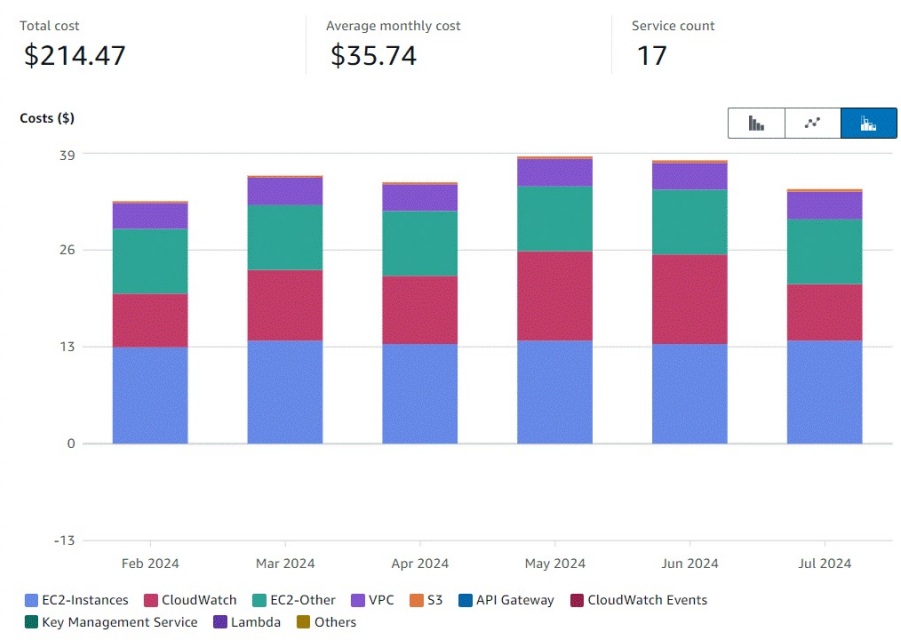


Figura F.1: Inversión en infraestructura de AWS

Costo total invertido: 214.47 dólares
Costo mensual promedio: 35.74 dólares

Anexo G

Tipos de Lenguajes

Considerando que la implementación se realizará en AWS, es importante elegir un lenguaje de programación que se integre correctamente con sus servicios y permita un desarrollo eficiente y escalable.

G.1. JavaScript (node.js)

- Integración con AWS: Tiene un buen soporte para servicios de AWS a través del AWS SDK para JavaScript.(MOZILLA, 2024)
- Escalabilidad: Manejo eficiente de operaciones asíncronas y escalabilidad horizontal.
- Buena eficiencia en el entorno de AWS, especialmente en operaciones de E/S no bloqueantes.
- Facilidad de Uso: Sintaxis clara y concisa, lo que facilita el desarrollo rápido.
- Seguridad: Capacidad para implementar prácticas de seguridad sólidas.
- Comunidad y Soporte: Gran comunidad y abundantes recursos de aprendizaje.

G.2. Python

- Integración con AWS:(AWS, 2024l) Buen soporte para servicios de AWS a través del AWS SDK para Python(A. Stross-Radschinski, 2015) (Boto3).(AWS, 2024d)
- Escalabilidad: Adecuado para aplicaciones escalables en el entorno de AWS.

- Rendimiento: Buen rendimiento en operaciones de procesamiento, aunque puede ser menor que Node.js en operaciones asíncronas.
- Facilidad de Uso: Sintaxis clara y legible, facilitando el desarrollo.
- Seguridad: Capacidad para implementar prácticas de seguridad sólidas.
- Comunidad y Soporte: Gran comunidad y abundantes recursos de aprendizaje.

G.3. Java

- Integración con AWS: Buen soporte para servicios de AWS a través del AWS SDK para Java.
- Escalabilidad: Java(AWS, 2024k) es adecuado para aplicaciones escalables en el entorno de AWS.
- Rendimiento: Buen rendimiento y eficiencia en operaciones de procesamiento.
- Facilidad de uso: Si bien tiene una sintaxis más compleja que Node.js y Python, está bien estructurada.
- Seguridad: Capacidad para implementar prácticas de seguridad sólidas.
- Comunidad y Soporte: Gran comunidad y recursos de aprendizaje, aunque menos popular que Node.js y Python.

G.4. GO

- Integración con AWS: Buen soporte para servicios de AWS a través del AWS SDK para Go.(AWS, 2024b)
- Escalabilidad: Excelente manejo de concurrencia y escalabilidad horizontal.
- Rendimiento: Muy eficiente en operaciones de procesamiento y escalabilidad en el entorno de AWS.
- Facilidad de Uso: Sintaxis simple y concisa, aunque puede tener una curva de aprendizaje para algunos desarrolladores.
- Seguridad: Capacidad para implementar prácticas de seguridad sólidas.
- Comunidad y Soporte: Comunidad en crecimiento y recursos de aprendizaje disponibles.

G.5. C# (.NET)

- Integración con AWS: Soporte para servicios AWS a través del AWS SDK para .NET.(AWS, 2024g)
- Escalabilidad: Es adecuado para aplicaciones escalables en el entorno de AWS.
- Rendimiento: Los servicios de AWS permiten crear aplicaciones de alto rendimiento con C#.(AWS, 2024f)
- Facilidad de uso: Es un lenguaje claro y organizado, fácil de usar.(litslink, 2024)
- Seguridad: Capacidad para implementar prácticas de seguridad sólidas.
- Comunidad y Soporte: Gran comunidad y abundantes recursos de aprendizaje.(Microsoft, 2024)

Criterio	Python	Node.js	Java	Go
Rendimiento	Medio	Medio	Alto	Alto
Facilidad de Uso	Alta	Alta	Media	Alta
Ecosistema	Amplio	Amplio	Amplio	Creciente
Integración con AWS	Buena	Buena	Buena	Buena
Escalabilidad	Media	Alta	Alta	Alta
Comunidad y Soporte	Alta	Alta	Alta	Media
Desarrollo Rápido	Alta	Alta	Media	Alta
Portabilidad	Alta	Alta	Alta	Alta
Cumplimiento Normativo	Alta	Alta	Alta	Alta

Tabla G.1: Comparativa de lenguajes de programación considerados.

Tomando en cuenta la integración, escalabilidad, rendimiento, facilidad de uso, costos asociados con el desarrollo en AWS y otros aspectos incluidos en la tabla G.1, **decidimos utilizar Node.js**. Esta tecnología nos brinda una excelente integración con los servicios de AWS, un alto rendimiento en operaciones asíncronas, una sintaxis clara y concisa, una gran comunidad.

Anexo H

Tipos bases de datos

El middleware almacenará una gran cantidad de información clínica, siendo la mayoría datos no estructurados, dado que pertenecen a diferentes formatos. Teniendo en cuenta características tales como la eficiencia, flexibilidad y escalabilidad, analizamos si utilizar una base de datos relacional (SQL) o no relacional (NoSQL).

H.1. Base de datos relacional (SQL)

Las bases de datos relacionales utilizan un esquema predefinido y estructurado.(ORACLE, 2024)

- Pros:
  - Fuerte consistencia y soporte para transacciones ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad).
  - Potente lenguaje de consulta (SQL) para realizar consultas complejas y uniones entre tablas.
  - Ampliamente adoptadas y compatibles con muchas herramientas y frameworks de la industria.
- Contras
  - Menos flexible para manejar datos no estructurados.
  - Escalar una base de datos relacional generalmente implica aumentar la capacidad de un solo servidor (escalabilidad vertical), lo cual puede ser limitado y costoso.
  - Menor desempeño al manejar grandes volúmenes de datos no estructurados.

H.2. Base de datos no relacional (NoSQL)

Las bases de datos NoSQL están diseñadas para manejar datos no estructurados o semi-estructurados, y pueden escalar horizontalmente agregando más servidores.(IBM, 2024b)

- Pros:
  - Capacidad para manejar datos en formatos variados sin necesidad de un esquema predefinido.
  - Fácilmente escalable agregando más nodos al sistema (escalabilidad horizontal).
  - Optimizada para el almacenamiento y recuperación rápida de datos no estructurados.
  - Incluye bases de datos de documentos, columnas, grafos y más, proporcionando opciones según la naturaleza de los datos.
- Contras:
  - Consistencia Eventual: Dependiendo del tipo de NoSQL, puede priorizar la disponibilidad y partición sobre la consistencia (modelo CAP(IBM, 2024a)).
  - Menor capacidad para realizar consultas complejas y transacciones múltiples comparado con SQL.
  - Existe ecosistema diverso, menor estandarización y más diversidad en herramientas y prácticas.

H.3. Solución híbrida (SQL + NoSQL)

Utilizar una combinación de bases de datos relacionales y no relacionales para aprovechar los beneficios de ambos enfoques.

- Pros:
  - Utiliza lo mejor de ambos mundos, ya que aprovecha la consistencia y las consultas complejas de SQL junto con la flexibilidad y escalabilidad de NoSQL.
  - Datos adecuadamente distribuidos: Almacenar datos estructurados en SQL y datos no estructurados en NoSQL según sus características y necesidades.
  - Uso eficiente de recursos mediante la distribución de cargas de trabajo entre SQL y NoSQL.
- Contras:

- Mayor complejidad en la implementación y gestión de dos sistemas diferentes.
- Necesidad de mecanismos para sincronizar datos entre las dos bases de datos.
- Costos adicionales por gestionar y mantener dos tipos de bases de datos.

Criterio	Relacional (SQL)	No Relacional (NoSQL)	Solución Híbrida (SQL + NoSQL)
Integridad de Datos	Alta, soporte ACID	Variable, generalmente consistencia eventual	Alta en SQL, variable en NoSQL
Flexibilidad del Esquema	Baja	Alta	Alta
Escalabilidad	Vertical	Horizontal	Horizontal (NoSQL), Vertical (SQL)
Desempeño con Datos No Estructurados	Bajo	Alto	Alto (NoSQL), Bajo (SQL)
Consultas Complejas	Potentes consultas SQL	Menos potentes, dependiendo del tipo	Potentes en SQL, menos potentes en NoSQL
Estandarización	Alta, ampliamente adoptado	Menor, ecosistema más diverso	Alta en SQL, diversa en NoSQL
Complejidad de Gestión	Baja	Media	Alta
Costos	Variable, depende del uso	Variable, depende del uso	Potencialmente alto debido a la gestión dual

Tabla H.1: Tabla comparativa de tipos de bases de datos.

Tras evaluar lo antes mencionado, lo ilustrado en tabla H.1, y considerando que la mayor parte de la información que se almacenará consistirá en documentos no estructurados y necesitamos flexibilidad y escalabilidad, utilizaremos una base de datos NoSQL. Esto permitirá manejar eficientemente la información clínica en sus diversos formatos y escalar horizontalmente según sea necesario. No obstante, no se descarta el uso futuro de bases de datos relacionales para almacenar información en caso de que sea necesario realizar consultas complejas que surjan a partir de nuevas funcionalidades.

H.4. Tipos de bases de datos NoSQL

Teniendo en cuenta que se ha decidido utilizar una base de datos no relacional, consideraremos MongoDB y Amazon DynamoDB. Basaremos la elección en criterios como flexibilidad, escalabilidad, rendimiento, facilidad de uso, costos, y cumplimiento de la normativa de protección de datos clínicos de Uruguay.

H.5. MongoDB

MongoDB es una base de datos NoSQL orientada a documentos que almacena datos en un formato flexible, similar a JSON, llamado BSON (Binary JSON).(MongoDB, 2024a)(MongoDB, 2024c)

- Pros:
  - Flexibilidad de esquemas, ya que MongoDB almacena documentos en formato BSON, permitiendo cambios en la estructura de datos.
  - Soporta consultas medianamente complejas y agregaciones.
  - Diseñado para escalar horizontalmente mediante el sharding.(MongoDB, 2024b)
  - Disponible tanto en infraestructura propia (on-premise) como en servicios gestionados en la nube (MongoDB Atlas).
  - Comunidad y soporte: Existe una amplia comunidad de usuarios y buena documentación, además de soporte empresarial disponible.
- Contras:
  - Consistencia eventual (las actualizaciones tardarán en llegar a otras réplicas): A veces, dependiendo de la configuración, puede ofrecer consistencia eventual en lugar de consistencia fuerte
  - El sharding y la gestión de múltiples nodos pueden ser costosos y complejos de administrar.

H.6. Amazon DynamoDB

Amazon DynamoDB(AWS, 2024h) es un servicio de base de datos NoSQL totalmente gestionado, proporcionado por AWS. Ofrece un rendimiento rápido y predecible con una alta disponibilidad y escalabilidad. DynamoDB se utiliza para aplicaciones que requieren baja latencia y tiempos de respuesta rápidos, como aplicaciones web, móviles, de IoT (Internet of Things), y de juegos en tiempo real.

- Pros:
  - Ofrece un rendimiento estable con respecto a la latencia, velocidad de respuesta y capacidad de procesamiento, incluso a gran escala.

Criterio	MongoDB	Amazon DynamoDB
Flexibilidad del Esquema	Alta	Alta
Consultas Complejas	Potentes, flexible con agregaciones	Menos flexibles, orientadas a clave-valor
Escalabilidad	Horizontal mediante sharding	Horizontal automática
Rendimiento	Alto, pero puede requerir ajuste manual	Consistente y predecible
Facilidad de Uso	Intuitiva, con buena documentación	Alta, gestión automática por AWS
Integración	Multiplataforma	Integración nativa con AWS
Cumplimiento Normativo	Depende del despliegue (Atlas recomendado)	Cumple con normativas incluyendo la de Uruguay
Costos	Variables, pueden ser altos con sharding	Variables, pueden aumentar con alta carga
Despliegue	On-premise, en la nube (MongoDB Atlas)	En la nube (AWS)

Tabla H.2: Tabla comparativa de bases de datos no relacionales.

- Gestiona automáticamente la capacidad y el rendimiento mediante la escalabilidad horizontal.
- Replicación automática de datos en múltiples regiones de AWS, asegurando alta disponibilidad y recuperación ante desastres.
- Integración nativa con otros servicios de AWS, facilitando la implementación de soluciones en la nube.
- Cumple con las normativas de seguridad y protección de datos clínicos de Uruguay según la sección 2.4

- Contras:
  - Ofrece consistencia eventual por defecto, aunque permite configuraciones para consistencia fuerte a costa de rendimiento.
  - Los costos pueden aumentar significativamente con el uso intensivo y la necesidad de lectura/escritura consistente.
  - Menos flexible en comparación con el lenguaje de consulta de MongoDB.

Basados en lo antes mencionado y en los criterios evaluados en la tabla H.2, determinamos que Amazon DynamoDB es la mejor opción por su capacidad de gestión automática, rendimiento predecible y alta disponibilidad, junto con el cumplimiento de la normativa de protección de datos clínicos de Uruguay.

Aunque MongoDB ofrece flexibilidad y consultas más potentes, DynamoDB proporciona una solución más robusta y menos compleja de gestionar a gran escala, especialmente en entornos como este, que requieren alta disponibilidad. Cabe aclarar que, si bien la mayor desventaja de DynamoDB es la poca capacidad para realizar consultas complejas, en los requerimientos de este proyecto no se detectaron consultas de este tipo.

Anexo I

Recepción de información para la generación de documentos

Para generar los documentos clínicos, es necesario contar con información sobre el autor, el paciente y los repositorios de documentos. Hay dos enfoques considerados para gestionar esta información:

- Recibir toda la información necesaria desde el generador de información, generar el documento y enviarla al repositorio.
- Mantener datos pre cargados del paciente y autor en el middleware, y recibir solo los identificadores de estos datos junto con la información específica del documento a generar.

La información del repositorio se utiliza exclusivamente en el middleware y contiene claves de firma, por lo que debe estar protegida.

Es importante tener en cuenta que, la información del repositorio se utiliza exclusivamente en el middleware y contiene las claves para la firma del documento clínico, por lo que debe estar protegida. A continuación presentamos las dos alternativas evaluadas para el almacenamiento de datos adicionales para la generación de documentos.

I.1. Recibir toda la información desde el generador de información

En este enfoque, el middleware recibe toda la información necesaria (datos del autor, del paciente y del documento) directamente desde el generador de información. El middleware procesa la información según el formato requerido y la envía al repositorio.

- Pros
  - Simplicidad en la arquitectura del middleware.
  - Menor necesidad de sincronización y almacenamiento en el middleware.
  - Cada solicitud contiene toda la información necesaria para generar el documento.
- Contras
  - Mayor cantidad de datos transferidos en cada solicitud.
  - Dependencia de la precisión y completitud de la información enviada por los generadores de documentos.
  - Potencialmente mayor latencia debido al procesamiento de grandes cantidades de datos.

I.2. Mantener datos pre guardados en el middleware y recibir identificadores

En este enfoque, el middleware mantiene una base de datos con la información pre cargada de pacientes y autores. Los generadores de documentos envían solo los identificadores de esta información junto con los datos específicos del documento.

- Pros
  - Reducción en la cantidad de datos transferidos en cada solicitud.
  - Mejor control y consistencia en la información de pacientes y autores.
  - Eficiencia en el procesamiento de documentos.
- Contras
  - Necesidad de sincronización entre los generadores de documentos y la base de datos del middleware.
  - Complejidad adicional en la gestión y protección de la base de datos.
  - Riesgo de inconsistencias si los datos pre guardados no se actualizan correctamente.

Considerando estos aspectos y los mencionados en la tabla I.1, utilizaremos una combinación de ambas estrategias, diferenciando el tratamiento de la información del paciente y el autor de la información del repositorio:

- Información de paciente y autor: recibiremos la totalidad de esta información desde el generador dado que no es un gran volumen de datos. Esto simplifica la arquitectura del middleware y elimina la necesidad de sincronización y gestión de una base de datos adicional.

Criterio	Recibir Toda la Información	Mantener Datos Preguardados
Simplicidad de Implementación	Alta	Media
Cantidad de Datos Transferidos	Media	Baja
Dependencia de Sincronización	Baja	Alta
Control y Consistencia	Baja	Alta
Latencia del Procesamiento	Alta	Baja
Complejidad en la Gestión	Baja	Alta

Tabla I.1: Tabla comparativa de alternativas de recepción de información.

- Información de repositorios: como esta información se utiliza únicamente dentro del middleware, la almacenaremos y gestionaremos dentro del mismo.

De esta forma, nos encargamos de gestionar únicamente la información propia del middleware sin ocuparnos de sincronizar la información administrada por terceros.

Anexo J

Detalles del envío a repositorios

J.1. Salud digital

Formato: XML
Método de Envío: Web Service.
Proceso: La función Lambda convierte el documento al formato XML requerido y lo envía al endpoint del servicio web de salud digital. Si el envío es exitoso, se actualiza el estado del documento a "SENT". Si falla, se registra el error y se programa un reintento.

J.2. Repositorios FHIR

Formato: JSON.
Método de Envío: API REST.
Proceso: La función Lambda convierte el documento al formato JSON requerido y realiza una solicitud POST a la API REST del repositorio FHIR. Si el envío es exitoso, se actualiza el estado del documento a "SENT". Si falla, se registra el error y se programa un reintento.

J.3. Métodos custom

Formato: Depende del repositorio.
Método de Envío: Puede variar (Web Service, API REST, etc.).
Proceso: La función Lambda puede incluir lógica específica para enviar documentos a repositorios particulares que requieren métodos de envío customizados.

Esto puede incluir autenticación específica, transformación de formatos, y más.

Anexo K

Tabla de estados de documentos

En la tabla K.1 se encuentran los diferentes estados por los que puede pasar el proceso de generación de un documento. Estos estados pueden ser internos, o externos. Los estados internos tienen mayor información sobre el ciclo de vida del documento, y los estados externos se utilizan para el seguimiento externo del proceso. Esto quiere decir que mientras la generación del documento está en progreso, un consultante externo solo verá el estado "IN_PROGRESS". En esta tabla (K.1) se ve además la correspondencia entre los estados internos manejados por DOCLUY y los estados devueltos a sistemas externos.

ESTADO INTERNO	**ESTADO DISPOSITIVO**
QUEUED	PENDING
PROCESS_STARTED	IN_PROGRESS
PROCESS_STARTED_ERROR	IN_PROGRESS
ERROR_VALIDATING_DOCUMENT	IN_PROGRESS
DOCUMENT_VALIDATED	IN_PROGRESS
INVALID_DOCUMENT	ERROR
CDA_GENERATED	IN_PROGRESS
ERROR_GENERATING_CDA	IN_PROGRESS
CDA_SIGNED	IN_PROGRESS
ERROR_SIGNING_CDA	IN_PROGRESS
XDS_GENERATED	IN_PROGRESS
ERROR_GENERATING_XDS	IN_PROGRESS
XDS_QUEDED	IN_PROGRESS
ERROR_QUEUEING_XDS	IN_PROGRESS
XDS_SENT	SENT
ERROR_SENDING_XDS	IN_PROGRESS

Tabla K.1: Tabla de estado de documentos

Anexo L

Definición de API de creación de documentos

En la tabla L.1 se describe cada campo que debe recibir la solución para poder generar un documento clínico y enviarlo a un repositorio.

A continuación se muestra un ejemplo de una solicitud recibida por la API que genera un documento clínico.


```
{
    "applicationId": "0001",
    "repositoryId": "ed736c1e-8bc6-4a41-9782-a8d5c52e8c36",
    "externalIdentifier": "0507_0001",
    "externalVersion": 1,
    "orderNumber": 202405060005,
    "author": {
        "dependency": "Clinica Central",
        "firstName": "Juan",
        "identityAssigningAuthorityCountryId": "UY",
        "identityAssigningAuthorityId": "0002",
        "identityValue": "125489465",
        "lastName": "Rodriguez",
        "lastName2": "Gonzalez",
        "middleName": "Manuel",
        "phoneNumber": "099885511",
        "phoneNumberEquipmentTypeId": "0002",
        "phoneNumberUseCodeId": "0007",
        "prefix": "Dr.",
        "role": "Medico",
        "specialty": "Traumatologo"
    },
    "document": {
        "attachmentB64": "bls3lke8gk...",
```

```
        "attachmentTypeId": "0004",
        "documentDescription": "Reporte DINABANG de H/Q Ratio",
        "documentName": "DINABANG - H/Q Ratio",
        "formatCodeId": "0001",
        "languageId": "es",
        "securityCodeId": "0001",
        "statusId": "ST0001"
    },
    "encounter": {
        "axis1Id": "OD1N0002",
        "axis2Id": "OD2N0005",
        "axis3Id": "OD3N0111",
        "endDate": "20230319154500",
        "institutionId": "001",
        "startDate": "20230319153000"
    },
    "patient": {
        "birthdate": "19800630",
        "firstName": "Maria",
        "genderId": "0003",
        "identityAssigningAuthorityCountryId": null,
        "identityAssigningAuthorityId": "0001",
        "identityValue": "12324586",
        "institutionIdentityValue": "75641",
        "lastName": "Fernandez",
        "lastName2": "Perez",
        "middleName": "Laura"
    }
}
```


Campo	Tipo de dato	Requerido/ Opcional
applicationId: Aplicación que generó la creación del CDA. Es obligatorio si el autor es una máquina. Opcional si es una persona.	string	R/O
externalIdentifier: Identificador único del documento en el sistema externo que genera la información. El identificador debe ser único para la institución.	string	R
externalVersion: Versión del documento en el sistema externo.	int	R
orderNumber: Número de orden único que identifica el acto asistencial.	int	O

Campo	Tipo de dato	Requerido/ Opcional
repositoryId: Identificador del repositorio en el que se registrará el documento.	string	R
Información del autor del documento		
author.dependency: Dependencia o local de la institución donde se generó el acto clínico	string	O
author.firstName: Primer nombre del autor del documento. Es obligatorio si el autor es una persona. Opcional si es una máquina.	string	R/O
author.identityAssigningAuthorityCountryId: País al que corresponde el documento de identidad que identifica al autor del documento. Este campo es requerido sólo si el tipo de documento es CEDULA DE IDENTIDAD EXTRANJERA o PASAPORTE EXTRANJERO. Para el resto de los tipos de documento este campo es opcional.	string	R/O
author.identityAssigningAuthorityId: Tipo de documento de identidad que identifica al autor del documento. Es obligatorio si el autor es una persona. Opcional si es una máquina.	string	R/O
author.identityValue: Número de documento de identidad que identifica al autor del documento. Es obligatorio si el autor es una persona. Opcional si es una máquina.	string	R/O
author.lastName: Primer apellido del autor del documento. Es obligatorio si el autor es una persona. Opcional si es una máquina.	string	R/O
author.lastName2: Segundo apellido del autor del documento.	string	O
author.middleName: Segundo nombre del autor del documento.	string	O
author.phoneNumber: Número de teléfono del autor del documento. Se utiliza para notificación de certificaciones médicas y enfermedades de notificación obligatoria. El campo es obligatorio si se necesita alguna de las notificaciones.	string	R/O
author.phoneNumberEquipmentTypeId: Tipo de equipo del número de teléfono del autor del documento. El campo es obligatorio si se necesita realizar notificaciones.	string	R/O

Campo	Tipo de dato	Requerido/ Opcional
author.phoneNumberUseCodeId: Use code del número de teléfono del autor del documento. El campo es obligatorio si se necesita realizar notificaciones.	string	R/O
author.prefix: Prefijo del autor del documento	string	O
author.role: Rol del autor del documento	string	O
author.specialty: Especialidad del autor del documento	string	O
Información del documento		
document.attachmentB64: Imagen o pdf en base 64 del contenido del CDA.	string	R
document.attachmentTypeId: Tipo de contenido del CDA.	string	R
document.documentDescription: Descripción asociada al documento que se está generando. La descripción no es visible en el CDA.	string	O
document.documentName: Título asociado al documento que se está generando.	string	R
document.formatCodeId: Tipo de documento que se genera en el CDA.	string	R
document.languageId: Idioma en el que se encuentra la información en el documento.	string	R
document.securityCodeId: Tipo de confidencialidad del documento.	string	R
document.statusId: Status del documento.	string	R
Información del acto asistencial		
encounter.axis1Id: Identificador del eje 1 del tipo del documento. Tipo de documento general.	string	R
encounter.axis2Id: Identificador del eje 2 del tipo del documento. Tipo de documento detallado.	string	R
encounter.axis3Id: Identificador del eje 3 del tipo del documento. Servicio.	string	R
encounter.endDate: Fecha y hora en la que finalizó el acto clínico en formato YYYYMMDDHHMMSS	string	R
encounter.institutionId: Institución que genró el acto clínico. Si se envía vacía se utilizará la institución que almacenará el documento.	string	O
encounter.startDate: Fecha y hora en la que inició el acto clínico en formato YYYYMMDDHHMMSS	string	R
Información del paciente		

Campo	Tipo de dato	Requerido/ Opcional
patient.birthdate: Fecha de nacimiento del paciente. Formato YYYYMMDD	string	O
patient.firstName: Primer nombre del paciente.	string	R
patient.genderId: Género del paciente.	string	R
patient.identityAssigningAuthorityCountryId: País al que corresponde el documento de identidad que identifica al paciente. Este campo es requerido si el tipo de documento es CEDULA DE IDENTIDAD EXTRANJERA o PASAPORTE EXTRANJERO. Para el resto de los tipos de documento este campo es opcional.	string	R/O
patient.identityAssigningAuthorityId: Tipo de documento de identidad que identifica al paciente.	string	R
patient.identityValue: Número de documento de identidad que identifica al paciente.	string	R
patient.institutionIdentityAssigningAuthorityId: Identificador del sistema que generó el identificador del paciente (patient.institutionIdentityValue) dentro de la institución. Si no se envía este campo, se tomará el valor por defecto configurado en el repositorio.	string	R/O
patient.institutionIdentityValue: Identificador del paciente dentro de la institución.	string	R
patient.lastName: Primer apellido del paciente.	string	R
patient.lastName2: Segundo apellido del paciente.	string	O
patient.middleName: Segundo nombre del paciente.	string	O

Tabla L.1: Información requerida en la API de creación de documentos.

Anexo M

Ejemplo de CDA y transacción ITI-41

A continuación se muestra un ejemplo de documento CDA generado por la solución y la implementación de la transacción ProvideAndRegisterDocumentSet (ITI-41) que se utiliza para almacenar el CDA en el Repositorio XDS local y registrarlo en el Registro XDS local y Nacional.

La transacción ITI-41 ya incluye el mensaje MTOM utilizado para realizar la transmisión al servicio SOAP.

El CDA generado se basa en los lineamientos definidos en el documento *Guía de implementación CDA Mínimo HL7 V3 CDA-R2* (Salud Digital, 2019) provisto por Salud Digital.

La transacción ProvideAndRegisterDocumentSet (ITI-41) y el mensaje MTOM generados se basan en los lineamientos definidos en el documento *Guía Técnica de Metadatos XDS* (Salud Digital, 2022) también provisto por Salud Digital.

En la tabla M.1 se describen los parámetros utilizados para la creación del CDA, transacción ITI-41 y mensaje MTOM.

Ejemplo de documento CDA generado:

```
<?xml version="1.0" encoding="UTF-8"?>
<SignedClinicalDocument
      xmlns="urn:salud.uy/2014/signed-clinical-document"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="urn:hl7-org:v3 signed-clinical-
       documentV2_5.xsd">
 <!-- elemento raiz del CDA -->
 <ClinicalDocument xmlns="urn:hl7-org:v3"
       xmlns:voc="urn:hl7-org:v3/voc">
  <!-- estructura fija de esquema de CDA normativo de HL7 -->
  <typeId extension="POCD_HD000040"
    root="2.16.840.1.113883.1.3"/>
  <!-- identificacion unica del documento -->
```

```
<id root="[documentOID]"/>
<!-- eje 1 de la Ontologia de Documentos, codificado en
 LOINC -->
<code code="[encounter.axis1.code]"
   codeSystem="[encounter.axis1.codeSystem]"
   codeSystemName="[encounter.axis1.codeSystemName]"
   displayName="[encounter.axis1.name]"/>
<!-- nombre del documento -->
<title>[document.documentName]</title>
<!-- fecha del acto asistencial -->
<effectiveTime value="[encounter.startDate]"/>
<!-- valor de confidencialidad del documento -->
<confidentialityCode code="[document.securityCode.code]"
      codeSystem="[document.securityCode.codeSystem]"/>
<!-- idioma en que el documento esta redactado -->
<languageCode code="[document.language.code]"/>
<!-- identificador que es comun sobre todas las revisiones
 del documento -->
<setId root="[documentSetId]"/>
<versionNumber value="[documentVersion]"/>
<recordTarget>
 <!-- persona a la que pertenece ese documento clinico-->
 <patientRole>
  <id extension="[patient.identityValue]"
   root="[patient.identityAssigningAuthority.oid]"/>
  <patient>
   <!-- datos que identifican al paciente -->
   <name>
    <given>[patient.firstName]</given>
    <given>[patient.middleName]</given>
    <family>[patient.lastName]</family>
    <family>[patient.lastName2]</family>
   </name>
   <!-- Sexo del paciente -->
   <administrativeGenderCode code="[patient.gender.code]"
           codeSystem="[patient.gender.codeSystem]"
           displayName="[patient.gender.name]"/>
   <!-- fecha de nacimiento del paciente -->
   <birthTime value="[patient.birthdate]"/>
  </patient>
 </patientRole>
</recordTarget>
<author>
 <!-- tiempo en que la persona o dispositivo comenzaron su
  participacion como autor -->
 <time value="[encounter.startDate]"/>
 <assignedAuthor>
  <!-- datos que identifican al autor del documento -->
  <id extension="[author.identityValue]"
   root="[author.identityAssigningAuthority.oid]"/>
```

```
  <!-- datos del autor del documento -->
  <!-- Si el autor es una persona se incluye assignedPerson
   y no se incluye assignedAuthoringDevice -->
  <assignedPerson>
   <name>
    <given>[author.firstName]</given>
    <given>[author.middleName]</given>
    <family>[author.lastName]</family>
    <family>[author.lastName2]</family>
   </name>
  </assignedPerson>
  <!-- Si el autor es un dispositivo se incluye
   assignedAuthoringDevice y no se incluye
   assignedPerson -->
  <!--
  <assignedAuthoringDevice>
   <softwareName>[application.name]</softwareName>
  </assignedAuthoringDevice>
  -->
  <!-- Institucion que representa al autor del documento -->
  <representedOrganization>
   <id root="[encounter.institution.oid]"/>
   <name>[encounter.institution.name]</name>
  </representedOrganization>
 </assignedAuthor>
</author>
<custodian>
 <!-- datos de la institucion que esta a cargo de la
  custodia del documento -->
 <assignedCustodian>
  <representedCustodianOrganization>
   <id root="[repository.custodianOrganizationOID]"/>
   <name>[repository.custodianOrganizationName]</name>
  </representedCustodianOrganization>
 </assignedCustodian>
</custodian>
<componentOf>
 <!-- detalle del encuentro entre el paciente y el
  integrante del equipo de salud -->
 <encompassingEncounter classCode="ENC">
  <code code="[encounter.axis2.code]"
     codeSystem="[encounter.axis2.codeSystem]"
     codeSystemName="[encounter.axis2.codeSystemName]"
     displayName="[encounter.axis2.name]"/>
  <!-- eje 2 de la Ontologia de Documentos -->
  <effectiveTime xsi:type="IVL_TS">
   <low value="[encounter.startDate]"/>
   <high value="[encounter.endDate]"/>
  </effectiveTime>
  <!-- eje 3 de la Ontologia de Documentos -->
```

```
    <location typeCode="LOC">
     <healthCareFacility classCode="SDLOC">
      <code code="[encounter.axis3.code]"
         codeSystem="[encounter.axis3.codeSystem]"
         codeSystemName="[encounter.axis3.codeSystemName]"
         displayName="[encounter.axis3.name]"/>
     </healthCareFacility>
    </location>
   </encompassingEncounter>
  </componentOf>
  <component>
   <nonXMLBody>
    <text mediaType=
       "[document.contentType.cdaNonXMLBodyMediaType]"
       representation=
          "[document.contentType.cdaNonXMLBodyRepresentation]">
    [document.attachmentB64]
   </text>
   </nonXMLBody>
  </component>
 </ClinicalDocument>
 <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
  <!-- firma del documento -->
  <SignedInfo>
   <CanonicalizationMethod Algorithm=
      "http://www.w3.org/TR/2001/REC-xml-c14n-20010315#
      WithComments"/>
   <SignatureMethod Algorithm=
      "http://www.w3.org/2001/04/xmldsig-more#rsa-sha512"/>
   <Reference URI="">
    <Transforms>
     <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#
     enveloped-signature"/>
    </Transforms>
    <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc
    #sha512"/>
    <DigestValue>XYe5VmEJaJ6aM1EEsOLnxSG/FyklC3Lb/nHzkoFCWVbc
    AZuBzaDOMWuGIGJIy4UltVn0eKpvACQ0
       3+gfU8L6Gw==</DigestValue>
   </Reference>
  </SignedInfo>
  <SignatureValue>[signature]</SignatureValue>
 </Signature>
</SignedClinicalDocument>
```

Ejemplo de transacción ITI-41 y mensaje MTOM:

```
POST [repository.endpointHostname]:[repository.endpointPort]/
 RepositorioXDS/adocumentrepository_services?wsdl HTTP/1.1
Content-Type: multipart/related; boundary=----=
```

```
 _Part_8_819376715.1465835825714; type="application/xop+xml";
 start="<rootpart@soapui.org>"; start-info="text/xml";
Host: [repository.endpointHostname]
Transfer-Encoding: chunked

------=_Part_8_819376715.1465835825714
Content-Type: application/xop+xml; charset=UTF-8; type="appli
 cation/soap+xml"
Content-Transfer-Encoding: binary
Content-ID: rootpart@soapui.org

<?xml version='1.0' encoding='UTF-8'?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
 soap/envelope/"
    xmlns:urn="urn:ihe:iti:xds-b:2007"
    xmlns:urn1="urn:oasis:names:tc:ebxml-regrep:xsd:lcm:3.0"
    xmlns:urn2="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0">
 <soapenv:Header/>
 <soapenv:Body>
  <urn:ProvideAndRegisterDocumentSetRequest>
   <urn1:SubmitObjectsRequest>
    <urn2:RegistryObjectList>
     <!--ExtrinsicObject: Atributos descriptivos del documen-
      to (metadata del documento que se intercambia). Cada EO
       representa un solo documento que se identifica de for-
       ma unica con el atributo UniqueId-->
     <urn2:ExtrinsicObject home="urn:oid:[repository.reposito
      ryHomeCommunityId]"
          id="1.[documentOID]"
          isOpaque=""
          lid=""
          mimeType="text/xml"
          objectType="urn:uuid:7edca82f-054d-47f2-a032-
           9b2a5b5186c1"
          status="urn:oasis:names:tc:ebxml-regrep:StatusType:
           [document.status.status]">
      <!--Fecha de creacion del documento-->
      <urn2:Slot name="creationTime"
        slotType="">
       <urn2:ValueList>
        <urn2:Value>[encounter.startDate]</urn2:Value>
       </urn2:ValueList>
      </urn2:Slot>
      <!--Lenguaje de los campos con contenido en texto libre
       en el documento-->
      <urn2:Slot name="languageCode"
        slotType="">
       <urn2:ValueList>
        <urn2:Value>[document.language.code]</urn2:Value>
       </urn2:ValueList>
```

```
</urn2:Slot>
<!--Hora de inicio de la consulta-->
<urn2:Slot name="serviceStartTime"
  slotType="">
 <urn2:ValueList>
  <urn2:Value>[encounter.startDate]</urn2:Value>
 </urn2:ValueList>
</urn2:Slot>
<!--Hora de fin de la consulta-->
<urn2:Slot name="serviceStopTime"
  slotType="">
 <urn2:ValueList>
  <urn2:Value>[encounter.endDate]</urn2:Value>
 </urn2:ValueList>
</urn2:Slot>
<!--Identificacion asociada al paciente al cual se le
 realizo el acto clinico del documento-->
<urn2:Slot name="sourcePatientId"
  slotType="">
 <urn2:ValueList>
  <urn2:Value>[patient.institutionIdentityValue]^^^
   &amp;[encounter.institution.assigningAuthority]&amp;
   ISO</urn2:Value>
 </urn2:ValueList>
</urn2:Slot>
<!--Codigo unico del repositorio en el cual se guardara
 el documento-->
<urn2:Slot name="repositoryUniqueId"
  slotType="">
 <urn2:ValueList>
  <urn2:Value>[repository.oid]</urn2:Value>
 </urn2:ValueList>
</urn2:Slot>
<!--Informacion demografica del paciente sobre el cual
 se esta generando el documento-->
<urn2:Slot name="sourcePatientInfo"
  slotType="">
 <urn2:ValueList>
  <urn2:Value>PID-3|[patient.institutionIdentityValue]
   ^^^&amp;[encounter.institution.assigningAuthority]&
   amp;ISO~[patient.identityValue]^^^&amp;[patient.iden
   tityAssigningAuthority.oid]&amp;ISO</urn2:Value>
  <urn2:Value>PID-5|[patient.lastName]^[patient.middle
   Name]^[patient.firstName]^^</urn2:Value>
  <urn2:Value>PID-7|[patient.birthdate]</urn2:Value>
  <urn2:Value>PID-8|[patient.gender.code]</urn2:Value>
 </urn2:ValueList>
</urn2:Slot>
<!--CPOE, ATNA-->
<urn2:Slot name="CPOE"
```

```
  slotType="">
 <urn2:ValueList>
  <urn2:Value>[orderNumber]</urn2:Value>
 </urn2:ValueList>
</urn2:Slot>
<!--OIDApplication, ATNA-->
<urn2:Slot name="OIDApplication"
  slotType="">
 <urn2:ValueList>
  <urn2:Value>[application.oid]</urn2:Value>
 </urn2:ValueList>
</urn2:Slot>
<urn2:Name>
 <urn2:LocalizedString charset=""
      xml:lang=""
      value="[document.documentName]"/>
</urn2:Name>
<!-- Descripcion asociada al documento que se esta
 generando-->
<urn2:Description>
 <urn2:LocalizedString charset=""
      xml:lang=""
      value="[document.documentDescription]"/>
</urn2:Description>
<!--Autor-->
<urn2:Classification classificationNode=""
     classificationScheme="urn:uuid:93606bcf-9494-43ec-
      9b4e-a7748d1a838d"
     classifiedObject="1.[documentOID]"
     home=""
     id="cl01"
     lid=""
     nodeRepresentation=""
     objectType="urn:oasis:names:tc:ebxml-regrep:Object
      Type:RegistryObject:Classification"
     status="">
 <urn2:Slot name="authorInstitution"
   slotType="">
  <urn2:ValueList>
   <urn2:Value>[encounter.institution.name]^^^^^^^^^
    [encounter.institution.oid]</urn2:Value>
  </urn2:ValueList>
 </urn2:Slot>
 <!--Autor del documento-->
 <urn2:Slot name="authorPerson"
   slotType="">
  <urn2:ValueList>
   <urn2:Value>[author.identityValue]^[author.lastName]
    ^[author.firstName]^[author.middleName]^^[author.
    prefix]&amp;[author.identityAssigningAuthority.oid]
```

```
    &amp;amp;ISO</urn2:Value>
   </urn2:ValueList>
  </urn2:Slot>
  <!--Rol del autor del documento-->
  <urn2:Slot name="authorRole"
    slotType="">
   <urn2:ValueList>
    <urn2:Value>[author.role]</urn2:Value>
   </urn2:ValueList>
  </urn2:Slot>
  <!--Especialidad del autor del documento-->
  <urn2:Slot name="authorSpecialty"
    slotType="">
   <urn2:ValueList>
    <urn2:Value>[author.specialty]</urn2:Value>
   </urn2:ValueList>
  </urn2:Slot>
  <urn2:Slot name="authorLocation"
    slotType="">
   <urn2:ValueList>
    <urn2:Value>[author.dependency]</urn2:Value>
   </urn2:ValueList>
  </urn2:Slot>
  <urn2:Slot name="authorTelecommunication"
    slotType="">
   <urn2:ValueList>
    <urn2:Value>^[author.phoneUseCode.useCode]^[author.
     phoneEquipmentType.equipmentType]^^^^[author.phone
     Number]^^</urn2:Value>
   </urn2:ValueList>
  </urn2:Slot>
 </urn2:Classification>
 <!--ClassCode(tipo de documento)-->
 <urn2:Classification classificationNode=""
      classificationScheme="urn:uuid:41a5887f-8865-4c09-
       adf7-e362475b143a"
      classifiedObject="1.[documentOID]"
      home=""
      id="cl02"
      lid=""
      nodeRepresentation="[encounter.axis1.code]"
      objectType="urn:oasis:names:tc:ebxml-regrep:Object
       Type:RegistryObject:Classification"
      status="">
  <urn2:Slot name="codingScheme"
    slotType="">
   <urn2:ValueList>
    <urn2:Value>[encounter.axis1.codeSystem]</urn2:Value>
   </urn2:ValueList>
  </urn2:Slot>
```

```
 <urn2:Name>
  <urn2:LocalizedString charset=""
       xml:lang=""
       value="[encounter.axis1.name]"/>
 </urn2:Name>
</urn2:Classification>
<!--ConfidentialityCode-->
<urn2:Classification classificationNode=""
     classificationScheme="urn:uuid:f4f85eac-e6cb-4883-
      b524-f2705394840f"
     classifiedObject="1.[documentOID]"
     home=""
     id="cl03"
     lid=""
     nodeRepresentation="[document.securityCode.code]"
     objectType="urn:oasis:names:tc:ebxml-regrep:Object
      Type:RegistryObject:Classification"
     status="">
 <urn2:Slot name="codingScheme"
   slotType="codingScheme">
  <urn2:ValueList>
   <urn2:Value>[document.securityCode.codeSystem]</urn2
    :Value>
  </urn2:ValueList>
 </urn2:Slot>
 <urn2:Name>
  <urn2:LocalizedString charset=""
       xml:lang=""
       value="[document.securityCode.name]"/>
 </urn2:Name>
</urn2:Classification>
<urn2:Classification classificationNode=""
     classificationScheme="urn:uuid:a09d5840-386c-46f2-
      b5ad-9c3699a4309d"
     classifiedObject="1.[documentOID]"
     home=""
     id="cl05"
     lid=""
     nodeRepresentation="[document.formatCode.code]"
     objectType="urn:oasis:names:tc:ebxml-regrep:Object
      Type:RegistryObject:Classification"
     status="">
 <urn2:Slot name="codingScheme"
   slotType="">
  <urn2:ValueList>
   <urn2:Value>[document.formatCode.codeSystem]
    </urn2:Value>
  </urn2:ValueList>
 </urn2:Slot>
 <urn2:Name>
```

```
  <urn2:LocalizedString charset=""
       xml:lang=""
       value="[document.formatCode.name]"/>
 </urn2:Name>
</urn2:Classification>
<!-- HealthCareFacilityTypeCode-->
<urn2:Classification classificationScheme="urn:uuid:f33
 fb8ac-18af-42cc-ae0e-ed0b0bdb91e1"
     classifiedObject="1.[documentOID]"
     id="cl06"
     objectType="urn:oasis:names:tc:ebxml-regrep:Object
      Type:RegistryObject:Classification"
     nodeRepresentation="[encounter.institution.health
      CareFacilityType.code]">
 <urn2:Slot name="codingScheme">
  <urn2:ValueList>
   <urn2:Value>[encounter.institution.healthCareFacili
    tyType.codeSystem]</urn2:Value>
  </urn2:ValueList>
 </urn2:Slot>
 <urn2:Name>
  <urn2:LocalizedString charset=""
       xml:lang=""
       value="[encounter.institution.healthCareFacility
        Type.name]"/>
 </urn2:Name>
</urn2:Classification>

<!--TypeCode-->
<urn2:Classification classificationNode=""
     classificationScheme="urn:uuid:f0306f51-975f-434e-
      a61c-c59651d33983"
     classifiedObject="1.[documentOID]"
     home=""
     id="cl08"
     lid=""
     nodeRepresentation="[encounter.axis2.code]"
     objectType="urn:oasis:names:tc:ebxml-regrep:Object
      Type:RegistryObject:Classification"
     status="">
 <urn2:Slot name="codingScheme"
   slotType="">
  <urn2:ValueList>
   <urn2:Value>[encounter.axis2.codeSystem]</urn2:Value>
  </urn2:ValueList>
 </urn2:Slot>
 <urn2:Name>
  <urn2:LocalizedString charset=""
       xml:lang=""
       value="[encounter.axis2.name]"/>
```

```
 </urn2:Name>
</urn2:Classification>
<!--PracticeSettingCode-->
<urn2:Classification classificationNode=""
      classificationScheme="urn:uuid:cccf5598-8b07-4b77-
       a05e-ae952c785ead"
      classifiedObject="1.[documentOID]"
      home=""
      id="cl07"
      lid=""
      nodeRepresentation="[encounter.axis3.code]"
      objectType="urn:oasis:names:tc:ebxml-regrep:Object
       Type:RegistryObject:Classification"
      status="">
 <urn2:Slot name="codingScheme"
   slotType="">
  <urn2:ValueList>
   <urn2:Value>[encounter.axis3.codeSystem]
    </urn2:Value>
  </urn2:ValueList>
 </urn2:Slot>
 <urn2:Name>
  <urn2:LocalizedString charset=""
       xml:lang=""
       value="[encounter.axis3.name]"/>
 </urn2:Name>
</urn2:Classification>
<urn2:ExternalIdentifier home=""
      id="ei01"
      identificationScheme="urn:uuid:58a6f841-87b3-4a3e
       -92fd-a8ffeff98427"
      lid=""
      objectType="urn:oasis:names:tc:ebxmlregrep:Object
       Type:RegistryObject:ExternalIdentifier"
      registryObject="1.[documentOID]"
      status=""
      value="[patient.institutionIdentityValue]^^^&amp;
       [encounter.institution.assigningAuthority]&amp;
       ISO">
 <urn2:Name>
  <urn2:LocalizedString charset=""
       xml:lang=""
       value="XDSDocumentEntry.patientId"/>
 </urn2:Name>
</urn2:ExternalIdentifier>
<urn2:ExternalIdentifier home=""
      id="ei02"
      identificationScheme="urn:uuid:2e82c1f6-a085-4c72
       -9da3-8640a32e42ab"
      lid=""
```

```
        objectType="urn:oasis:names:tc:ebxmlregrep:Object
         Type:RegistryObject:ExternalIdentifier"
        registryObject="1.[documentOID]"
        status=""
        value="[documentOID]">
  <urn2:Name>
   <urn2:LocalizedString charset=""
        xml:lang=""
        value="XDSDocumentEntry.uniqueId"/>
  </urn2:Name>
 </urn2:ExternalIdentifier>
</urn2:ExtrinsicObject>
<!--RegistryPackage / SubmissionSet-->
<!--status, El estado del ciclo de vida del Submission
 Set, toma los valores approved o deprecated-->
<urn2:RegistryPackage home="urn:oid:[repository.
 repositoryHomeCommunityId]"
     id="2.[documentOID]"
     lid=""
     objectType="urn:uuid:7edca82f-054d-47f2-a032-
      9b2a5b5186c1"
     status="urn:oasis:names:tc:ebxml-regrep:StatusType:
      [document.status.status]">
 <!--CPOE opcional-->
 <urn2:Slot name="CPOE"
   slotType="">
  <urn2:ValueList>
   <urn2:Value>[orderNumber]</urn2:Value>
  </urn2:ValueList>
 </urn2:Slot>
 <!-- OIDApplication opcional-->
 <urn2:Slot name="OIDApplication"
   slotType="">
  <urn2:ValueList>
   <urn2:Value>[application.oid]</urn2:Value>
  </urn2:ValueList>
 </urn2:Slot>
 <!--Este atributo representa el momento en el cual el
  Submission Set es generado-->
 <urn2:Slot name="submissionTime"
   slotType="">
  <urn2:ValueList>
   <urn2:Value>{{docTimestamp]]</urn2:Value>
  </urn2:ValueList>
 </urn2:Slot>

 <!--Author, es el mismo que se define en el
  ExtrinsicObject-->
 <urn2:Classification classificationNode=""
      classificationScheme="urn:uuid:a7058bb9-b4e4-4307-
```

```
     ba5b-e3f0ab85e12d"
    classifiedObject="2.[documentOID]"
    home=""
    id="cl09"
    lid=""
    nodeRepresentation=""
    objectType="urn:oasis:names:tc:ebxml-regrep:Object
     Type:RegistryObject:Classification"
    status="">
<urn2:Slot name="authorPerson"
  slotType="">
 <urn2:ValueList>
  <urn2:Value>[author.identityValue]^[author.lastName]
   ^[author.firstName]^[author.middleName]^^[author.
   prefix]&amp;[author.identityAssigningAuthority.oid]
   &amp;ISO</urn2:Value>
 </urn2:ValueList>
</urn2:Slot>
<urn2:Slot name="authorInstitution"
  slotType="">
 <urn2:ValueList>
  <urn2:Value>[encounter.institution.name]^^^^^^^^^
   [encounter.institution.oid]</urn2:Value>
 </urn2:ValueList>
</urn2:Slot>
<!--Rol del autor del documento-->
<urn2:Slot name="authorRole"
  slotType="">
 <urn2:ValueList>
  <urn2:Value>[author.role]</urn2:Value>
 </urn2:ValueList>
</urn2:Slot>
<!--Especialidad del autor del documento-->
<urn2:Slot name="authorSpecialty"
  slotType="">
 <urn2:ValueList>
  <urn2:Value>[author.specialty]</urn2:Value>
 </urn2:ValueList>
</urn2:Slot>
<urn2:Slot name="authorLocation"
  slotType="">
 <urn2:ValueList>
  <urn2:Value>[author.dependency]</urn2:Value>
 </urn2:ValueList>
</urn2:Slot>
<urn2:Slot name="authorTelecommunication"
  slotType="">
 <urn2:ValueList>
  <urn2:Value>^[author.phoneUseCode.useCode]^[author.
   phoneEquipmentType.equipmentType]^^^^[author.phone
```

```
    Number]^^</urn2:Value>
  </urn2:ValueList>
 </urn2:Slot>
 <urn2:Name>
  <urn2:LocalizedString charset=""
       xml:lang=""
       value=""/>
 </urn2:Name>
</urn2:Classification>
<!--ContentTypeCode, El codigo que especifica el tipo
 de actividad clinica que dio lugar a la colocacion
 de los documentos en este Submission Set-->
<!--nodeRepresentation, codificacion SNOMED-CT para
 ContentTypeCode-->
<urn2:Classification classificationNode=""
     classificationScheme="urn:uuid:aa543740-bdda-424e-
      8c96-df4873be8500"
     classifiedObject="2.[documentOID]"
     home=""
     id="cl10"
     lid=""
     nodeRepresentation="[encounter.axis3.code]"
     objectType="urn:oasis:names:tc:ebxml-regrep:Object
      Type:RegistryObject:Classification"
     status="">
 <urn2:Slot name="codingScheme"
   slotType="">
  <urn2:ValueList>
   <urn2:Value>[encounter.axis3.codeSystem]
    </urn2:Value>
  </urn2:ValueList>
 </urn2:Slot>
 <urn2:Name>
  <urn2:LocalizedString charset=""
       xml:lang=""
       value="[encounter.axis3.name]"/>
 </urn2:Name>
</urn2:Classification>
<!--PatientId-->
<urn2:ExternalIdentifier home=""
      id="ei03"
      identificationScheme="urn:uuid:6b5aea1a-874d-4603
       -a4bc-96a0a7b38446"
      lid=""
      objectType="urn:oasis:names:tc:ebxmlregrep
       :ObjectType:RegistryObject:ExternalIdentifier"
      registryObject="2.[documentOID]"
      status=""
      value="[patient.institutionIdentityValue]^^^&amp;
       [encounter.institution.assigningAuthority]&amp;
```

```
        ISO">
   <urn2:Name>
    <urn2:LocalizedString charset=""
        xml:lang=""
        value="XDSSubmissionSet.patientId"/>
   </urn2:Name>
  </urn2:ExternalIdentifier>
  <!--SouceId-->
  <urn2:ExternalIdentifier home=""
        id="ei04"
        identificationScheme="urn:uuid:554ac39e-e3fe-47fe
         -b233-965d2a147832"
        lid=""
        objectType="urn:oasis:names:tc:ebxmlregrep:Object
         Type:RegistryObject:ExternalIdentifier"
        registryObject="2.[documentOID]"
        status=""
        value="[encounter.institution.oid]">
   <urn2:Name>
    <urn2:LocalizedString charset=""
        xml:lang=""
        value="XDSSubmissionSet.sourceId"/>
   </urn2:Name>
  </urn2:ExternalIdentifier>
  <!--UniqueId-->
  <urn2:ExternalIdentifier home=""
        id="ei05"
        identificationScheme="urn:uuid:96fdda7c-d067-4183
         -912e-bf5ee74998a8"
        lid=""
        objectType="urn:oasis:names:tc:ebxmlregrep:Object
         Type:RegistryObject:ExternalIdentifier"
        registryObject="2.[documentOID]"
        status=""
        value="[documentOID]">
   <urn2:Name>
    <urn2:LocalizedString charset=""
        xml:lang=""
        value="XDSSubmissionSet.uniqueId"/>
   </urn2:Name>
  </urn2:ExternalIdentifier>
 </urn2:RegistryPackage>
 <urn2:Classification classificationNode="urn:uuid:
  a54d6aa5-d40d-43f9-88c5-b4633d873bdd"
      classificationScheme=""
      classifiedObject="2.[documentOID]"
      home=""
      id="urn:uuid:1d4d08bc-85cc-4596-8fdc-4b5410a6feae"
      lid=""
      nodeRepresentation=""
```

```
            objectType="urn:oasis:names:tc:ebxml-regrep:Object
             Type:RegistryObject:Classification"
            status="">
         <urn2:Slot name=""
           slotType="">
          <urn2:ValueList>
           <urn2:Value/>
          </urn2:ValueList>
         </urn2:Slot>
         <urn2:Name>
          <urn2:LocalizedString charset=""
               xml:lang=""
               value=""/>
         </urn2:Name>
        </urn2:Classification>
        <!--Asociacion entre ExtrinsicObject y RegistryPackage /
         Association-->
        <urn2:Association associationType="urn:oasis:names:tc:
         ebxml-regrep:AssociationType:HasMember"
            home=""
            id=""
            lid=""
            objectType=""
            sourceObject="2.[documentOID]"
            status=""
            targetObject="1.[documentOID]">
         <urn2:Slot name="SubmissionSetStatus"
           slotType="">
          <urn2:ValueList>
           <urn2:Value>Original</urn2:Value>
          </urn2:ValueList>
         </urn2:Slot>
        </urn2:Association>
       </urn2:RegistryObjectList>
     </urn1:SubmitObjectsRequest>
     <urn:Document id="1.[documentOID]">
      <xop:Include href="cid:725023342769"
        xmlns:xop="http://www.w3.org/2004/08/xop/include"/>
     </urn:Document>
    </urn:ProvideAndRegisterDocumentSetRequest>
   </soapenv:Body>
</soapenv:Envelope>

------=_Part_8_819376715.1465835825714
Content-Type: text/xml
Content-Transfer-Encoding: binary
Content-ID: <725023342769>

[Aqui se incluye el CDA visto previamente]
```

------=_Part_8_819376715.1465835825714--

Variable	Req/ Op	Descripción
application.name	O	Nombre del dispositivo o aplicación que genera la información del documento. Se utiliza cuando autor es dispositivo y no persona
application.oid	O	OID de la aplicación que generó la transacción
author.dependency	O	Dependencia o local de la institución donde se generó el acto clínico
author.firstName	R/O	Primer nombre del autor del documento
author.identity-AssigningAuthority.oid	R/O	OID de Assigning Authority correspondiente a author.identityValue. (Salud Digital, 2017a) (Salud Digital, 2017b) Si author.identityValue corresponde al identificador de un dispositivo, este campo es opcional.
author.identityValue	R	Identificador de persona autor o identificador del dispositivo. Ej: número de CI, nro de caja de profesionales. (Salud Digital, 2017b) Puede incluir uno o más identificadores. En caso que autor sea un dispositivo de laboratorio, debe tener la siguiente estructura: 2.16.858.2.[IdInstitución].72771.[ConsecutivoInterno] [Guía para la Gestión de OID.pdf] 2.16.858: nodo raiz para uso de un OID en Uruguay. 2: identifica a objetos dentro de Uruguay (0 para organizaciones y 1 para personas) [IdInstitución]: Id de la institución que genera el documento asignado en Nodo de organizaciones. (Ej: 10003153 corresponde al Id de Cooperativa Médica de Rocha) 72771: Objeto Sistema Información de Laboratorios [ConsecutivoInterno]: Consecutivo Interno dentro de la organización para numerar el sistema de laboratorio
author.lastName	R/O	Primer apellido del autor del documento
author.lastName2	O	Segundo apellido del autor del documento
author.middleName	O	Segundo nombre del autor del documento

Variable	Req/ Op	Descripción
author.phoneNumber	O	Es utilizado para la notificación de certificaciones médicas y enfermedades de notificación obligatoria
author.phoneUseCode.useCode	O	Tipo de comunicación a utilizar para la notificación de certificaciones médicas y enfermedades de notificación obligatoria. Opciones: BP(Beeper), CP(Cellular Phone), FX(Fax), Internet(Internet Address: Use Only If Telecommunication Use Code Is NET), MD(Modem), PH(Telephone), TDD(Telecommunications Device for the Deaf), TTY(Teletypewriter), X.400(X.400 email address: Use Only If Telecommunication Use Code Is NET)
author.phoneUseCode.useCode	O	Medio de comunicación a utilizar para la notificación de certificaciones médicas y enfermedades de notificación obligatoria. opciones: ASN(Answering Service Number), BPN(Beeper Number), EMR(Emergency Number), NET(Network (email) Address), ORN(Other Residence Number), PRN(Primary Residence Number), PRS(Personal), VHN(Vacation Home Number), WPN(Work Number)
author.prefix	O	Prefijo del nombre del autor del documento
author.role	O	Rol del autor del documento
author.specialty	O	Especialidad del autor del documento
document.attachmentB64	R	Representación en base 64 debe realizarse según codificación definida en RFC 2045. (N. Freed, N. Borenstein, 2021)
document.contentType.cdaNonXMLBodyMediaType	O	Codificación de los datos enviados dentro del CDA. Identifica un método para interpretar o presentar los datos. Opciones: image/gif, image/tiff, text/rtf, application/pdf, image/g3fax, text/html, image/jpeg, image/png, y text/plain(default)
document.contentType.cdaNonXMLBodyRepresentation	O	Indica si el contenido es un texto, o una codificación en base 64.

Variable	Req/ Op	Descripción
document.document-Description	O	Descripción asociada al documento que se está generando. Aunque name y description son campos opcionales según los perfiles IHE, los mismos son altamente recomendables.
document.document-Name	R	Título asociado al documento.
document.format-Code.code	R	Código que determina si el documento es un PDF embebido en un CDA o texto embebido en un CDA
document.format-Code.codeSystem	R	OID del sistema de codificación del código document.formatCode.code
document.format-Code.name	R	Descripción del código document.formatCode.code. Documento PDF escaneado o documento de texto escaneado
document.language. code	R	Idioma en que el documento está redactado. Los valores se toman de la codificación de ISO 639-1 (Library of Congress, 2017)
document.security-Code.code	R	Código de confidencialidad del documento. Opciones: N (confidencialidad normal), R (confidencialidad restringida), V (confidencialidad muy restringida).
document.security-Code.codeSystem	R	OID de la codificación del código document.securityCode.code
document.security-Code.name	R	Nombre del código de confidencialidad del documento. Opciones: Normal (confidencialidad normal), Restricted (confidencialidad restringida), Very Restricted (confidencialidad muy restringida). En el ejemplo estaba Normal.
document.status. status	R	Status del documento. Valores posibles: “Approved” o “Deprecated”.

Variable	Req/ Op	Descripción
documentOID	R	OID del documento. Representación numérica universal y única que permite la identificación inequívoca del documento (Salud Digital, 2017a) Se compone de los siguientes campos: 2.16.858.2.[institucionID].67430.[docTimestamp].[docConsecutivoInterno].[docNroHIS] 2.16.858: nodo raiz para uso de un OID en Uruguay. 2: identifica a objetos dentro de Uruguay (0 para organizaciones y 1 para personas) [institucionID]: Id registrada en UNAOID de la institución que genera el documento asignado en Nodo de organizaciones. (Ej: 10003153 corresponde al Id de Cooperativa Médica de Rocha) 67430: Objeto Documento Historia Clínica Electrónica. Fijo para HCEN [docTimestamp]: Fecha de creación del documento en formato AAAAMMDDHHMMSS - ISO 8601:2004 (International Organization for Standardization, 2023) [docConsecutivoInterno]: Consecutivo Interno generado en el HIS o aplicación que genera el CDA [docNroHIS]: Id HIS o aplicación que genera el documento
documentSetId	R	OID de la primer versión del documento. Es obligatorio cuando el documento tiene más de una versión.
documentVersion	R	Número entero utilizado para versionar las actualizaciones de un documento. Es de uso obligatorio cuando el documento tiene más de una versión.
encounter.axis1.code	R	Código LOINC (Regenstrief Institute, 2023) del eje 1 de ontología de documento. Tipo de documento general. Valores provistos por Salud Digital
encounter.axis1.codeSystem	R	OID del sistema de codificación de encounter.axis1.code. En HCEN se utilizan códigos LOINC
encounter.axis1.codeSystemName	R	Nombre correspondiente al OID encounter.axis1.codeSystem. En HCEN se utiliza LOINC

Variable	Req/ Op	Descripción
encounter.axis1. name	R	Descripción del eje 1 de ontología de documentos correspondiente a código encounter.axis1.code
encounter.axis2.code	R	Código LOINC (Regenstrief Institute, 2023) del eje 2 de ontología de documento. Especifica el tipo preciso de documento que se está generando. Valores provistos por Salud Digital.
encounter.axis2.codeSystem	R	OID del sistema de codificación de encounter.axis2.code. En HCEN se utilizan códigos LOINC
encounter.axis2.codeSystemName	R	Nombre correspondiente al OID encounter.axis2.codeSystem. En HCEN se utiliza LOINC
encounter.axis2. name	R	Descripción del eje 2 de ontología de documentos correspondiente a código encounter.axis2.code
encounter.axis3.code	R	Código SNOMED-CT (HL7, 2023a) del eje 3 de ontología de documentos. Especifica la especialidad clínica realizada donde se generó el documento. Valores provistos por Salud Digital
encounter.axis3.codeSystem	R	OID del sistema de codificación de encounter.axis3.code. En HCEN se utilizan códigos SNOMED-CT
encounter.axis3.codeSystemName	R	Nombre correspondiente al OID encounter.axis3.codeSystem. En HCEN se utiliza SNOMED-CT
encounter.axis3. name	R	Descripción del eje 3 de ontología de documentos correspondiente a código encounter.axis3.code
encounter.endDate	R	Fecha de fin del evento clínivo. Formato: AAAAMMDDHHMMSS

Variable	Req/ Op	Descripción
encounter.institution.assigningAuthority	R	OID de Assigning Authority (HIS) correspondiente a patient.institutionIdentityValue (Salud Digital, 2017a) (Salud Digital, 2021a) Debe tener la siguiente estructura: 2.16.858.2.[IdInstitución].72768.[ConsecutivoInterno] 2.16.858.2: nodo raiz para objetos en Uruguay. [IdInstitución]: Id de la institución que genera el documento asignado en Nodo de organizaciones. 72771: Objeto Sistema Información de Laboratorios [ConsecutivoInterno]: Consecutivo Interno dentro de la organización para numerar el sistema
encounter.institution.healthCareFacilityType.code	R	Código de la clasificación de SNOMED del tipo de centro de salud en la cual se desarrolló el acto clínico documentado
encounter.institution.healthCareFacilityType.codeSystem	R	OID del sistema de codificación del código encounter.institution.healthCareFacilityType.code. En HCEN se utiliza códigos SNOMED-CT
encounter.institution.healthCareFacilityType.name	R	Nombre correspondiente al código encounter.institution.healthCareFacilityType.code
encounter.institution.name	R	Nombre de la institución que representa al autor del documento
encounter.institution.oid	R	OID identificador de la institución que representa al autor del documento. OID definido por UNAOID (AGESIC, 2023b)
encounter.startDate	R	Fecha de inicio del evento clínico. Formato: AAAAMMDDHHMMSS
orderNumber	O	Identifica el acto asistencial mediante el número de orden (único) generado dentro de la institución o prestador
patient.birthdate	O	Fecha de nacimiento del paciente. Formato: AAAAMMDD
patient.firstName	R	Primer nombre del paciente
patient.gender.code	R	Código de género del paciente. Opciones: 0-Desconocido, 1-Masculino, 2-Femenino, 9-No aplica (AGESIC, 2023a)
patient.gender.codeSystem	R	OID del sistema de codificación de género de persona. (AGESIC, 2023a)

Variable	Req/ Op	Descripción
patient.gender.name	R	Nombre del código de género del paciente. Opciones: 0-Desconocido, 1-Masculino, 2-Femenino, 9-No aplica
patient.identityAssigningAuthority.oid	R	OID del tipo de documento a nivel nacional del paciente (Salud Digital, 2017a) (Salud Digital, 2017b)
patient.identityValue	R	Documento a nivel nacional del paciente. Ej: número de CI (Salud Digital, 2017b)
patient.institutionIdentityValue	R	Identificador del paciente dentro de la institución.
patient.lastName	R	Primer apellido del paciente
patient.lastName2	O	Segundo apellido del paciente
patient.middleName	O	Segundo nombre del paciente
repository.custodianOrganizationName	O	Nombre de la institución que custodia el documento
repository.custodianOrganizationOID	R	OID registrado en UNAOID de la institución que custodia el documento
repository.endpointHostname	R	Hostname del endpoint del respositorio XDS/Appliance donde se enviará el CDA
repository.endpointPort	R	Puerto del hostname del respositorio XDS/Appliance donde se enviará el CDA
repository.oid	R	OID del repositorio donde se almacenará el documento CDA
repository.repositoryHomeCommunityId	R	OID asociado a una comunidad en la cual se puede acceder al documento. Es fijo para Uruguay con valor "2.16.858.2.10000675.73183.1"
signature	R	Firma electrónica del CDA

Tabla M.1: Especificación de parámetros definidos en la generación del CDA, la transacción ITI-41 y el mensaje MTOM.

Anexo N

Aplicación desarrollada para pruebas: Demo DOCLUY

Para poder visualizar los CDA generados por la solución, debemos consultarlos al servidor XDS con los servicios que este provee. Para comprobar la correcta generación del CDA y el envío al Repositorio XDS local, desarrollamos una pequeña aplicación que consume los servicios del XDS local que consulta la lista de documentos CDA registrados de una persona y luego permite consultar un CDA específico.

Para visualizar el CDA se utilizó la *Hoja de estilo para visualización de CDA* (Salud Digital, 2021b) provista por Salud Digital para transformar el CDA y visualizarlo como se visualizaría en el sitio de HCEN.

En la figura N.1 se muestra la página inicial de la aplicación, donde se ingresa el identificador interno del paciente para buscar los CDA que tiene registrados.

En la figura N.2 se muestra el resultado de búsqueda de documentos del paciente. Se puede seleccionar un CDA para visualizarlo.

La figura N.3 muestra el CDA seleccionado como se vería en el sitio de HCEN.

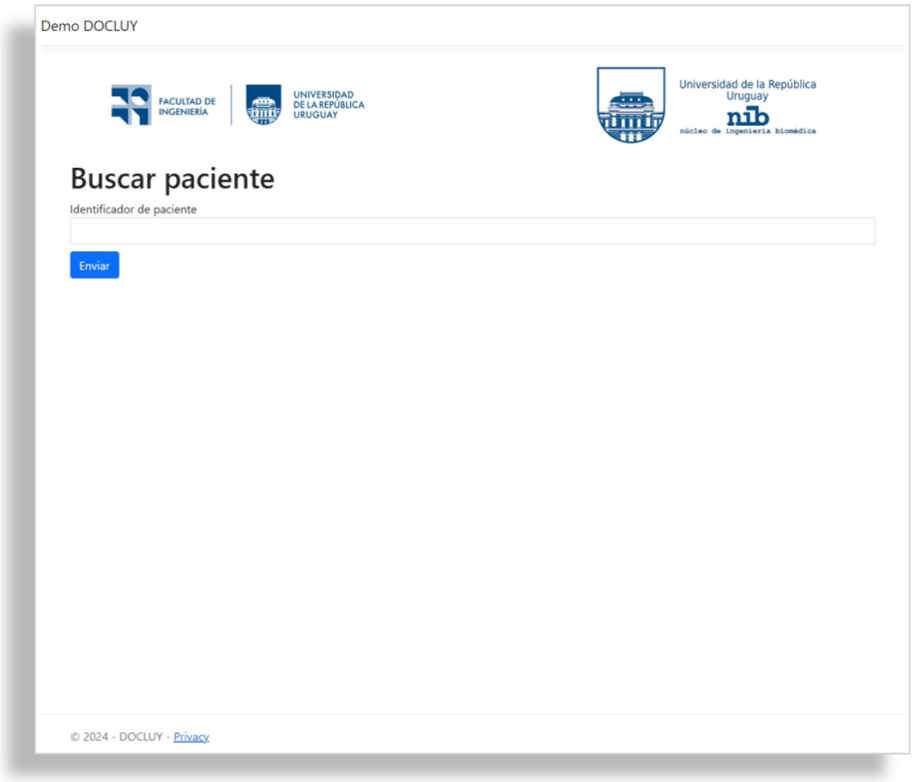


Figura N.1: Demo DOCLUY, página inicial de la aplicación.

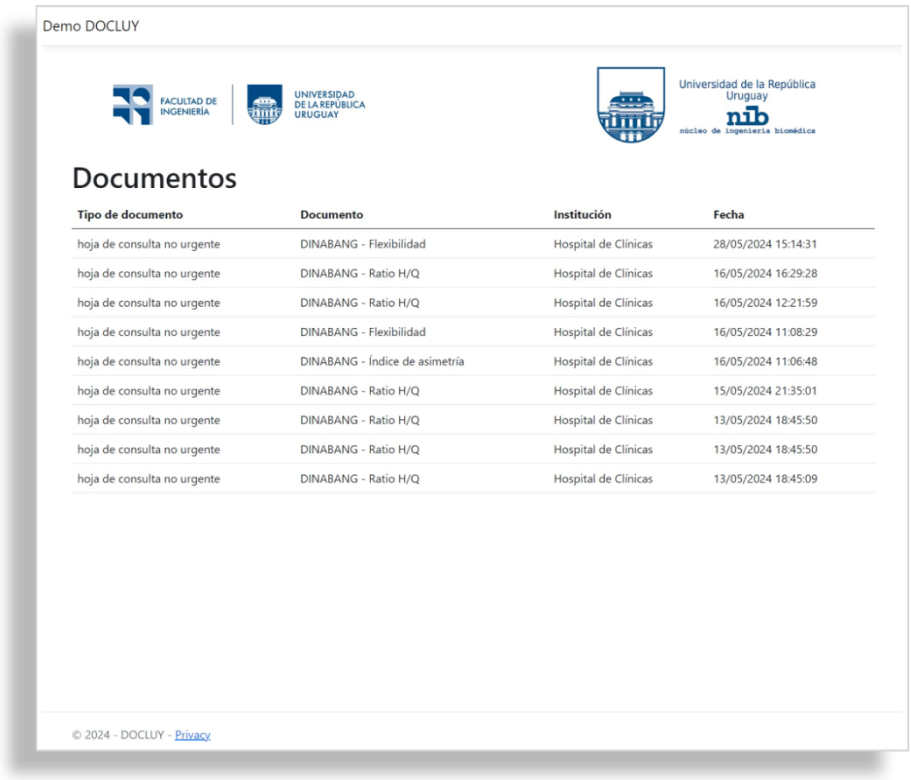


Figura N.2: Demo DOCLUY, lista de documentos CDA registrados en Repositorio XDS local de una persona.

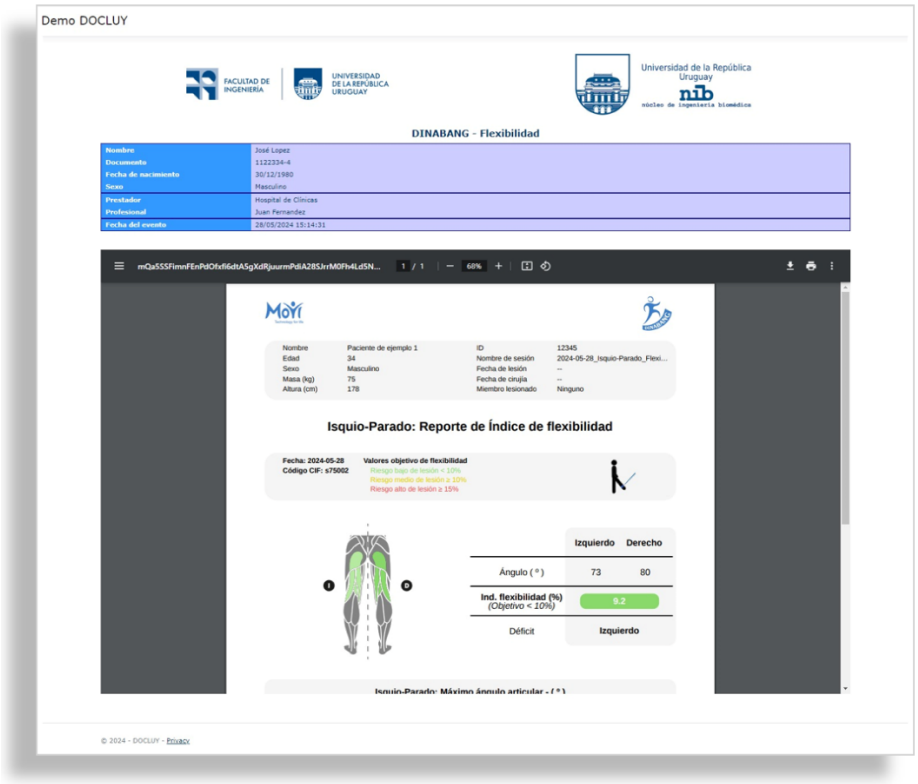


Figura N.3: Demo DOCLUY, visualización de un CDA.

Anexo Ñ

Ingeniería de muestra 2022

El equipo de proyecto ha tenido el privilegio de participar en Ingeniería de Muestra. Este evento, organizado por la Facultad de Ingeniería de la Universidad de la República y su Fundación Julio Ricaldoni, representa un momento destacado en el calendario académico y es un testimonio del compromiso de la facultad con la excelencia en la educación y la promoción de la investigación. Durante nuestra participación, nuestro equipo ha tenido la oportunidad de compartir nuestro proyecto con estudiantes, profesores y profesionales de la Ingeniería. Además de contribuir activamente como expositores en este evento, hemos tenido el honor de colaborar estrechamente con otros participantes y asistentes, lo que ha enriquecido nuestro propio conocimiento y ha fomentado valiosas conexiones.

En la figura Ñ.1 ilustramos el póster presentado en el evento.

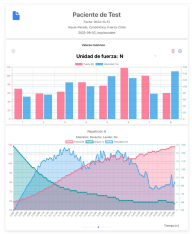
DINABANG-CDA: Registro de rehabilitación de miembros inferiores en la Historia Clínica Electrónica Nacional

G. Capote, L. Sosa, C. Trías, R. Barboza,
J. Domínguez, D. Santos y F. Simini
Núcleo de Ingeniería Biomédica de las Facultades de Medicina e Ingeniería - IIE
MOVI Technology For Life, Uruguay

Ingeniería deMuestra
14 y 15 de octubre de 2022

Dinabang para la rehabilitación

DINABANG es un **dispositivo portable y liviano** que hace un seguimiento durante la **rehabilitación y el entrenamiento deportivo**, calculando parámetros de interés como fuerza, ángulo y velocidad para mostrarlos en tiempo real. A través de una aplicación para dispositivos móviles, el profesional obtiene una realimentación durante el ejercicio, y **herramientas para analizar la evolución del paciente** a través de sucesivas sesiones. Cuenta a su vez con funcionalidades que contribuyen a la **prevención de lesiones y la evaluación del alta médica**.



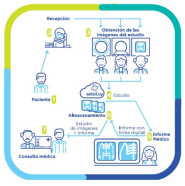
Equipo clínico conectado a la Historia Clínica Electrónica

El objetivo principal de este proyecto es **disponibilizar información de uso clínico generada por dispositivos médicos en la Historia Clínica Electrónica Nacional**. En particular DINABANG, pero no limitado al mismo.

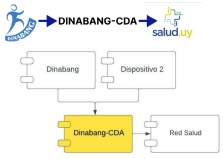
Esto permitirá generar i**nformes de sesiones** de diagnóstico y seguimiento de entrenamiento de miembros inferiores realizados con el instrumento DINABANG utilizando **estándares internacionales** de documentación clínica y **enviarlos automáticamente** a la Historia Clínica Nacional

Norma Salud.uy

La Plataforma Salud.uy interconecta a todos los prestadores de salud del país a través de Red Salud. Provee componentes de infraestructura que brindan soporte al funcionamiento de los Servicios HCEN así como a otros servicios y aplicaciones. Red Salud es una red privada de alta velocidad que interconecta a todos los prestadores de salud del país, tiene alta disponibilidad, es escalable, segura, auditada por CERTuy y monitoreada 24/7.



Dinabang-CDA para el seguimiento de la evolución



DINABANG-CDA incorpora componentes de software que proporcionan la interacción necesaria para interconectar los datos generados por el **dispositivo** con la **integración a Red Salud** en los diferentes prestadores de salud. Además permite la integración de multipledispositiovs (no solo DINABANG), genera informes extra de evolución, se encarga de la identificación de médico y del paciente y exporta los reportes en los formatos requeridos por el receptor (CDA para salud.uy)

Fundación Julio Ricaldoni
FACULTAD DE INGENIERÍA
UNIVERSIDAD DE LA REPÚBLICA URUGUAY

Figura Ñ.1: Póster presentado en Ingeniería de Muestra

Anexo O

Integración de sistema SISENF con DOCLUY

Para la integración de SISENF con DOCLUY se utilizó un componente de software que funciona como cliente, invocando las operaciones provistas por el API para poder generar y subir a Salud Digital un CDA con un pdf adjunto que contiene información de la consulta. Todas las solicitudes a los métodos provistos por la API requieren autenticación a través de una API Key enviada en un header HTTP.

SISENF interactúa con DOCLUY luego de guardar la consulta de enfermería en la base de datos. El proceso consiste en generar un pdf con los datos básicos de la consulta, obtener el repositorio asociado a SISENF y luego subir el documento a dicho repositorio.(G. Bonilla, M. Rocanova, 2023)

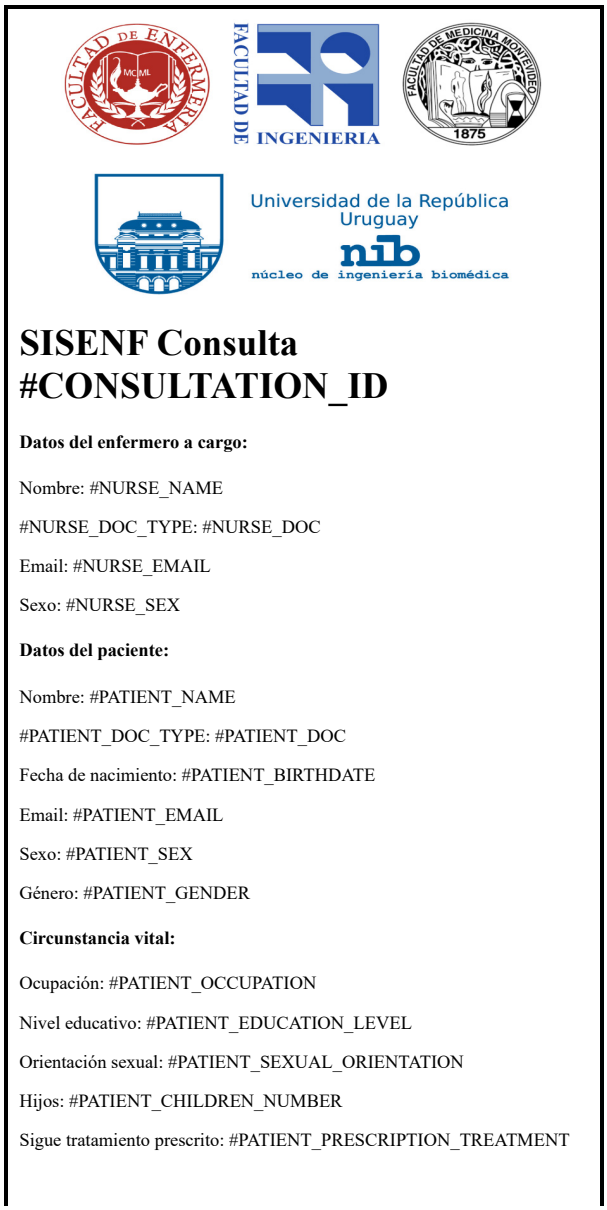


Figura O.1: PDF enviado por SISENF a DOCLUY

Anexo P

Integración de dispositivo DINABANG con DOCLUY

La empresa MOVI(MOVI, 2023) realizó un desarrollo propio creando una nueva versión de su aplicación, done se consume la API de DOCLUY para generar documentos CDA que contengan sus reportes y enviarlos a la HCEN.

A continuación, se presenta con capturas de pantalla la secuencia de pasos que se realiza en DINABANG para realizar el envío de un reporte a HCEN a través de DOCLUY.

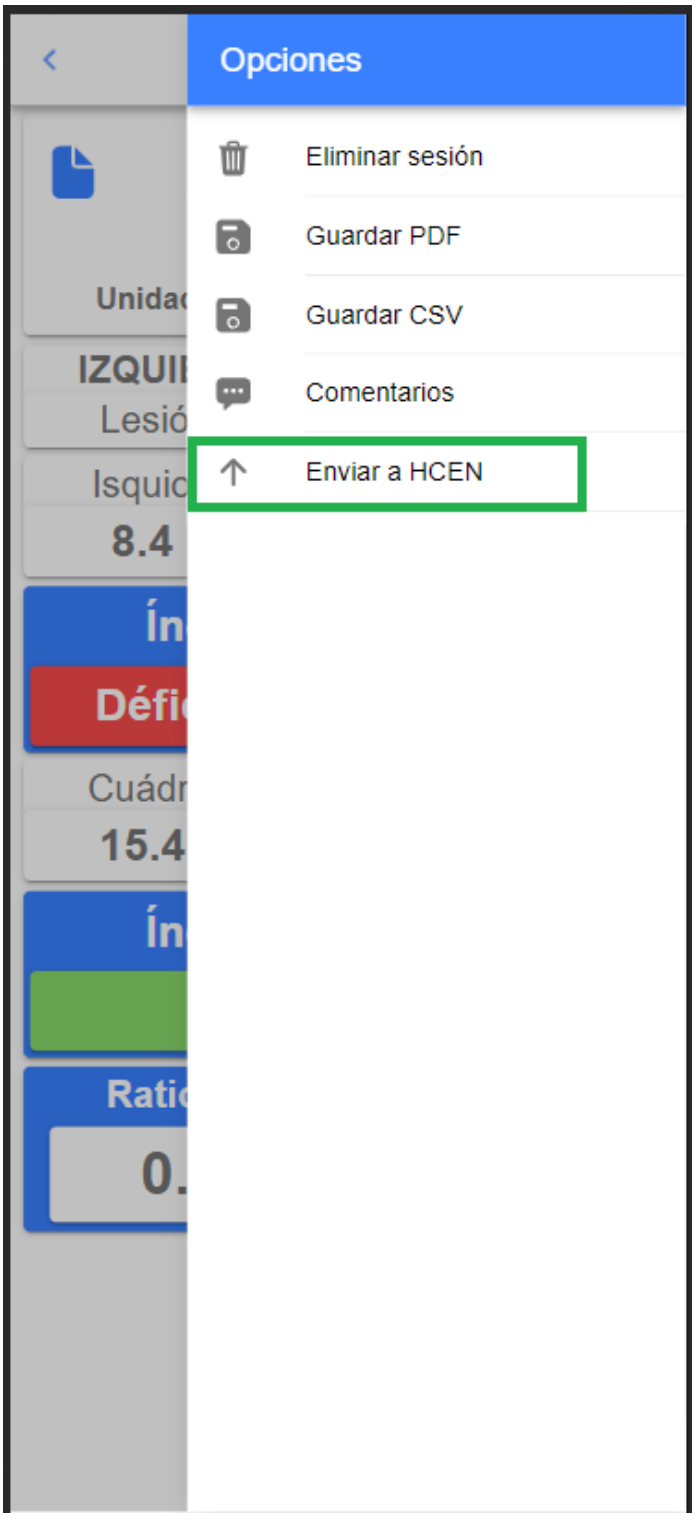


Figura P.1: Secuencia en DINABANG: Paso 1
Se selecciona enviar un reporte a HCEN.

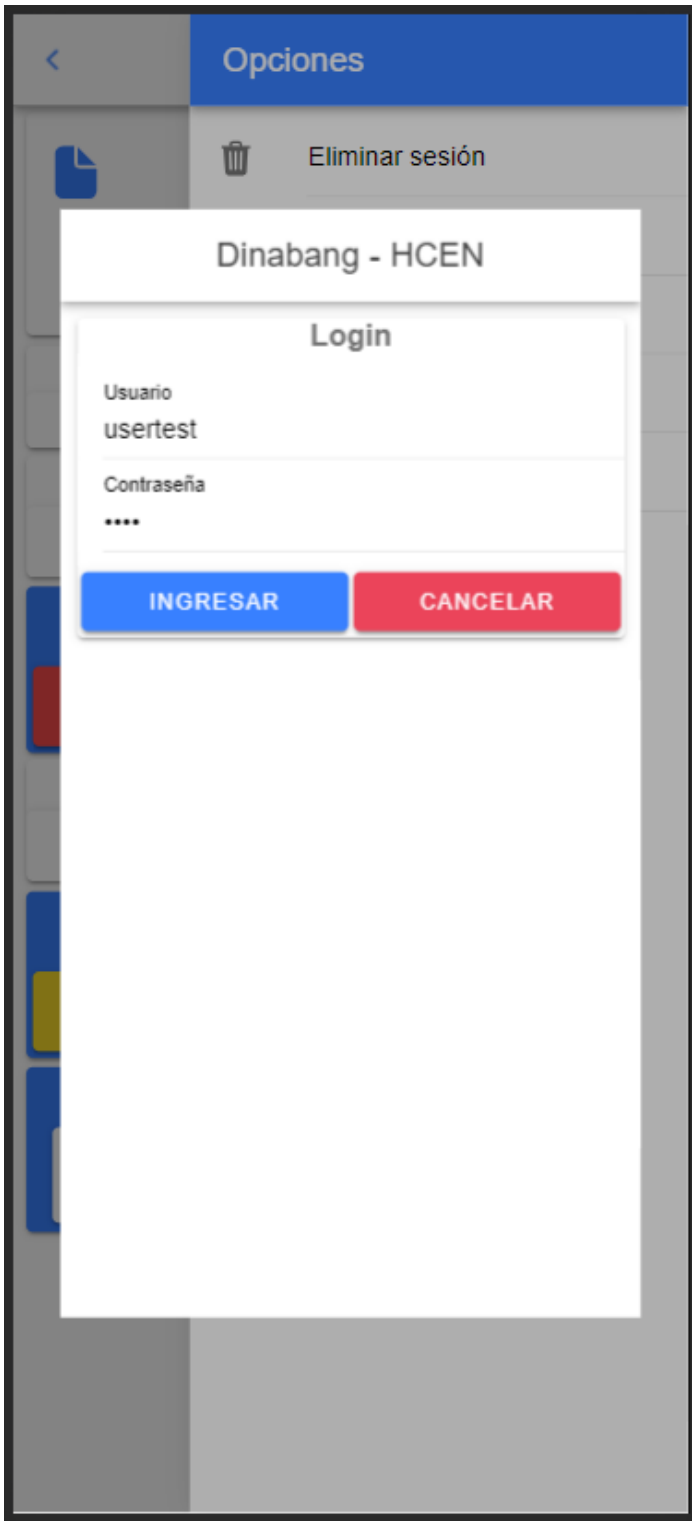


Figura P.2: Secuencia en DINABANG: Paso 2
Se autentica el usuario nuevamente para confirmar que tiene permisos para enviar un reporte a HCEN.

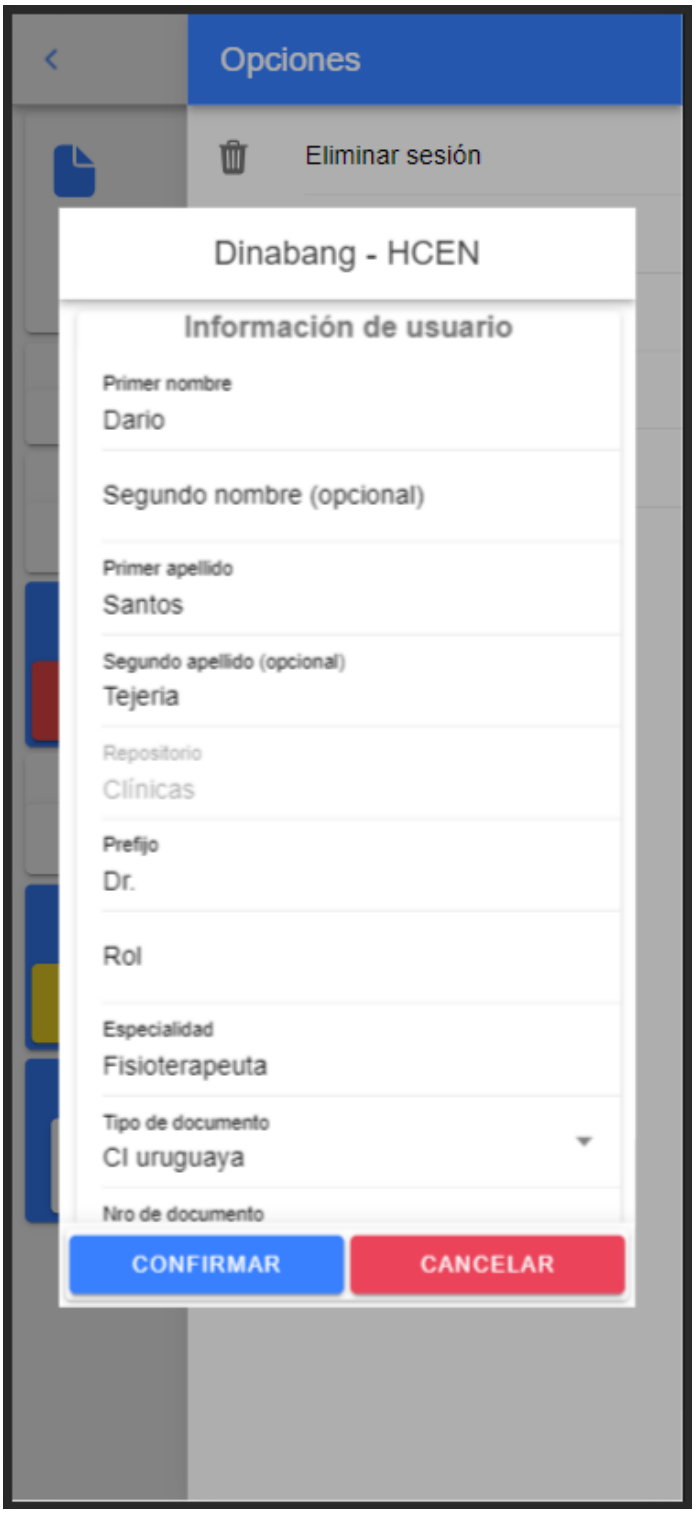


Figura P.3: Secuencia en DINABANG: Paso 3
Ingreso de información requerida para el envío a HCEN.

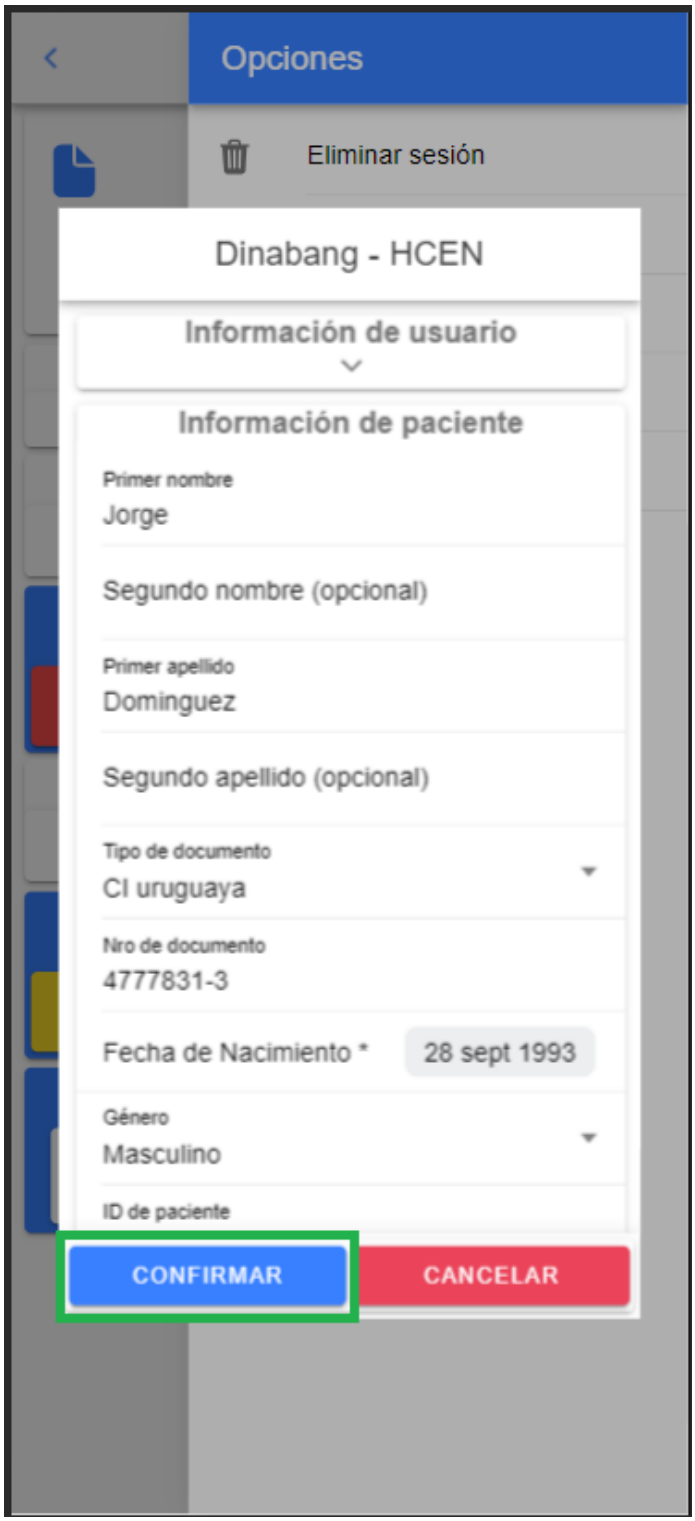


Figura P.4: Secuencia en DINABANG: Paso 4
Se inicia el envío del reporte.

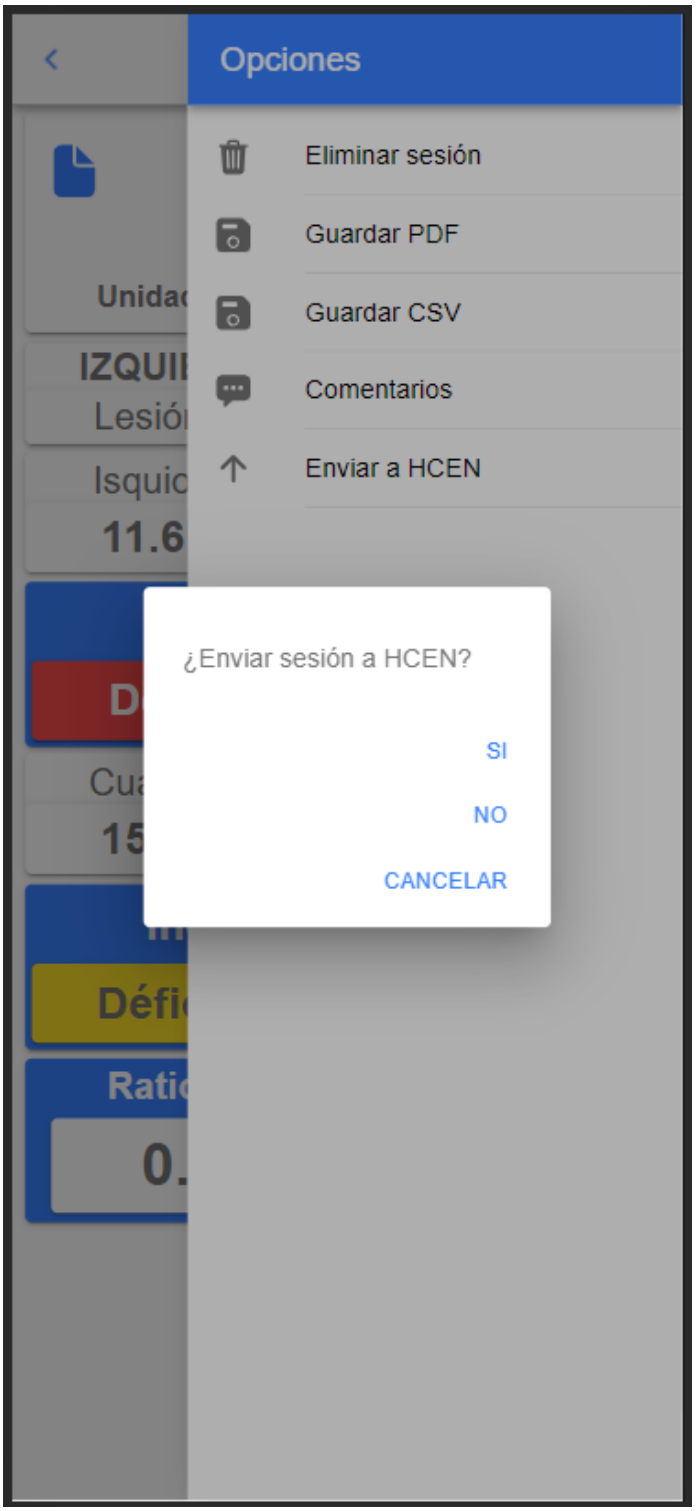


Figura P.5: Secuencia en DINABANG: Paso 5
Se confirma el envío del reporte.

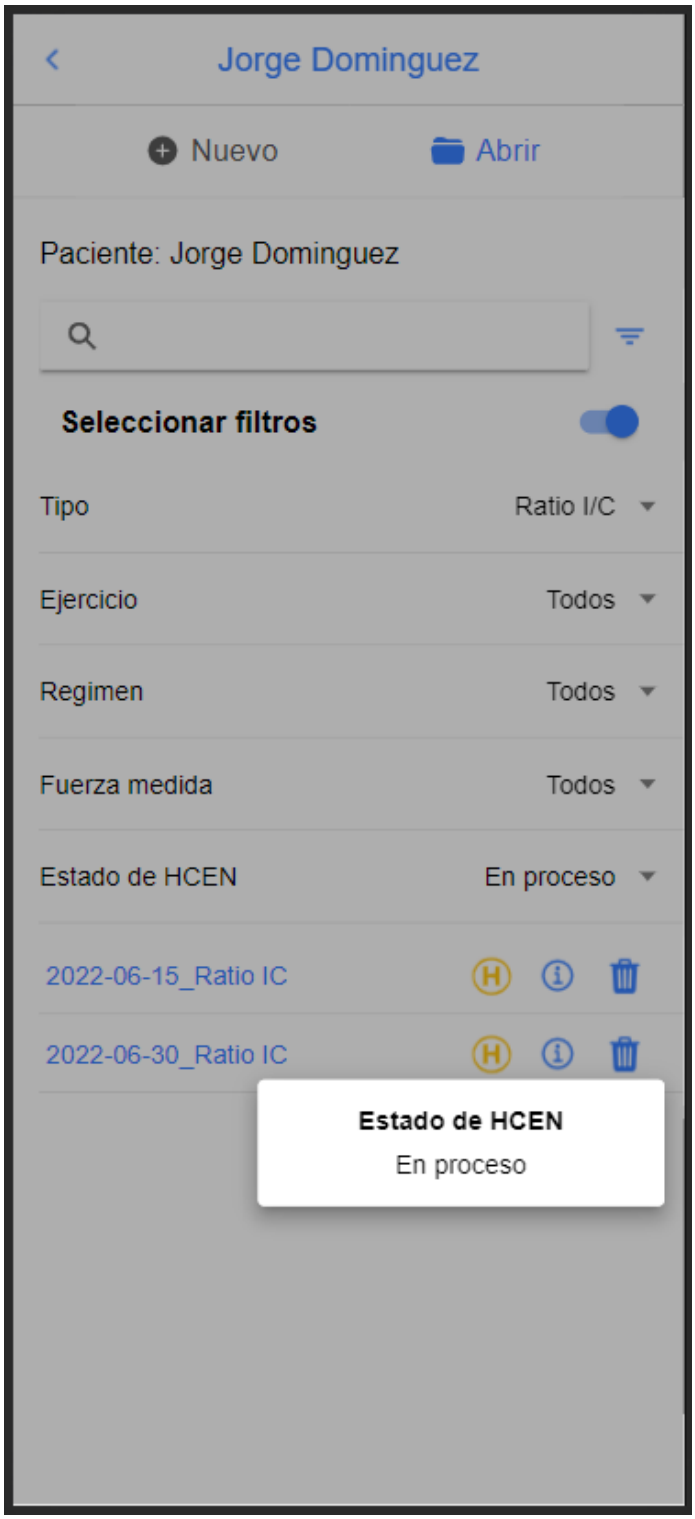


Figura P.6: Secuencia en DINABANG: Paso 6
Estado del envío

P.0.1. Secuencia alternativa donde se intenta enviar un documento que ya fue enviado previamente

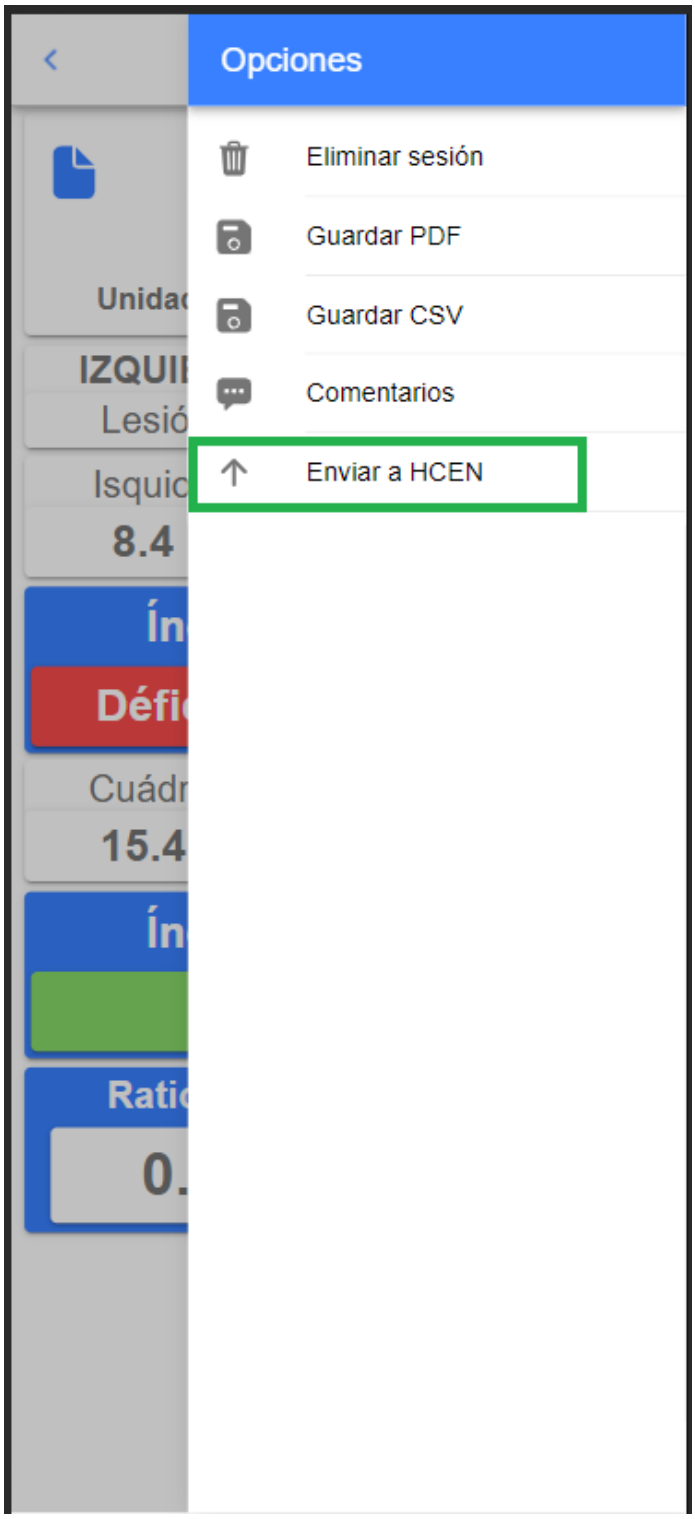


Figura P.7: Secuencia alternativa en DINABANG: Paso 1
Se selecciona enviar un reporte a HCEN.

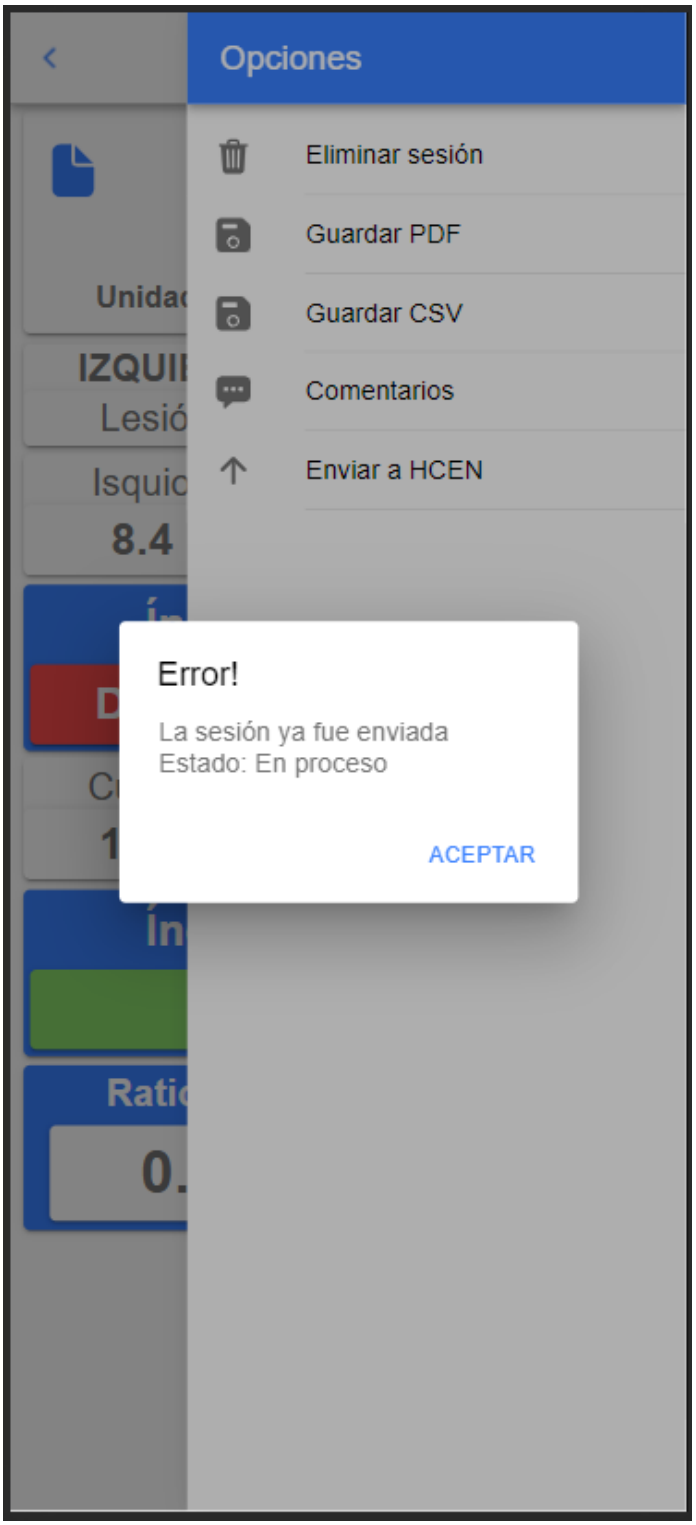


Figura P.8: Secuencia alternativa en DINABANG: Paso 2
Se notifica que la sesión ya enviada y se encuentra en proceso.