

**UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
VALPARAÍSO - CHILE**



**“ANÁLISIS, DISEÑO Y DESARROLLO DE UN PROTOTIPO DE
UN MÓDULO DE GESTIÓN DE MEMORANDOS PARA LA
INTRANET DEL DEPARTAMENTO DE INFORMÁTICA DE LA
UTFSM”**

MAXIMILIANO JOSÉ BOMBIN SANHUEZA

**MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN INFORMÁTICA**

**Profesor Guía: Carlos Castro
Profesor Correferente: Luis Hevia**

agosto de 2024

Dedicatoria

Le dedico el trabajo de esta memoria a todas las personas que fueron parte de ella y que me ayudaron a desarrollarla. Pero, por sobre todo, le dedico esta memoria a mi familia: a mi Madre, mi Padre, mi Hermano y mis Hermanas, quienes nunca me han dejaron de apoyar y creer en mí, a pesar de todas las dificultades y tiempo transcurrido.

AGRADECIMIENTOS

Como mencionado en la dedicatoria, agradezco a mi familia: mi Madre, mi Padre, mi Hermano y mis Hermanas, por el cariño y apoyo que me han mostrado siempre y que fue crucial en juntar las fuerzas para lograr poner término de esta memoria y a mi etapa universitaria.

También quiero agradecer a mi profesor guía, Don Carlos Castro, por aceptarme como su memorista en todo este tiempo, siempre atento a mi situación y dispuesto a ayudarme y a continuar el desarrollo.

También quiero agradecer al DI por permitirme y apoyarme en el desarrollo de esta memoria. Sobre todo, quiero agradecer a las Unidades de Dirección y Pregrado, las que han sido las más interesadas y que fueron cruciales en mantener el proyecto andando. De estas unidades un especial agradecimiento a Ángela Miranda, Érika Guzmán, José Miguel González, Marcela Maldonado y el Jefe de Carrera Pedro Godoy, sin su apoyo, interés y gran disposición, esta memoria no se hubiera logrado completar.

RESUMEN

Actualmente, muchas organizaciones están digitalizando sus procesos y avanzando en su transformación digital.

El Departamento de Informática (DI) de la UTFSM no es una excepción. Cuenta con un portal web, la Intranet, para ayudar a sus empleados a realizar sus tareas administrativas. El DI busca activamente áreas que se beneficiarían de la digitalización.

Un proceso aún no digitalizado es el uso y la gestión de Memorandos, un proceso que, al analizarse, reveló oportunidades de mejora, muchas de las cuales fueron solicitadas por los propios miembros del DI.

Estas oportunidades de mejora inspiran esta tesis. Corresponde a la oferta de un Sistema de Gestión de Memorandos Digital que facilita la creación, el envío, el almacenamiento, la autorización y la recuperación de Memorandos. El sistema se propone como un módulo especializado de la Intranet, digitalizando y centralizando la gestión en este portal web.

El objetivo principal es facilitar el proceso para los colaboradores del DI, con el usuario como el centro y guía del desarrollo, facilitando su trabajo y contribuyendo a la gestión del DI.

Esta tesis comienza con la investigación del contexto, luego presenta la solución propuesta, describiendo el diseño, los requisitos y los componentes; a continuación, detalla las funcionalidades de la solución. Finalmente, se ofrecen las conclusiones de la experiencia.

Palabras Claves-Memorando, transformación digital, gestión, sistema, en línea

ABSTRACT

Currently, many organizations are digitizing their processes and moving forward in their digital transformation.

UTFSM's Informatics Department (DI) is no exception. It has a web portal, The Intranet, to help its employees perform their administrative tasks. The DI is actively looking for areas that would benefit from digitization.

One process that has not yet been digitized is the use and management of memos, a process that upon analysis revealed several opportunities for improvement, many of which were requested by DI members themselves.

These opportunities for improvement inspire this thesis. It corresponds to the offer of a digital memorandum management system that facilitates the creation, sending, storage, authorization and retrieval of memorandums. The system is proposed as a specialized module of the Intranet, digitizing and centralizing the management in this web portal.

The main objective is to facilitate the process for DI collaborators, with the user as the guiding center of development, facilitating their work and contributing to the management of the DI.

This thesis begins with the research of the context, then it presents the proposed solution, describing the design, the requirements and the components; then it details the functionalities of the solution. Finally, the conclusions of the experience are offered.

Keywords- *memorandum, digital transformation, management, system, online*

GLOSARIO

DI: Departamento de Informática

I, T & S: Infraestructura y Tecnología; o Infraestructura, Tecnología y Servicios

UTFSM: Universidad Técnica Federico Santa María

FES: Firma Electrónica Simple

Rails o RoR: Ruby on Rails

MVC: Model-View-Controller

ÍNDICE DE CONTENIDO

Agradecimientos	III
Resumen.....	IV
Abstract	V
Glosario	VI
Índice de Contenido	VII
Índice de Figuras	XI
Índice de Tablas.....	XII
INTRODUCCIÓN.....	14
Capítulo 1: DEFINICIÓN DEL PROBLEMA.....	16
1.1 Contexto.....	16
1.1.1 El Departamento de Informática (DI)	16
1.1.2 Memorandos en el DI.....	17
1.1.3 Descripción del Proceso Actual de Gestión de Memorandos.....	20
1.1.4 Diagnóstico.....	21
1.1.5 Fortalezas del Sistema Actual	23
1.1.6 Conclusión	24
1.2 Objetivos	24
1.3 Alcances de la Solución	25
1.4 Resultados Esperados	25
Capítulo 2: ESTADO DEL ARTE.....	26
2.1 Marco Conceptual.....	26
2.1.1 Filosofía Human Centered Design y Framework Design Thinking	26
2.1.2 Patrón de Diseño Model-View-Controller (MVC)	27
2.1.3 Ruby on Rails (RoR)	28
2.1.4 Asociaciones Principales de Active Record en Ruby On Rails.....	32
2.1.4.1 Polimorfismo	33
2.2 Tecnologías y Librerías Parte del Módulo.....	35
2.2.1 Summernote	35

2.2.2 WKHTMLTOPDF (WK HTML To PDF)	35
2.2.3 Wicked_pdf	36
2.2.4 Active Storage	36
2.2.5 CanCanCan	42
2.2.6 El Portal de la Intranet del Departamento de Informática	48
Capítulo 3: PROPUESTA DE SOLUCIÓN	51
3.1 Descripción General de la Solución.....	51
3.2 Diseño.....	52
3.2.1 Requerimientos.....	52
3.2.1.1 Criterios.....	52
3.2.1.2 Listado de Requerimientos	53
3.2.2 Diagrama de Entidades y Relaciones de Base de Datos	60
3.2.3 Estructura y Campos de un Memorando.....	62
3.2.3.1 Logo de la Universidad.....	62
3.2.3.2 Número Correlativo	62
3.2.3.3 Destinatarios	63
3.2.3.4 Emisor	63
3.2.3.5 Fecha de Creación	64
3.2.3.6 Fecha de Remisión o Confirmación	64
3.2.3.7 Fecha de Autorización.....	64
3.2.3.8 Asunto	64
3.2.3.9 Cuerpo Mensaje	64
3.2.3.10 Documentos Adjuntos (Opcional).....	65
3.2.3.11 Información de Contacto	65
3.2.4 Actores o Perfiles de Usuario.....	66
3.2.4.1 Redactores	66
3.2.4.2 Redactor de Dirección	68
3.2.4.3 Emisor	70
3.2.4.4 Emisor de Dirección	72
3.2.4.5 Mantenedores y Administradores	74

3.2.4.6 Huésped	74
3.2.5 Contactos (Memo Contacto).....	74
3.2.5.1 Nombre Completo	75
3.2.5.2 Cargo	75
3.2.5.3 Departamento.....	75
3.2.5.4 UID (User ID)	75
3.2.5.5 Uso de los Contactos.....	75
3.2.5.6 Permisos de Actores Sobre los Contactos	75
3.3 Implementación	77
3.3.1 Contactos	77
3.3.1.1 Crear y Editar Contacto	77
3.3.1.2 Caso de Uso - Ver y Listar Contacto	80
3.3.1.3 Eliminar Contacto.....	80
3.3.2 Creación y Edición	81
3.3.2.1 Casos de Uso – Creación Memorando	82
3.3.2.2 Caso de uso – Editar Memorando.....	90
3.3.2.3 Diagrama de Secuencia – Crear Memorando	90
3.3.2.4 Diagrama de Secuencia - Editar Memorando	93
3.3.2.5 Visualización.....	93
3.3.2.6 Caso de Uso – Visualización de Memorando.....	94
3.3.3 Plantillas	94
3.3.3.1 Caso de Uso – Crear Plantilla	95
3.3.3.2 Caso de Uso – Crear Borrador a partir de Plantilla	96
3.3.3.3 Diagrama de Secuencia – Crear Borrador a partir de Plantilla	96
3.3.4 Eliminación de Memorando.....	97
3.3.4.1 Eliminación de Memorando.....	97
3.3.5 Remisión de Memorando, Más Envío Notificación al Emisor.....	98
3.3.5.1 Caso de Uso – Remisión de Memorando.....	98
3.3.5.2 Diagrama de Secuencia – Remisión de Memorando	100
3.3.6 Adición de Firma Electrónica Simple.....	100

3.3.6.1 Caso de Uso Agregar Firma Electrónica Simple	101
3.3.7 Autorización Digital	102
3.3.7.1 Caso de Uso – Autorización de Memorando	103
Diagrama de Secuencia – Autorización Memorando	104
3.3.8 Búsqueda.....	105
3.3.8.1 Listados	105
3.3.8.2 Búsquedas	106
3.3.8.3 Caso de Uso – Búsqueda de Memorandos por Criterios	106
3.3.9 Permisos.....	107
Capítulo 4: VALIDACIÓN DE LA PROPUESTA	114
4.1 Prueba General	114
4.1.1 Resultado General de las Pruebas	114
4.1.2 Retroalimentaciones	115
4.1.2.1 Redactora de Dirección: Angela Miranda, secretaria de Dirección	115
4.1.2.2 Redactora de Pregrado: Erika Guzmán, secretaria de Pregrado	115
4.1.2.3 Redactor de Dirección: José Miguel González, Encargado de Gestión Financiera	115
4.1.2.4 Emisor de Pregrado: Pedro Godoy, jefe de Carrera Ingeniería Civil en Informática	115
4.1.3 Formularios de Análisis de Experiencia	116
4.2 Futuro del Prototipo	116
Capítulo 5: CONCLUSIONES.....	118
5.1 Conclusiones por Objetivo	118
5.1.1 Conclusión por Objetivo Principal.....	118
5.1.2 Conclusión por Objetivos Específicos	118
5.2 Conclusiones Generales	119
5.3 Conclusiones Personales	120
Capítulo 6: RECOMENDACIONES	121
Capítulo 7: Anexos	122
7.1 Anexo A: Entrevistas	122
Capítulo 8: Bibliografía	126

ÍNDICE DE FIGURAS

Ilustración 1-1 Organigrama del DI - Fuente: Página web del DI.....	17
Ilustración 1-2 Organigrama interno del Comité Ejecutivo - Fuente: Elaboración Propia	17
Ilustración 1-3 Esquema de Memorando - Fuente: Elaboración Propia.....	19
Ilustración 2-1 Diagrama Funcional MVC de Ruby On Rails, Fuente: Elaboración Propia.....	30
Ilustración 2-2 Un Ejemplo de MVC en una aplicación Web de Ruby On Rails, Fuente: Elaboración Propia	31
Ilustración 2-7 Ejemplo de Asociación Polimórfica, Fuente: Rails Association Basics - https://guides.rubyonrails.org/association_basics.html	34
Ilustración 2-11 Gema image_processing en Gemfile	37
Ilustración 2-12 Tabla Active Storage Blobs, Fuente: Elaboración Propia con pgModeler	38
Ilustración 2-13 Tabla Active Storage Attachments, Fuente: Elaboración Propia con pgModeler	39
Ilustración 2-14 Relación entre Active Storage Blobs y Active Storage Attachments, Fuente: Elaboración Propia con pgModeler.....	40
Ilustración 2-15 Relación entre Active Storage y Memorandos, Fuente: Elaboración Propia con pgModeler.....	41
Ilustración 2-17 Campos de ActiveStorage en el Modelo de Memorando	42
Ilustración 2-19 Línea gemfile para CanCanCan	43
Ilustración 2-20 Comando para inicializar CanCanCan	43
Ilustración 2-21 Plantilla de la Clase Ability de CanCanCan.....	43
Ilustración 2-22 Métodos Que Definen Permisos de CanCanCan	44
Ilustración 2-23 Métodos Que Consultan Permisos de CanCanCan	44
Ilustración 2-24 Ejemplos de Permisos Definidos para Modelo de Memorando.....	44
Ilustración 2-25 Etiquetas Predefinidas de CanCanCan.....	45
Ilustración 2-26 Más Ejemplos de etiquetas de CanCanCan	46
Ilustración 2-27 Ejemplos de Consulta de Permisos con CanCanCan.....	47
Ilustración 2-28 Helper especializado Authorize	47
Ilustración 2-29 Helpers Load_and_authorize_resource de CanCanCan	48
Ilustración 2-31 Página de Inicio de la Intranet, Fuente: La Intranet del DI	49

Ilustración 2-35 Página de Inicio de la Intranet con Sección de Memorandos	50
Ilustración 2-36 Zoom a Sección de Memorandos del Menú Lateral de la Intranet	50
Ilustración 3-1 Diagrama de Base de Datos del Prototipo	61
Ilustración 3-2 Diagrama de Casos de Uso de Redactor	67
Ilustración 3-3 Diagrama de Casos de Uso de Redactor de Dirección	69
Ilustración 3-4 Diagrama de Casos de Uso de Emisor.....	71
Ilustración 3-5 Diagrama de Casos de Uso de Emisor Dirección	73
Ilustración 3-6 Diagrama de Casos de Uso de Huésped	74
Ilustración 3-7 Diagrama de Casos de Uso de Contactos de Actor No Admin	76
Ilustración 3-8 Diagrama de Casos de Uso de Contactos de Administrador	77
Tabla 3-11 Caso de Uso: Crear Borrador a partir de Plantilla	96
Tabla 3-12 Tabla Caso de Uso Remisión de Memorando	98
Ilustración 3-10 Formulario de Creación de Roles de la Intranet, Fuente: Elaboración Propia	108
Ilustración 3-11 Roles de Usuarios de Memorandos Para la Intranet, Fuente: Elaboración Propia	109
Ilustración 3-12 Diagrama de Permisos por Perfil de Usuario, Fuente: Elaboración Propia	111
Ilustración 3-13 alias_action de CanCanCan creados para los Permisos del Módulo, Fuente: Elaboración Propia	112

ÍNDICE DE TABLAS

Tabla 1-1 Diagnóstico.....	23
Tabla 1-2 Fortalezas Procedimiento Actual	24
Tabla 3-1 Tabla de Requerimientos	53
Tabla 3-2 Caso de Uso: Crear y Editar Contacto, Fuente: Elaboración Propia.....	78
Tabla 3-3 Caso de Uso: Editar Contacto, Fuente: Elaboración Propia	79
Tabla 3-4 Caso de Uso Ver y Listar Contacto, Fuente: Elaboración Propia	80
Tabla 3-5 Caso de Uso: Eliminar Contacto, Fuente: Elaboración Propia	81
Tabla 3-6 Flujo Principal Caso de Uso: Crear Memorando	82
Tabla 3-7 Caminos Alternativos Caso de Uso: Crear Memorando	85

Tabla 3-8 Caso de Uso: Editar Memorando	90
Tabla 3-9 Caso de Uso: Visualización de Memorando.....	94
Tabla 3-10 Caso de Uso: Crear Plantilla	95
Ilustración 3-9 Caso de Uso: Eliminación de Memorando.....	97
Tabla 3-13 Caso de Uso: Agregar Firma Electrónica Simple	101
Tabla 3-14 Caso de Uso Autorización de Memorando	103
Tabla 3-15 Caso de Uso Búsqueda por Criterios, Fuente: Elaboración Propia	106
Tabla 7-1 Preguntas Entrevistas.....	122

INTRODUCCIÓN

En la actualidad muchas organizaciones se encuentran realizando grandes esfuerzos para digitalizar sus procesos, avanzando en su transformación digital.

El Departamento de informática (DI) de la UTFSM no es la excepción, hace algunos años, al igual que la Universidad completa se encuentra llevando a cabo esta transformación. Esto se ha visto reflejado en constantes desarrollos de software para apoyar a la gestión y el trabajo de sus integrantes.

Entre estos avances, podemos encontrar el desarrollo del portal web de la Intranet en donde se han digitalizado varias funciones administrativas. Con el tiempo, este portal ha tenido nuevas funcionalidades, digitalizando aún más procesos. Frecuentemente se planifican nuevos proyectos para extenderla.

En cuanto a las funciones aún no digitalizadas, podemos encontrar la gestión de los Memorandos del DI, que se usan para formalizar y respaldar comunicaciones oficiales en las que interactúan el director, subdirector, jefe de carrera, profesores, y personal administrativo, en beneficio de una correcta administración y operación del DI.

Al analizar el sistema de gestión actual de Memorandos, al ver sus procesos, se puede apreciar que estos aún dependen del uso físico del papel. Además, al escuchar a los integrantes del DI que están a cargo de la gestión de ellos, se pueden encontrar sugerencias y deseos de mejora. De estas consideraciones se pueden apreciar grandes oportunidades y la justificación de la digitalización de este proceso, considerando vital para el óptimo funcionamiento del Departamento contar con un apropiado sistema centralizado y amigable de gestión de Memorandos, que permita agilizar y aliviar este trabajo, volviéndolo más agradable para aquellos a cargo.

Esta es la razón primordial que motiva esta memoria. En esta se busca implementar un sistema digital centralizado de gestión de Memorandos, *paperless*, en la forma de un módulo de extensión para la Intranet. Este módulo, al igual que la Intranet, se encuentra desarrollado mediante el uso del *Framework Ruby On Rails*.

Este módulo, tiene como principal aporte la centralización de todas las operaciones asociadas en un solo medio amigable y hecho a la medida de los usuarios participantes de esta gestión, tomando en cuenta sus problemas, modernizando el proceso en su beneficio, mientras que simultáneamente, se logra aportar a la transformación digital del DI.

Para abordar estas oportunidades de mejora, se realizó un proceso de investigación y levantamiento de requerimientos, entrevistando a distintos actores y usuarios, quienes orientaron en cada una de las instancias las etapas de diseño y de desarrollo, asignándole una importancia alta a sus ideas y recomendaciones.

En conclusión, el trabajo efectuado en esta memoria otorga un medio tecnológico que moderniza la comunicación interna del Departamento, aportando a su transformación digital y facilitando la labor de sus integrantes, a la vez que busca una migración fácil al nuevo módulo de la Intranet.

La memoria comienza identificando el problema en profundidad, aclarando objetivos y resultados esperados; en la sección 2 se presenta un marco conceptual y tecnologías ocupadas en el módulo; en la sección 3 se presenta la solución en detalle; en la sección 4 se presentan mecanismos de validación de lo propuesto; y finalmente, en la sección 5, se presentan las conclusiones del proceso.

Capítulo 1:DEFINICIÓN DEL PROBLEMA

1.1 Contexto

1.1.1 El Departamento de Informática (DI)

El Departamento de Informática (DI) ha ocupado históricamente para su comunicación interna Memorandos.

Los Memorandos son mensajes oficiales, parecidos a un correo electrónico, firmados y timbrados por su emisor, quien corresponde a un miembro habilitado del DI, cuya firma y autorización valida y respalda su contenido. Los Memorandos se ocupan para comunicar que se solicitó, informó o se realizó alguna acción, respaldan que esta comunicación ocurrió, sirviendo luego de registro histórico; tarea que aporta a la administración, contabilidad y determinación de responsabilidades del Departamento. Corresponden al método de comunicación interno oficial del DI entre sus unidades y con otros departamentos, p.ej. el Departamento de Finanzas, de Recurso Humanos, la Dirección General de Docencia, entre otros.

A continuación, se presenta la organización del DI.

Un componente central del DI es el Comité Ejecutivo, el cual se encuentra compuesto por 6 unidades: Dirección, Pregrado, Postgrado, Investigación, Vinculación con el Medio e Infraestructura, Tecnología y Servicios (I, T & S), constituyendo un equipo total de alrededor de 60 personas, con cada una perteneciendo a una unidad en particular. Algunos colaboradores de este equipo están habilitados para emitir Memorandos, p.ej. director, subdirectores, jefes y, en algunos casos, profesores, entre otros colaboradores, asociando administrativamente con esa unidad los Memorandos que emiten. Todos los miembros, habilitados para emitir o no, utilizan Memorandos en sus comunicaciones oficiales. Anualmente el DI emite unos 500 Memorandos.

A continuación, se presenta el organigrama general del DI.

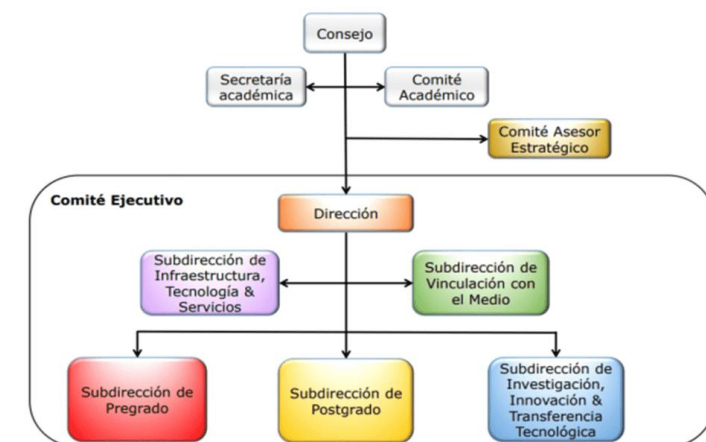


Ilustración 1-1 Organigrama del DI - Fuente: Página web del DI

Luego, detallando el Comité Ejecutivo y sus unidades, tenemos:

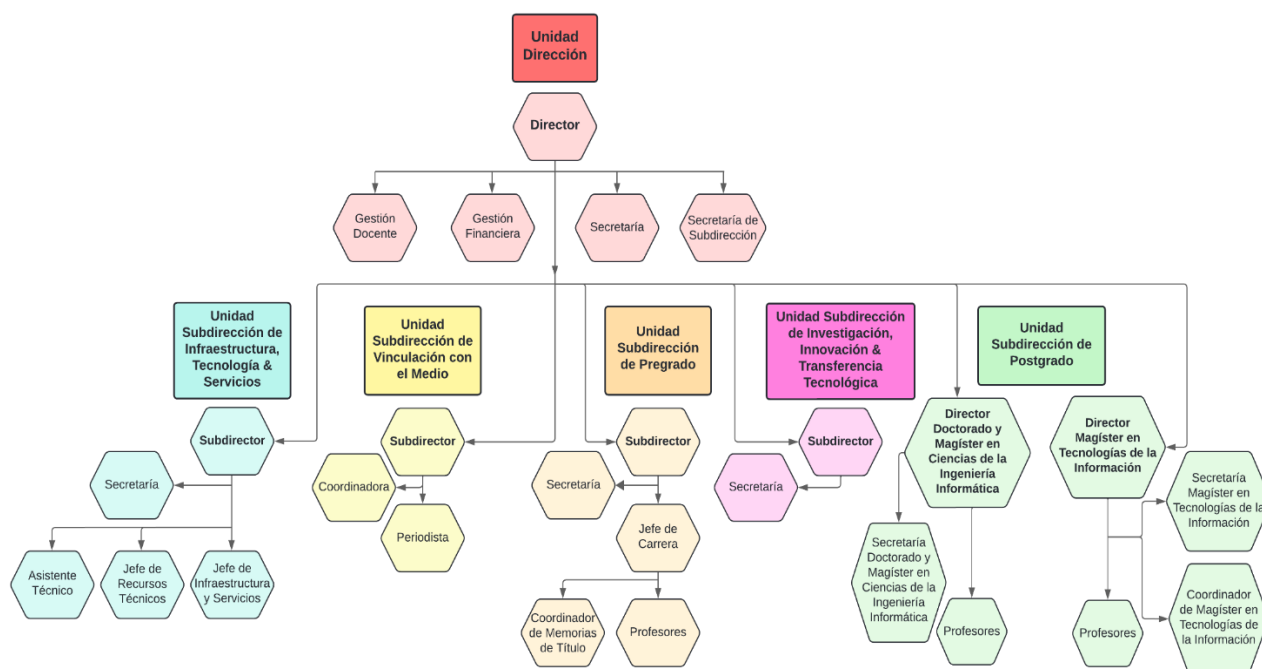


Ilustración 1-2 Organigrama interno del Comité Ejecutivo - Fuente: Elaboración Propia

En su conjunto el DI es encabezado por el Director, y cada unidad se encuentra dirigida por un subdirector. En general, cada unidad, tiene una sola Secretaría en donde se centraliza su administración, siendo la excepción la Dirección y Postgrado, ambos poseyendo más de una Secretaría.

1.1.2 Memorandos en el DI

Si bien el correo electrónico ha ido reemplazando el Memorando como método de comunicación oficial, este documento se sigue utilizando principalmente por las unidades de Dirección y Pregrado al ofrecer un medio para dejar una constancia formal firmada y numerada de una comunicación.

Los casos de uso de los Memorandos varían dependiendo de la unidad. Pregrado y Postgrado, se destacan la emisión de solicitudes de convalidaciones o autorizaciones para atender a un determinado curso, así como también, están las autorizaciones a alumnos a acceder a determinados lugares o permanecer en ellos en horarios no comunes, p.ej. en laboratorios, salas u oficinas. Además de solicitudes de pagos, p.ej. de remuneraciones.

Los Emisores de Pregrado y Postgrado son comúnmente los respectivos subdirectores, jefe de carrera y demás jefes, y los respectivos profesores. Estos últimos son el origen de los temas académicos, como la reserva de salas o laboratorios, entre otras funciones.

Por otro lado, Dirección, Vinculación con el Medio e Infraestructura, Tecnología y Servicios (I, T & S), principalmente emiten solicitudes de pagos y transferencias de diferentes naturalezas, como pago de viáticos, pago de remuneraciones, abono a cuentas, recuperación de gastos, entre otros.

Todos los Memorandos comparten una estructura y datos comunes, independientes de su caso de uso específico. A continuación, se exhibe un esquema de un Memorando vigente a la fecha.



**UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA**

DEPARTAMENTO
DE INFORMÁTICA

4 Casa Central, Valparaíso

5 día, #día de mes de año

6 DI-Unidad-#Memo/AA

A : Autoridad Destino
Cargo del Destinatario
Departamento de Destino

1

DE : Autoridad Emisora
Cargo del Emisor
Departamento de Informática

2

ASUNTO : Un Asunto o Referencia

3

Estimado Destinatario

7

El cuerpo del mensaje. Detalla concisamente el asunto.

Este suele corresponder, pero no limitarse, a solicitudes para la efectucción de pagos, traspaso de fondos, solicitudes de realizar e informar presupuestos, reserva de salas, retiro de equipamiento, autorización de asistencia a cursos, autorización de propuestas de memoria etc.

Esperando tener una buena acogida,

Saluda atentamente,



Autoridad Emisora



CC: destino.con.copia@usm.cl destino.con.copia.2@usm.cl

1

sec:unidad@inf.utsm.cl

9

Dirección: Avda. España 1680, Valparaíso

+56 32 265 4000

4

Ilustración 1-3 Esquema de Memorando - Fuente: Elaboración Propia

1 - Destinatario, corresponde a quién va dirigido el Memorando, puede ser de otra Unidad o de otro Departamento. Se detalla su nombre, cargo y departamento. También se pueden especificar otras personas en la sección CC (Con Copia), que recibirán una copia del Memorando, presencialmente, por medio del sistema de correspondencia de la Universidad, o por correo electrónico, especificando su nombre y correo electrónico.

2 - Emisor, corresponde a un miembro del DI habilitado para emitir, que requiere respaldar oficialmente una comunicación. Se detalla su nombre completo, su cargo y el DI como su Departamento. No puede haber emisores que no sean del DI.

3 - Asunto o Referencia, es el título o resumen del contenido.

4 - Campus, especifica la sede de la UTFSM a la cual pertenece el Memorando y se indica la ciudad de ubicación de esta. Además, al pie de página se especifican los datos de contacto asociados al campus.

5 - Fecha de emisión, es la fecha en la que se emitió el Memorando, implicando cuando se validó para despacharse.

6 - Número Correlativo, identifica únicamente a un Memorando. Su formato es una concatenación de su Departamento de origen, en este caso el DI, su Unidad, un número y su año de emisión en formato corto, p.ej. - *DI- $\{Sigla\}$ Unidad- $\{ \#Memorando \}$ /AA* -. Cada Unidad mantiene su propio correlativo, siendo la numeración mantenida manualmente. Esto representa una oportunidad de mejora al automatizar su generación, liberando al usuario de una gestión de verificación en relación con el último correlativo utilizado, además de potenciales errores.

7 – Cuerpo, mensaje conciso que detalla, explica y vincula el Memorando a la acción que se quiere comunicar, registrar y respaldar.

8 - Firma y Timbre, corresponde a la firma del emisor y timbre de la Universidad o del DI. La firma puede ser o de puño y letra o una firma electrónica simple. No se pudo apreciar que todos los Memorandos cumplieran con el mismo estándar de firma y timbre simultáneos.

9 – Usuario del DI Creador, corresponde al colaborador que creó el Memorando, vinculándolo con la repartición administrativa de la Unidad a la pertenece. Corresponde a la respectiva Secretaría.

La composición de emisor, con su firma, número correlativo y fecha de emisión, le otorgan al Memorando sus principales características de oficialidad y de registro firmado numerado. Además, permiten que sea verificado y recuperado en el futuro. Con esta composición logra respaldar y llevar un registro histórico de las comunicaciones del DI.

1.1.3 Descripción del Proceso Actual de Gestión de Memorandos

Existe un procedimiento para la creación y gestión de Memorandos que funciona de manera similar en todas las Unidades.

Todo Memorando que un miembro habilitado del DI necesite emitir se debe solicitar a la Secretaría de la Unidad respectiva, concentrándose en cada una de estas la gestión de los Memorandos pertenecientes a esta unidad.

Las Secretarías están a cargo principalmente de una sola persona. En caso de ausencia, se cuentan con reemplazantes como plan de contingencia.

El proceso de emisión consiste en los siguientes pasos:

- 1- Por solicitud de un emisor, la Secretaría respectiva crea un Memorando válido con datos críticos suministrados por él, como el destinatario, el cuerpo del mensaje y la firma; y datos agregados por la misma Secretaría, como la fecha de emisión y el número correlativo.
- 2- La Secretaría respectiva redacta el Memorando por medio de Microsoft Word en su computador local de trabajo. En esta etapa se asigna al Memorando un número correlativo, que la Secretaría mantiene manualmente.
- 3- Una vez redactado, lo envía por correo electrónico o por la correspondencia del DI, al Emisor solicitante, para que lo firme. Para el envío físico debe primero imprimirlo, lo que continúa siendo una obligación en ocasiones.
- 4- El Emisor firma el Memorando por medio o de una firma electrónica simple o de puño y letra. Una vez firmado, lo devuelve a la Secretaría respectiva, también enviándolo por correo electrónico o por correspondencia.
- 5- La Secretaría respectiva almacena el Memorando firmado recibido. Si fue firmado físicamente, deben primero escanearlo. Luego, lo archivan en su computador de trabajo.
- 6- Cada Secretaría determina su orden y organización de los Memorandos en sus computadores locales de trabajo. En el caso de Dirección se sincronizan y comparten por la red local entre la Secretaría y Gestión Financiera. Esto provoca que la mantención y búsqueda de estos dependa en un alto grado del encargado de la Secretaría. Una vez almacenado, para consultar, verificar y buscar un Memorando específico, se debe buscar en este computador local particular. Para facilitar esta búsqueda, las Secretarías suelen mantener índices de Memorandos emitidos, con datos relevantes como el número correlativo, fecha de emisión, asunto, emisor, destinatario, entre otros, en planillas de MS-Excel.
- 7- Finalmente, la Secretaría respectiva envía el Memorando firmado al destinatario y a los especificados como *Con Copia*. Lo puede enviar por correo electrónico o físicamente en papel por correspondencia. Las Secretarías son el punto de despacho de los Memorandos de la unidad, como también el punto de recepción de aquellos remitidos hacia la unidad.

1.1.4 Diagnóstico

Del análisis e investigación del contexto actual, se logran detectar las siguientes oportunidades de mejora.

Situación Actual	Oportunidad de Mejora	Solución Propuesta
Creación manual de Memorandos en MS-Word	Falta de formato estándar.	Sistema que aporte con un formato estándar, un estándar

	Falta de estándar de firma. Se debe estar consciente de campos como el número correlativo y la fecha de emisión.	de pie de firma y automatización de campos en la redacción de Memorandos, con un formulario y plantillas.
Asignación y mantención manual de número correlativo de Memorandos	Riesgo de asignación incorrecta. Involucra tiempo para consultarlo manualmente y para compartirlo con otros usuarios.	Sistema que genere automáticamente siempre el correcto número correlativo.
Firma manual de Memorandos	Firma física requiere impresión, envío y posterior escaneo del documento firmado.	Autorización y vinculación con el emisor digital de un Memorando por medio de un sistema digital de fácil acceso remoto.
Envío principalmente físico de Memorandos	Proceso puede tomar varios minutos considerando los envíos a firmarse y luego de vuelta a almacenarse. Dependencia en personal de correspondencia.	Sistema de envío digital automático de Memorandos y notificaciones por correo electrónico.
Almacenamiento local en computadores de Secretarías - Acceso y Almacenamiento	Acceso de Memorandos almacenados solo disponible localmente. Dependencia en alto grado en una persona de Secretaría. RespalDOS de la información no formales. Mayor vulnerabilidad de pérdida de datos e interrupción de operación frente a fallos de los computadores locales.	Almacenamiento centralizado y accesible desde cualquier lugar si se cuenta con conexión a internet. Sistema informatizado siempre disponible. Almacenamiento en servidores del DI, con mantenimiento dedicado y respaldos.

Almacenamiento local en computadores de Secretaría - Búsqueda y Recuperación	Búsqueda de Memorandos manual en el sistema de archivos. Organización de Memorandos se debe mantener manualmente por la Secretaría, considerando la creación de carpetas, el archivado de los documentos, el nombrado correcto de los archivos, etc. Secretarías mantienen índices de los Memorandos emitidos por medio de planillas de Excel	Provisión de un listado de Memorandos que permita una fácil y rápida búsqueda, y recuperación apoyada por criterios. Sistema que libere a la Secretaría del trabajo de mantener un orden de archivos o índices.
Falta de un repositorio centralizado	Unidades e información desconectadas. Se hace más difícil analizar la información y generar reportes de algún tipo.	Repositorio centralizado para un fácil acceso remoto, mayor integración y comunicación entre Unidades, e información disponible para realizar análisis.

Tabla 1-1 Diagnóstico

1.1.5 Fortalezas del Sistema Actual

Cabe destacar que el sistema actual posee fortalezas que hay que considerar a la hora de diseñar la solución, impidiendo o mitigando su potencial ausencia. La nueva solución debe contemplar el heredar estas características.

Fortaleza	Impacto
Histórico	A prueba de contingencias y altamente familiar para los miembros de DI.
Efectivo	Cumple satisfactoriamente su función de gestión de Memorandos.
Resistente	No depende de internet ni en alto grado en energía, ya que siempre se puede volver a la redacción con firma manual y al de envío por correspondencia.

Resiliente	Al ocupar esencialmente papel, es menos vulnerable a un ataque informático.
------------	---

Tabla 1-2 Fortalezas Procedimiento Actual

De estas fortalezas se concluye que

- La solución debe enfocarse en asegurar una migración fácil y cómoda para el usuario
- Debe proveer ayudas para su uso
- Debe considerar un plan de contingencia en caso de perderse la conexión a internet.

1.1.6 Conclusión

Con lo anterior, se puede visualizar que el proceso actual de gestión de Memorandos presenta grandes oportunidades de mejora al diseñar un medio digital, centralizado y especializado.

1.2 Objetivos

1.3.1 - Objetivo Principal

Desarrollar un módulo para la Intranet del DI de la UTFSM, que se encargue de la creación, almacenamiento, envío, registro y búsqueda de Memorandos, para lograr facilitar, amenizar y acelerar el trabajo de comunicación oficial del departamento y alivianar la carga a aquellos actores que realizan este trabajo. Para de esta manera, aportar a la transformación digital del departamento y a su administración.

1.3.2 - Objetivos Específicos

1. Diseñar una estructura y formato para estandarizar la creación de los Memorandos.
2. Ofrecer una experiencia automatizada, rápida y simple de edición de Memorandos. Esto comprende, principalmente, proveer de las funciones básicas y más usadas de edición de texto, además, de ofrecer el autocompletado de campos recurrentes.
3. Diseñar un sistema en el cual personas con cuenta en la Intranet puedan crear Memorandos, pero solo el Emisor correspondiente pueda firmarlos.
4. Diseñar un sistema de autorización de Memorandos, es decir, que el Emisor pueda autorizar y dar validez a un Memorando virtualmente, sin tener que imprimir el documento.
5. Diseñar un sistema de envíos automáticos de Memorandos:
 - a. Envío de notificación al Emisor, de la presencia de un Memorando en espera de su firma.
 - b. Una vez autorizado, envío del documento a los destinatarios especificados

6. Diseñar listado y búsqueda de Memorandos por medio de diferentes filtros, por ejemplo:

a. Listados

- i. Listar los Memorandos creados por el usuario actual
- ii. Listar Memorandos a la espera de ser autorizados
- iii. Listar Memorandos pendientes por autorizar, en el caso de que el usuario actual sea el Emisor de aquellos Memorandos
- iv. Listar los Memorandos emitidos y autorizados por la unidad a la que pertenece el usuario

7. Búsquedas

- a. Buscar por Emisor
- b. Buscar por Destinatario
- c. Buscar por Rango de Fechas
- d. Buscar por Unidad
- e. Buscar por Número Correlativo
- f. Buscar por Asunto

1.3 Alcances de la Solución

El alcance de la solución corresponde al proceso de desarrollo de un proyecto propuesto para la Intranet.

El cual consiste en desarrollar un sistema de gestión de Memorandos, que digitalice el flujo estándar de su ciclo de vida, comprendiendo: emisión, almacenamiento, búsqueda, envío y recepción de los Memorandos del DI.

Enfocándose en las necesidades de sus actores más interesados, las unidades de Dirección y Pregrado. Materializándose en la implementación de un módulo funcional parte del portal de la Intranet.

No involucra un plan de pruebas general de la solución, ni tampoco su paso a producción.

1.4 Resultados Esperados

Se espera implementar con éxito un módulo a la intranet que se encargue exitosamente de la gestión de los Memorandos del DI y que se adopte fácil y rápidamente al departamento.

Capítulo 2: ESTADO DEL ARTE

2.1 Marco Conceptual

2.1.1 Filosofía Human Centered Design y Framework Design Thinking

Human-Centered Design (HCD) es una filosofía para desarrollo de software y diseño de productos, que se enfoca en priorizar la experiencia y las necesidades del usuario, centrándose el proceso de diseño en las personas, asegurando que las soluciones desarrolladas realmente aborden las necesidades y desafíos que expresan y experimentan. HCD se enfoca en comprender profundamente a los usuarios y los contextos en donde se desenvuelven.

Se compone de 3 grandes ideas:

El proceso comienza con la Empatía, lo que implica observar y escuchar a los usuarios para descubrir sus necesidades reales. Luego, se definen los problemas a resolver desde la perspectiva del usuario, generándose diferentes ideas creativas para solucionarlos. Finalmente, se prototipan y prueban las soluciones concebidas con los usuarios para obtener la retroalimentación que permita corregir, guiar y avanzar en la confección de la solución.

Design Thinking, corresponde a una instancia práctica o *framework* de Human-Centered Design. Permite reducir la incertidumbre sobre los más relevantes problemas raíces en algún contexto, para luego planificar y diseñar una solución. Se caracteriza por basarse en desarrollos ágiles y por declarar a los usuarios y personal como el factor más importante del desarrollo, involucrando e interactuando constantemente con todos los actores interesados y afectados por un producto.

Las 5 etapas que componen Design Thinking son:

Empatizar: Este paso inicial implica interactuar con los usuarios para entender sus necesidades y contextos. Se busca comprender sus emociones y motivaciones mediante observación e interacción directa. Esta fase establece una base sólida para el diseño de soluciones centradas en el usuario.

En el desarrollo de esta memoria se inspiró en esta fase ocupando la herramienta de entrevistas y conversaciones grupales para entender y captar las experiencias de los miembros del DI, principalmente de las Secretarías.

Definir: Con base en la información recopilada, se redefinen los problemas desde la perspectiva del usuario, identificando oportunidades de innovación.

En el contexto de la memoria, esto incluyó la detección de las oportunidades de mejora, convergiendo en la conclusión de la falta de completa y especializada digitalización en la gestión de memorandos, como una oportunidad de mejora raíz.

Idear: Se fomentan sesiones de generación de ideas, como *brainstorming*, para desarrollar múltiples soluciones potenciales.

En esta etapa se exploró la confección de un software gestión de Memorandos especializado y con las características centrales que se muestran en esta memoria. También se determinó la creación de un módulo como el tipo de solución a seguir.

Prototipar: Se crean prototipos de baja fidelidad de las ideas seleccionadas, lo que permite materializar conceptos de manera rápida y económica.

En el caso de esta memoria, se centró en confeccionar prototipos de alta fidelidad, tomando más tiempo en prototipar, pero teniendo una representación más confiable de la experiencia.

Testear: Los prototipos se prueban con usuarios reales, obteniendo retroalimentación directa que informa nuevas iteraciones del diseño.

En el caso de esta memoria, se mostraron los prototipos de alta fidelidad a funcionarios de Dirección y Pregrado, obteniendo retroalimentación de sus reacciones y observaciones.

Estas etapas no son secuenciales necesariamente, si no que concurrentes, además de ser parte de un proceso iterativo.

Una vez definidos los problemas raíces, se redactan los requerimientos. Estos permiten partir un proceso de exploración de ideas para concebir una solución, proceso que termina con un prototipo, el cual se pone a prueba con los usuarios. Se repetirá esta secuencia frecuentemente, pero no necesariamente desde el comienzo ni completa, para ajustar las definiciones y prototipos luego de la fase de prueba, tomando en cuenta la retroalimentación obtenida.

Esta memoria se inspiró en seguir una metodología basada en los pasos y herramientas de *Design Thinking*. Se buscó mantener el contacto con los potenciales usuarios del sistema y obtener su retroalimentación. Se mantuvo estrecha relación con las Unidades del DI más interesadas, siendo estas, Dirección y Pregrado, mediante sus Secretarías y otros actores, como Gestión y Finanzas, y se centró el trabajo de esta memoria en desarrollar una solución para ellos y a su medida.

2.1.2 Patrón de Diseño Model-View-Controller (MVC)

Model-View-Controller (MVC) es un patrón de diseño o de arquitectura de software muy popular en el desarrollo de aplicaciones web y de escritorio. Fue creado originalmente por Trygve Reenskaug en 1979 mientras trabajaba en el laboratorio de investigación de Xerox PARC. La intención original de MVC era mejorar la organización del código en aplicaciones con interfaces gráficas de usuario, permitiendo una separación clara entre la representación de la información y la interacción del usuario (National Library of Norway - Archive-It, 2018). Los desarrolladores encontraron que la separación de responsabilidades conducía a un menor acoplamiento, lo que a su vez hacía que el código fuera más fácil de escribir y mantener. Cada concepto o acción se expresaba en un solo lugar conocido. (Simon & Dave, 2023)

MVC separa una aplicación en tres componentes interconectados: el Modelo, la Vista y el Controlador. Este enfoque permite que cada componente se desarrolle de manera independiente, mejorando así la modularidad y escalabilidad del software (Wilf & John, 1990):

- **Modelo:** Representa los datos y la lógica de negocio de la aplicación. Es responsable de acceder a la capa de almacenamiento de datos, por ejemplo, una base de datos, definir las reglas de negocio y realizar validación de datos al llevarse a cabo alguna operación en los datos, y notificar a las Vistas de los cambios experimentados.
- **Vista:** Se encarga de la representación visual de los datos, es decir, todo lo que el usuario puede ver en pantalla. Las Vistas son responsables de preparar la respuesta a enviar al usuario, presentando los datos del modelo de una manera entendible, además de interactuar con él.
- **Controlador:** Actúa como un intermediario entre el Modelo y la Vista. Recibe las solicitudes y las entradas del usuario desde la Vista, las procesa, interactúa con los Modelos con los que deba, posiblemente modificándolos, y determina la respuesta adecuada o Vista a renderizar, pasándole los datos necesarios.

La filosofía detrás de MVC es la "separación de preocupaciones" (Separation of Concerns o SoC), lo que significa que cada parte del sistema tiene una responsabilidad clara y definida, permitiendo gestionar aplicaciones complejas al organizar el código en partes lógicas distintivas. MVC permite que los desarrolladores cambien o actualicen las interfaces de usuario sin tener que reescribir la lógica de negocio, lo que hace que el mantenimiento y la expansión de la aplicación sean más manejables (Sam & Dave, Capítulo - The Architecture of Rails Applications - Agile Web Development With Rails 7, 2023).

La Intranet fue desarrollada con *framework* Ruby on Rails. El módulo de Gestión de Memorandos, por ende, sigue el patrón, estando compuesto de Vistas, con las cuales interactúan los usuarios; Controladores, que reciben y procesan las entradas y solicitudes al servidor, preparando una respuesta e invocando una Vista; y Modelos, que representan a los recursos y entidades con la lógica del negocio involucrada y que se conectan a la base de datos, validando los datos al realizar alguna operación.

2.1.3 Ruby on Rails (RoR)

Ruby On Rails (RoR), comúnmente referido solamente como Rails, es un framework de código abierto escrito para el desarrollo de Aplicaciones Web en el lenguaje de programación Ruby. Está diseñado para facilitar la programación de aplicaciones siguiendo el patrón *Model-View-Controller* (MVC), que separa la aplicación en tres capas: Modelo, Vista y Controlador. También es compatible con el desarrollo de aplicaciones RESTful, lo que facilita la creación de Interfaces de Programación de Aplicaciones (API) (Rails-37signals, 2024).

Fue creado por David Heinemeier Hansson y publicado en su primera versión, Rails 1.0, en diciembre de 2005. Al momento del desarrollo de esta memoria, en agosto 2024, se encuentra en su versión 7.1.3.4. (Rails Core, 2024)

Rails no solo implementa el patrón MVC, sino que impone serias restricciones en cómo se puede estructurar una aplicación web, para asegurar el cumplimiento del patrón. Pero, justamente son estas restricciones las que permiten todas las facilidades y automatizaciones que agilizan el desarrollo.

Dentro de sus características y principios rectores, nos encontramos con:

- **Don't Repeat Yourself (DRY):** Este principio enfatiza la importancia de evitar la duplicidad en el código, lo que facilita su mantenimiento y reduce los errores. Modificar el código no es tan costoso como en otros frameworks, ya que serán pocas las clases que se verán afectadas. Dices lo que necesitas decir en un solo lugar, un lugar que a menudo sugieren las convenciones de la arquitectura MVC, y luego sigues adelante. En otros frameworks, en especial en los tiempos de origen de Rails, hacer un simple cambio en el esquema de la base de datos podía implicar una docena de cambios en el código en diferentes lugares. Rails en esto fue una revelación en esos tiempos.
- **Convention over Configuration:** Rails asume las configuraciones por defecto y tiene valores predeterminados sensatos para casi todos los aspectos de la integración de una aplicación. Si sigues las convenciones, puedes escribir una aplicación utilizando menos código. En caso de necesitar anular las convenciones, Rails también lo hace fácil.

Gran parte de la razón de la alta popularidad que alcanzó Rails se debió a los puntos anteriores, ya que no solo implementaba y permitía desarrollar aplicaciones escalables con MVC, sino que también, al asumir convenciones, permitía encargarse de gran parte de la configuración que enlaza los componentes de MVC, liberando al desarrollador de este trabajo, incrementando su productividad y eficiencia, a la vez que disminuye su esfuerzo. Luego, el principio DRY, permitía modificaciones más rápidas y limpias, ya que solo se debe cambiar el código en un solo lugar, volviendo el código más mantenible. De esta manera se puede obtener rápida y cómodamente una aplicación funcionando y con unidades de testing.

Dentro de sus fortalezas, podemos mencionar:

- **Agilidad en el Desarrollo:** Si se siguen sus convenciones por defecto, Rails permite un desarrollo rápido de aplicaciones gracias a su estructura y sus generadores de código
- **Rico Ecosistema:** Cuenta con una gran cantidad de gemas (librerías) que extienden su funcionalidad.
- **Pruebas Integradas:** Facilita la implementación de pruebas automatizadas para el software.
- **Comunidad Activa:** Ha mantenido buenos niveles de popularidad y tiene una comunidad de desarrolladores muy activa y colaborativa. En el índice TIOBE de agosto de 2024, se encuentra en la 15ª posición, subiendo de la 17ª del mes pasado. (Company, 2024)

Aunque también posee debilidades:

- **Rendimiento:** Puede ser menos eficiente en términos de rendimiento en comparación con otros *frameworks* más ligeros.
- **Curva de Aprendizaje:** Para los principiantes, puede ser abrumador debido a su amplio ecosistema, generadores y automatización, y, en especial, convenciones, ya que se deben

conocer de antemano para poder entender lo que está ocurriendo. Rails suele ser minimalista en el código y en los comentarios respectivos. Pero una vez que se entienden las bases, el desarrollo se agiliza, se requiere menos esfuerzo y líneas para realizar tareas y el código es más limpio y legible.

- **Nuevas Tecnologías:** Frente a nuevas tecnologías y arquitecturas, como al querer ocupar Microservicios, Rails, por la gran dependencia de su diseño en MVC, y sus restricciones y convenciones que lo imponen, no es tan conveniente ni eficiente a la hora de desarrollar aplicaciones que requieran o se beneficien de otra arquitectura, p.ej. la mencionada arquitectura de Microservicios.

A continuación, se muestra un diagrama de cómo funciona el patrón MVC en Ruby On Rails

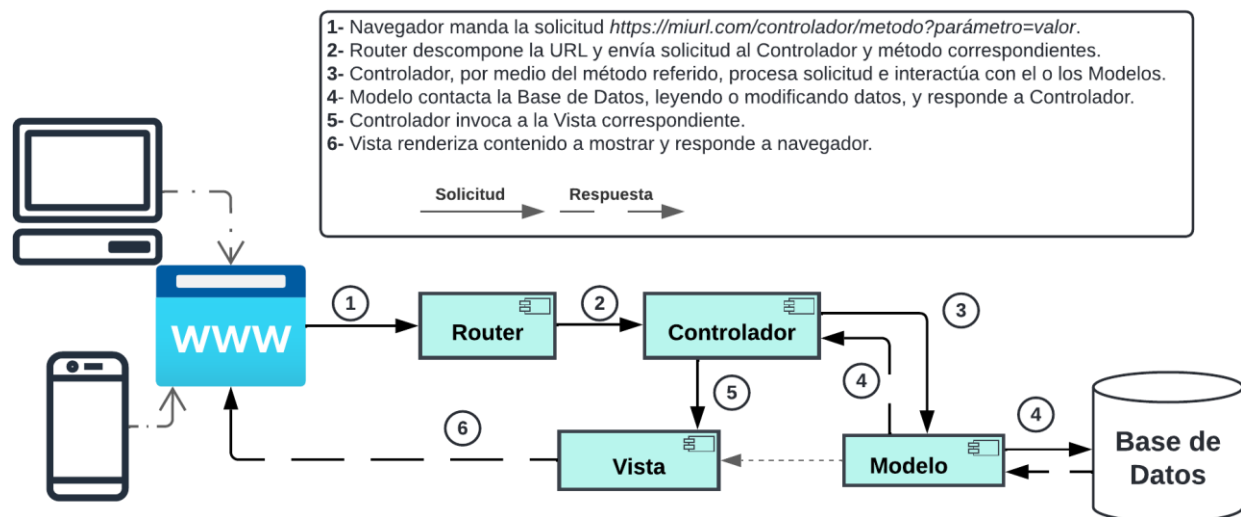


Ilustración 2-1 Diagrama Funcional MVC de Ruby On Rails, Fuente: Elaboración Propia

En una aplicación de Rails, una solicitud entrante se envía primero a un Router, que determina a dónde en la aplicación debe enviarse la solicitud y cómo debe procesarse. Finalmente, esta fase identifica un método particular (llamado acción en la jerga de Rails) del Controlador. La acción puede examinar los datos en la solicitud, interactuar con el o los Modelos o provocar que se invoquen otras acciones. Eventualmente, la acción prepara información para la Vista, que renderiza la respuesta al usuario.

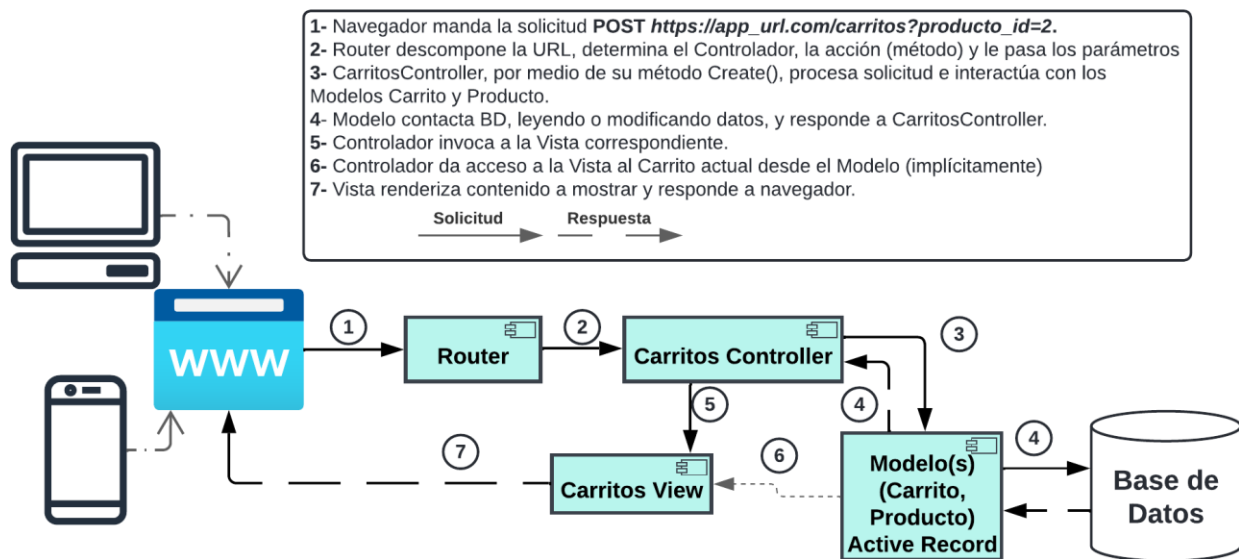


Ilustración 2-2 Un Ejemplo de MVC en una aplicación Web de Ruby On Rails, Fuente: Elaboración Propia

Un ejemplo práctico. Una aplicación ha mostrado previamente su página de catálogo de productos, y el usuario acaba de hacer clic en el botón Agregar al Carrito junto a uno de los productos. El botón se conecta a `https://app_url.com/carritos?producto_id=2`, donde `carritos` es un recurso en la aplicación asociado a un Modelo, Carrito, y 2 es el ID del producto seleccionado, siendo Producto otro Modelo.

El Router recibe la solicitud entrante y la descompone. La solicitud contiene una ruta, `/carritos?product_id=2`, y un método REST, siendo estos POST, GET, PUT, PATCH o DELETE, en este ejemplo corresponde a un POST. Rails toma la primera parte de la ruta, `/carritos`, como el nombre del Controlador y el `producto_id` como el ID de un producto. Por convención, los métodos POST están asociados con acciones `Create()`. En consecuencia, el Router sabe que tiene que invocar el método `Create()` en el Controlador *Carritos Controller*. Si fuera otro método, este se vería reflejado en la ruta (p.ej. `https://app_url.com/carritos/otra_accion?parametro=valor`)

El método `Create()` maneja la solicitud. En este caso, encuentra el carrito de compras del usuario actual, que es un objeto gestionado por el Modelo Carrito. También le pide al Modelo que encuentre la información del producto 2, buscando en el Modelo Producto. Luego le dice al carrito de compras que agregue ese producto a sí mismo. El Controlador le dice qué hacer, y el Modelo sabe cómo hacerlo.

Ahora que el carrito incluye el nuevo producto, podemos mostrárselo al usuario. El Controlador invoca el código de la Vista, pero antes de hacerlo, organiza las cosas para que la Vista tenga acceso al carrito desde el Modelo. En Rails, esta invocación suele ser implícita; las convenciones ayudan a vincular una Vista particular con una acción dada (Sam & Dave, Agile Web Development With Rails 7, 2023).

La implementación de MVC en Rails se manifiesta en las siguientes clases:

- **Active Record:** Es la clase que implementa el Modelo en un proyecto Rails. También corresponde a un ORM (Object-Relational Mapping) entidad que se encarga de conectarse con la Base de Datos. También realiza la conversión de filas de tablas de base de datos a Objetos Ruby en memoria, transformando una fila en un objeto cuyos nombres y valores de sus atributos son las columnas de la fila leída. Y viceversa, un objeto en memoria se almacena o modifica una fila en base de datos, con sus columnas, correspondiendo al nombre de los atributos del objeto y tomando el valor contenido en ellos. (Active Record Guide, 2024)
- **Action View:** Es la clase que implementa la Vista. Ocupa plantillas HTML apoyadas con ERB (Embedded Ruby) para configurar las respuestas al cliente. Puede por medio de *Action Helpers* se pueden encapsular comportamientos de presentación adicionales, como formateos, condicionales o comportamiento dinámico, también permiten acceder a valores del Modelo, aunque en general, el Controlador le pasa los parámetros del Modelo necesarios. Está asociada a un Controlador (Action View Ruby Guide, 2024)
- **Action Controller:** Es la clase que implementa el Controlador. Está asociado a un grupo de vistas. Procesa las entradas y solicitudes del usuario, dependiendo del método invocado por la ruta. Al finalizar, llama la vista correspondiente, enviándole los datos obtenidos de Modelos o resultados de su procesamiento (Action Controller Ruby Guide, 2024)

La Intranet está desarrollada utilizando Ruby On Rails versión 5.8.2.1

2.1.4 Asociaciones Principales de Active Record en Ruby On Rails

En Ruby On Rails, a los Modelos Active Record se les pueden declarar asociaciones con otros Modelos. Estas permiten representar en la aplicación relaciones declaradas en la Base de Datos. P.ej. una tabla que posee una llave foránea con aquella que posee la llave primaria. Se establece una abstracción de estas relaciones en la aplicación. Al establecer una, Rails nos permite, por medio de métodos, acceder desde un objeto a sus relacionados, poder crear o modificar desde un objeto a sus relacionados, poder presentar un objeto y también sus objetos relacionados, acceder desde un objeto a su objeto padre o a sus hijos, etc. Además, no solo ayuda a la abstracción y a agilizar el desarrollo, sino que también a nivel interno y de memoria, Rails optimiza estas relaciones, logrando que interactúen de manera eficiente, simplificando las consultas de datos relacionados y manteniendo la integridad de las conexiones. Rails admite varios tipos de asociaciones:

1. **belongs_to:** Indica que un Modelo que está conectado a otro en una relación que desde su lado es uno y solo uno. Tiende a ser complementada desde el otro lado con un `has_one` o `has_many`.
2. **has_one:** Indica una relación desde su lado de cero a uno.
3. **has_many:** Indica una relación desde su lado de cero o muchos

4. **has_many : through:** Establece una conexión de muchos a muchos a través de un Modelo de unión intermedio.
5. **has_and_belongs_to_many:** Configura una conexión de muchos a muchos directa, pero requiere de la creación de una tabla de unión en la base de datos.
6. **Polymorphic:** Permite que una Instancia de un Modelo pueda pertenecer - *belongs_to* – a más de un Modelo diferente por medio de una sola asociación.

Las asociaciones facilitan la recuperación y manipulación de datos mediante la gestión automática de uniones SQL y claves externas, proporcionando métodos para agregar, actualizar y eliminar objetos relacionados de un Modelo. Rails provee su [guía online](https://guides.rubyonrails.org/association_basics.html) - https://guides.rubyonrails.org/association_basics.html con más detalles.

Explicaremos en más detalle la asociación polimórfica

2.1.4.1 Polimorfismo

El polimorfismo corresponde a una asociación especial otorgada por Active Record, en la cual se permite crear un Modelo que puede pertenecer, *belongs_to*, a más de un tipo de Modelo.

Normalmente, cada asociación *belongs_to* crea un campo en el modelo, una llave foránea, para referenciar a aquel *record* al cual pertenece. Si se quiere que el mismo Modelo pueda pertenecer a otro Modelo adicional, se debe declarar otra asociación *belongs_to*, la cual creará otra llave foránea, referenciando al Modelo adicional.

Para proyectos muy grandes y para un Modelo que pueda pertenecer a muchos Modelos, en especial al avanzar el desarrollo de una aplicación, esto representan un problema. Se generan muchos campos, de los cuales potencialmente solo algunos estarán activos para algún registro en particular, con muchos estando vacíos (con valor NULL) mientras no referencien al Modelo correspondiente, ocupando mucho espacio y volviendo más difícil de leer y mantener.

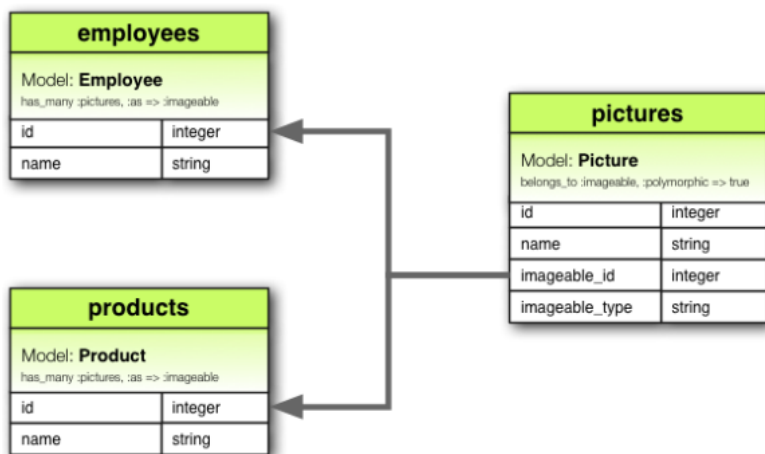
Las asociaciones de polimorfismo que ofrece Rails, vienen a resolver este problema al otorgar una manera de que un Modelo pueda *belongs_to* a diferentes modelos, dinámicamente, sin conocerlos de antemano, y sin crecer desmedidamente. Haciendo el código mucho más limpio, legible y mantenible.

Para lograr esto, debe declarar y hacer uso de 2 campos. Un campo *objeto_id* y otro *objeto_type*. Ambos conforman la llave foránea para la asociación polimórfica.

- **Objeto_id:** corresponde al campo que identifica a un registro en otra tabla, es decir, una llave foránea
- **Objeto_type:** Nombre de la Clase del Modelo de Rails al que corresponde al *objeto_id*

El *Objeto_type* identifica el Modelo en cuestión, y su tabla asociada en la BD, y el *Objeto_id* lo identifica dentro de ese Modelo, y en su tabla en la BD.

De esta manera se puede crear un Modelo que, en vez de pertenecerle a un solo Modelo en particular, puede pertenecerle su objeto_type y luego especificando el ID de ese objeto es su tabla respectiva, objeto_id.



```

class Picture < ApplicationRecord
  belongs_to :imageable, :polymorphic => true
end

class Employee < ApplicationRecord
  has_many :pictures, :as => :imageable
end

class Product < ApplicationRecord
  has_many :pictures, :as => :imageable
end
  
```

Ilustración 2-3 Ejemplo de Asociación Polimórfica, Fuente: [Rails Association Basics - https://guides.rubyonrails.org/association_basics.html](https://guides.rubyonrails.org/association_basics.html)

En el ejemplo anterior, se define la relación polimórfica *imageable*, se declara que una imagen, *picture*, que pertenecer, es decir, mantener una referencia en su estructura, con uno y solo un modelo polimórfico abstracto *imageable*. Luego, en los demás modelos que queramos relacionar con *picture*, podemos declararlos como que son un *imageable*, que *has_one* o *has_many pictures*, actuando como un *imageable*. Luego, en la tabla asociada al modelo *Picture*, *pictures*, tendremos los campos *imageable_id* y *imageable_type*. Para registrar que una *picture* particular pertenece a un *Product*, los campos polimórficos tendrán valores:

- *imageable_id*: #product_id, p.ej. 15
- *imageable_type*: 'Product'

En el caso de que una *picture* pertenezca a un *Employee*, tendremos:

- `imageable_id`: `#employee_id`, p.ej. 30
- `imageable_type`: `Employee'`

Luego, siempre se pueden agregar o quitar modelos dueños de *picture* al agregar o retirar la relación de los modelos dueños, pudiéndose extender esta relación *belongs_to* dinámicamente y sin cambiar el modelo o llenarlo de campos.

2.2 Tecnologías y Librerías Parte del Módulo

2.2.1 Summernote

Summernote es un editor de texto enriquecido WYSIWYG (What You See Is What You Get) basado en la web que permite a los usuarios crear contenido HTML fácilmente. Se pueda utilizar en aplicaciones web para proporcionar una manera sencilla y amigable de escribir, formatear y estructurar texto.

Entre sus características principales, se incluyen:

- **Interfaz visual intuitiva:** Permite a los usuarios formatear texto (negrita, cursiva, subrayado, listas, etc.) y agregar elementos como imágenes, enlaces, tablas y videos a través de botones.
- **Soporte para HTML5:** El contenido generado es compatible con HTML5, lo que facilita su integración en sitios web modernos.
- **Personalización:** Summernote es altamente personalizable, permitiendo a los desarrolladores modificar su apariencia, agregar o quitar funciones, y adaptarlo a las necesidades específicas de su aplicación.
- **Integración fácil con Frameworks Frontend:** Puede integrarse fácilmente con frameworks como Bootstrap y jQuery, lo que le da flexibilidad para ser utilizado en diferentes entornos de desarrollo.
- **Soporte de idiomas:** Summernote permite configurar diferentes idiomas para la interfaz del editor.

Dada su simpleza, facilidad instalación en una aplicación, flexibilidad y fácil integración con frameworks de frontend y su característica de poder incluir tablas en el contenido, es que se eligió usar para probar el prototipo por sobre la versión ligera recomendaba por Rails, Trix.

2.2.2 WKHTMLTOPDF (WK HTML To PDF)

WKHTMLTOPDF es una herramienta de código abierto (bajo la licencia LGPLv3) que permite convertir documentos HTML en archivos PDF y varios formatos de imagen utilizando el motor de renderizado Qt WebKit. Funciona completamente en modo “headless”, lo que significa que no requiere una pantalla, navegador o servicio de visualización para operar.

La herramienta se puede utilizar a través de la línea de comandos y también ofrece una biblioteca en C si se requiere integrar su funcionalidad en aplicaciones. Es especialmente útil para generar documentos

PDF dinámicamente desde contenido web, como facturas, informes, o, en nuestro caso, Memorandos, y se puede personalizar con una variedad de opciones.

Se encuentra disponible en todos los principales sistemas operativos, como Windows, MAC OS y diferentes distribuciones de Linux, como las basadas en Debian, por ejemplo, Ubuntu y Red-Hat, por ejemplo, Fedora, CentOS, etc.

En el caso de su mencionada librería escrita en C, esta se puede enlazar con aplicaciones escritas en otros lenguajes por medio de librerías *Wrapper* que ofrecen algunos ecosistemas de librerías de estos lenguajes, librerías que hacen de intermediarias entre *wkhtmltopdf* y la nueva aplicación.

Se debe tener en cuenta que el HTML a entregar como entrada a *wkhtmltopdf*, debe ser estar sanitizado y ser confiable, de lo contrario, se podría comprometer la seguridad del servidor que lo ejecuta. (Kulkarni-Truelsen, 2024)

2.2.3 Wicked_pdf

Wicked_pdf corresponde a una gema (librería) para Ruby on Rails que facilita la generación de archivos PDF a partir de HTML. Utiliza la herramienta *wkhtmltopdf* para convertir vistas HTML en documentos PDF, permitiendo a los desarrolladores escribir vistas como lo harían normalmente y dejar que Wicked_pdf se encargue de la conversión. Corresponde a una gema *wrapper* que actúa de intermediaria entre *wkhtmltopdf* y una aplicación Rails. Permitiendo la adición de la funcionalidad ofrecida por *wkhtmltopdf* a nuestra aplicación (Sterrett, 2024)

Wicked_pdf ofrece una interfaz amigable de usuario para poder integrar esta funcionalidad. Posee abstracciones que juntan varios comandos de *wkhtmltopdf*, para englobar procesos frecuentes. Ofrece una gran facilidad de uso, ya que permite utilizar vistas HTML existentes para generar PDFs, además de muchas opciones para personalizar el PDF resultante. También tiene una comunidad de usuarios activa que contribuye a su mantenimiento y mejora, lo que agrega valor a la mantención futura de la aplicación.

Esta gema es particularmente útil para crear reportes, facturas, y, en nuestro caso, Memorandos, pero también cualquier otro documento que requiera de un formato fijo y presentable.

Requiere que *wkhtmltopdf* esté instalada en el servidor que albergue la aplicación web. En este caso, el servidor que alberga – *hostea* - la Intranet.

2.2.4 Active Storage

Active Storage corresponde a una gema (librería) de Ruby On Rails que se encarga de la gestión de archivos adjuntos, incluyendo funciones para la carga y adjunción en formularios web; el almacenamiento en un medio persistente, como un disco duro local o en un servicio en la nube; y su manipulación y presentación en alguna Vista posterior. Pero, principalmente, permite asociar estos

archivos adjuntos a un Modelo. Crea la abstracción de que los archivos adjuntos le pertenecen a un objeto, pudiendo a partir de aquellos objetos referenciar y manipular estos archivos si se tratara de cualquier otro campo, p.ej. el poder presentar el archivo adjunto en una Vista asociada al objeto, entre otras acciones. También, Active Storage provee métodos para realizar las referenciaciones y manipulaciones.

En el caso del almacenamiento persistente, Active Storage facilita el almacenamiento de los archivos en la nube, en los servicios de Amazon S3, Google Cloud Storage o Microsoft Azure Storage. También se puede configurar el almacenamiento en disco local, ideal para el desarrollo y para realizar pruebas. Además, admite la duplicación de archivos a servicios subordinados para copias de seguridad y migraciones. Una vez más, estos archivos están vinculados a un Modelo, no importa donde se almacenen los archivos, Active Storage se encargará de vincularlos a los objetos correctos, que pertenezcan a un Modelo que declare el uso de Active Storage.

También, usando Active Storage, una aplicación puede aplicar transformaciones a archivos adjuntos que correspondan a imágenes. Por ejemplo: cambiar las dimensiones de una imagen para adaptarse a miniaturas o vistas previas; cambiar el formato de la imagen, por ejemplo, de JPEG a PNG; recorte (cropping); rotación; añadir efectos visuales a las imágenes, como cambios en el brillo o contraste; entre otras funciones.

Eso sí, Active Storage no realiza estas operaciones por sí sola, requiere de una librería de manipulación de imágenes. Active Storage pone a disposición los métodos para aplicar estas transformaciones a los archivos adjuntos que necesitemos, pero no hace el procesamiento de imágenes él, sino que invoca a esta dependencia.

Es más, Active Storage depende de los Sgtes. Softwares de terceros, que se deben instalar manualmente en el servidor que albergue la aplicación web. Estos son:

- [libvips](#) v8.6+ or [ImageMagick](#) para el análisis y transformación de imágenes,
- [ffmpeg](#) v3.4+ para vistas previas de videos y [ffprobe](#) para el análisis de video y audio.
- [poppler](#) or [muPDF](#) para vistas previas de PDF.

Además, el análisis y transformación de imágenes requiere de la gema `image_processing`. Si no está incluida en el proyecto, se puede adherir manualmente al *gemfile*.

```
gem "image_processing", ">= 1.2"
```

Ilustración 2-4 Gema image_processing en Gemfile

También Active Storage permite generar representaciones de imagen de archivos no-imágenes como PDF y videos. Por último, permite extraer metadatos de archivos adjuntados.

Para poder vincular los archivos adjuntos a un Modelo particular, y, por ende, a las instancias de este. Active Storage hace uso y debe agregar a la base de datos de un proyecto 2 tablas extras (esto para Active Storage versión 5.2, en versiones más recientes puede opcionalmente crear una 3a tabla)

Las tablas creadas por Active Storage son las siguientes:

Active Storage Blobs

active_storage_blobs		
 id	bigint	« pk »
 key	character varying	« nn »
 filename	character varying	« nn »
 content_type	character varying	
 metadata	text	
 byte_size	bigint	« nn »
 checksum	character varying	« nn »
 created_at	timestamp	« nn »
 PK: id	constraint	« pk »
 Index on key	index	« u »

Ilustración 2-5 Tabla Active Storage Blobs, Fuente: Elaboración Propia con pgModeler

Esta tabla almacena metadatos sobre los archivos adjuntos en almacenamiento persistente (Amazon S3, Microsoft Azure, disco local, etc.). Sus campos son:

- **Key:** Un string identificador único para el blob (Binary Largo Object, en este caso un archivo). Este campo se usa para ubicar y recuperar un archivo particular del servicio de almacenamiento.
- **Filename:** Nombre original del archivo al ser subido.
- **content_type:** el tipo MIME (Multipurpose Internet Mail Extensions) del archivo (image/png, application/pdf, etc.)
- **Metadata:** Una columna con un JSON (Javascript Object Notation) que almacena metadatos adicionales, como las dimensiones de una imagen, entre otros.
- **Byte_size:** El tamaño del archivo en bytes.
- **Checksum:** un hash MD5 (Message Digest 5) codificado en Base64, del contenido del archivo.
- **Created_at:** un timestamp indicando la fecha y hora en la cual el registro fue creado.

Active Storage Attachments

active_storage_attachments		
id	bigint	« pk »
name	character varying	« nn »
record_type	character varying	« nn »
record_id	bigint	« nn »
blob_id	bigint	« fk nn »
created_at	timestamp	« nn »
◆ ▲ ▼		
PK: id		constraint « pk »
FK: blob_id -> Active_Storage_Blobs.id		constraint « fk »
Index on blob_id		index
Uniqueness(record_type-record_id-name-blob_id)		index « U »

Ilustración 2-6 Tabla Active Storage Attachments, Fuente: Elaboración Propia con pgModeler

Es una tabla de unión (*join table*) que conecta la tabla asociada a un Modelo de la aplicación (p.ej. users, memorandos, units, etc.) con la tabla active_storage_blobs anterior. Sus campos son:

- **Name:** el nombre del campo en el Modelo que referencia al adjunto.
- **Record_type:** El nombre de la Clase del Modelo a la cual el archivo está asociado. Parte de una asociación polimórfica
- **Record_id:** el ID en la tabla de datos del objeto al cual el archivo está asociado. Parte de una asociación polimórfica.

En conjunto con record_type, se establece una asociación polimórfica, la cual permite que adjuntos se puedan asociar a diferentes Modelos. Ambos campos se utilizan para identificar al dueño de un determinado adjunto.

- **Blob_id:** Llave foránea que referencia al campo id en la tabla active_storage_blobs. Este campo enlaza el attachment específico en la tabla active_storage_attachments a un blob específico en la tabla active_storage_blobs.
- **Created_at:** el timestamp indicando cuando el registro fue creado.

Relación

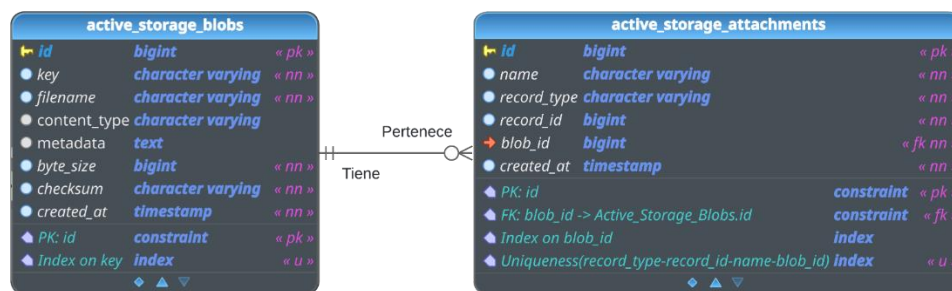


Ilustración 2-7 Relación entre Active Storage Blobs y Active Storage Attachments, Fuente: Elaboración Propia con pgModeler

Un Modelo puede tener muchos archivos adjuntos, y un mismo archivo adjunto puede ser adjuntado a más de un Objeto, y este Objeto puede ser de diferentes Modelos. Esto representa una relación Muchos-a-Muchos. Para poder almacenar y gestionar esto en una base de datos relacional, se debe crear una tabla intermedia, que tendrá asociaciones *belongs_to*, y por ende llaves foráneas, de ambas tablas que intermedia; tendrá una relación Muchos-a-Uno, con ambas.

Esta es exactamente la función de la tabla *active_storage_attachments*. Es la tabla intermedia que asocia un blob en la tabla *active_storage_blobs*, que posee los metadatos y referencia un archivo en almacenamiento persistente, con un registro o *record* de un Modelo.

Un registro de blob puede estar asociado a cero o muchos registros de attachments, mientras que un attachment solo puede estar asociado a uno y solo un blob. Por ende, un attachment le pertenece a un y solo a un blob, y un blob tiene cero o muchos attachments.

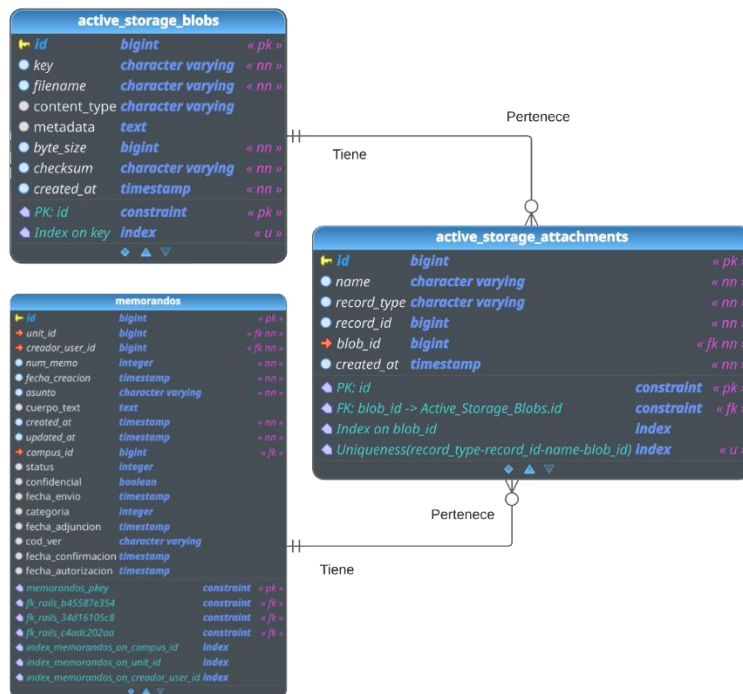


Ilustración 2-8 Relación entre Active Storage y Memorandos, Fuente: Elaboración Propia con pgModeler

En la imagen anterior podemos apreciar la tríada que conecta y vincula un archivo adjunto con un modelo.

Un attachment de *active_storage_attachments*, referencia a un blob de *active_storage_blobs*, por medio de la llave foránea *blob_id*, y a un memorando, de la tabla *memorandos*, por medio de la llave foránea polimórfica compuesta por *record_type* y *record_id*. Estos últimos campos permiten referenciar a cualquier modelo, en este caso, *record_type* = 'Memorando', siendo Memorando el Modelo en la aplicación que se comunica con la tabla *memorandos* en la base de datos. Luego, *record_id* = id del memorando. Con estos 2 campos se puede referenciar con exactitud el *record* dueño del *attachment*.

Por medio del attachment anterior, el blob especificado, que referencia y posee metadatos de un archivo adjunto en almacenamiento persistente, se vincula al memorando especificado, entendiéndose en la aplicación que este blob le pertenece a este memorando. Luego por medio del memorando, se puede acceder, presentar y modificar este blob según a conveniencia.

A partir de Ruby On Rails versión 5.2, se convirtió en una herramienta integrada de Rails, que viene preinstalada y fácilmente activable en aplicaciones ejecutando un par de comandos (Active Storage - Ruby On Rails , 2024).

En nuestro módulo se ocupa Active Storage para permitir almacenar y asociar archivos adjuntos a Memorandos específicos. Permitiendo asociar esta documentación extra requerida o complementaria del Memorando, con su contenido principal. Unificando todos los datos relacionados con un Memorando en un Modelo, el Modelo Memorando, con sus campos principales directos, más sus archivos adjuntos asociados.

```
# MODELO MEMORANDO -----  
class Memorando < ApplicationRecord  
  
  # Variables de Archivos Adjuntos  
  | # Manejados por Active Storage  
  has_many_attached :attachments  
  has_one_attached :scanned_sign_memorando
```

Ilustración 2-9 Campos de ActiveStorage en el Modelo de Memorando

Se presenta una sección del Modelo de Memorando. Podemos ver 2 líneas *has_many_attached* y *has_one_attached*, ambas son declaraciones para configurar que el Modelo Memorando tenga campos adicionales que se vinculan con archivos adjuntos, los cuales son vinculados y gestionados por Active Storage. En el primero, se define que un memorando puede tener cero o varios archivos adjuntos, asociados a un nuevo campo llamado *attachments*. En segundo lugar, se define que puede poseer cero o hasta un archivo adjunto, referenciado por su campo *scanned_sign_memorando*.

2.2.5 CanCanCan

CanCanCan es una gema (librería) de autorización para Ruby y Ruby on Rails que provee funcionalidad de permisos y autorización a recursos y métodos de una aplicación web. Restringe a qué recursos puede acceder un usuario determinado.

Funciona definiendo los permisos en un Modelo Active Record definido en el archivo llamado *ability.rb*, que se autogenera al configurar la gema. También se puede definir permisos en varios archivos *ability*, para modularizar. Esto permite que la lógica de los permisos se defina en un solo lugar, evitando que las definiciones de los permisos se dupliquen en los Controladores, Vistas o consultas a la base de datos, facilitando su mantención, expansión y prueba.

La gema está compuesta de dos partes principales:

Una biblioteca para autorizaciones, que corresponde a sus métodos y variables para definir las reglas y permisos para acceder a diferentes objetos. También, provee de métodos *helpers* que se pueden invocar en diferentes lugares para verificar los permisos definidos.

Provee otro set de métodos helpers especializados para simplificar el código en los Controladores de Rails, encargándose de la carga y verificación de permisos de Modelos automáticamente y reducir el código duplicado en estos.

Provee otro set de métodos *helpers* para simplificar el código en los Controladores de Rails. Estos se encargan de cargar instancias de modelos asociados al controlador y de revisar los permisos asociados a tales modelos automáticamente. De esta manera se reduce el código duplicado en los métodos del Controlador.

Para instalarle en el proyecto se puede agregar al *gemfile* la línea

```
gem 'cancancan'
```

Ilustración 2-10 Línea gemfile para CanCanCan

Luego ejecutando el comando a continuación por consola, se crea la clase Ability, en el archivo *ability.rb*, archivo en donde se definen los permisos y los *helpers* personalizados para consultar por permisos tanto en las Vistas y Controladores.

```
rails generate cancan:ability
```

Ilustración 2-11 Comando para inicializar CanCanCan

Con el comando se crea la clase más una pequeña plantilla de la clase Ability:

```
# /app/models/ability.rb

class Ability
  include CanCan::Ability

  def initialize(user)
  end
end
```

Ilustración 2-12 Plantilla de la Clase Ability de CanCanCan

CanCan::Ability es la clase base con el funcionamiento y métodos internos que nos permiten crear permisos y checks personalizados de los Usuarios de la aplicación.

Hay 2 métodos básicos en CanCanCan:

```
can actions, subjects, conditions
```

Ilustración 2-13 Métodos Que Definen Permisos de CanCanCan

Los que se usan para definir que el Usuario puede realizar ciertas *Actions* sobre ciertos *Subjects*, si se cumplen ciertas *Conditions*.

```
can? action, subject
```

Ilustración 2-14 Métodos Que Consultan Permisos de CanCanCan

Can? (con el signo de interrogación) corresponde al método que se usa para consultar si un Usuario está autorizado para realizar cierta *Action* sobre cierto *Subject*.

El primer tipo de métodos se definen en *ability.rb*, en la clase *Ability*.

Si tomamos como ejemplo el Modelo Memorando, tenemos los siguientes ejemplos:

```
1 - can :read, Memorando
2 - can :create, Memorando, unit_id: 2
3 - can [:read, :update], @memorando, { user: user, unit_id: 2 }
```

Ilustración 2-15 Ejemplos de Permisos Definidos para Modelo de Memorando

Como descrito, un permiso se define con la palabra clave *can* seguido de una acción que actúa sobre un sujeto, y en donde se cumplen ciertas condiciones.

En el primer ejemplo, se declara que el Usuario actual puede realizar la acción con etiqueta *:read* sobre el modelo Memorando. Quiere decir el Usuario actual tendrá autorización de realizar cualquier acción sobre una instancia de Memorando o sobre la misma clase, esté protegida bajo la etiqueta *:read*.

En el segundo ejemplo, se declara que el usuario actual puede realizar cualquier acción sobre una instancia o sobre la clase Memorando, que esté resguardada bajo la etiqueta *:create*, pero solo si la unidad de este Memorando es la con id 2, que en la Intranet corresponde a la Unidad de Pregado. Es decir, el usuario actual solo podrá *:create* Memorandos de clase de pregrado

En el tercer ejemplo, tenemos un arreglo de acciones disponibles para el usuario actual, pero esta vez sobre una instancia u objeto de la clase Memorando específico, no sobre la clase en sí. Además, tenemos un hash con condiciones, que indican que el objeto @memorando debe tener como dueño al usuario actual y debe pertenecer a la unidad 2, es decir, Pregrado.

Cabe mencionar que las acciones son en estricto rigor solo etiquetas, no son necesariamente métodos de algún Controlador, uno puede ocupar una etiqueta y luego revisar ese permiso en un método de algún Controlador o Vista sin ese nombre explícito. Pero, si se usa una etiqueta de una acción que sí corresponde a un método de un Controlador, luego, con la ayuda de los Helpers especializados, se puede automatizar la autorización, consultando en el método con nombre *Action*, el permiso asociado a la etiqueta *:Action*. Lo mismo aplica para el *Subject*, no está limitado a ser un Modelo o instancia de uno, puede ser etiqueta general, que luego se puede declarar el permiso asociado en el lugar deseado.

Como mencionado, Rails sigue el patrón MVC y el principio RESTful de acceso a recursos cuya gestión se basa primordialmente en operaciones CRUD (Create, Read, Update, Delete). CanCanCan ofrece etiquetas predeterminadas que se mapean a métodos por defecto y presentes en la gran mayoría de controladores, estos son:

```
read: [:index, :show]
create: [:new, :create]
update: [:edit, :update]
destroy: [:destroy]
```

Ilustración 2-16 Etiquetas Predefinidas de CanCanCan

Es decir, un usuario actual con el permiso de :read, quiere decir que puede ver el listado de tal Modelo (:index) y la Vista Detalles de una instancia específica (:show).

En Rails, el método :new del Controlador invoca al formulario de creación de un objeto, y el método :create se encarga de crear un nuevo objeto a partir de lo que el usuario completó en el formulario. Como todo aquel que pueda crear un objeto de un determinado Modelo requiere poder acceder al formulario de creación (:new) y a la vez a la lógica de creación (:create), CanCanCan agrupó ambas acciones en una sola, en la etiqueta :create. Lo mismo aplica con :edit y :update, bajo la etiqueta :update, siendo el equivalente a :new y :create, respectivamente.

```
4 - can :remitir, Memorando, { user: user, unit_id: 2 }  
5 - can :read, :admin_dashboard, if user.admin?  
6 - can :manage, Memorando  
7 - can :manage, :all
```

Ilustración 2-17 Más Ejemplos de etiquetas de CanCanCan

En el cuarto ejemplo se presenta una etiqueta personalizada, que puede corresponder a un método existente en el `MemorandosController` o no.

En el quinto ejemplo, se muestra un caso en donde el sujeto no es un Modelo, sino que solo una etiqueta personalizada. Este representa que el usuario actual puede acceder a la acción `:read` del sujeto `:admin_dashboard`, en caso de ser el usuario administrador. Es decir, que el usuario puede leer (acceder a listado y vista detalles) un objeto, método o sección de una vista que se declare como `:admin_dashboard`.

En los ejemplos sexto y séptimo se presenta otra etiqueta proveída por CanCanCan, `:manage`, que es una etiqueta especial que declara que se tiene acceso a todos los métodos públicos asociados al sujeto. En este caso se declara acceso a todos los métodos y acciones asociadas a la clase `Memorando`.

Por último, en el séptimo ejemplo, vemos la etiqueta de sujeto `:all`, también proveída por CanCanCan, que denota cualquier sujeto: objeto, clase o genérico. Por ende, la línea 7 declara acceso completo, a todos los métodos asociados a un sujeto, para cualquier sujeto. En otras palabras, genuino acceso completo.

Luego los permisos se pueden consultar con el equivalente `can?`:

```
1 - can? :read, Memorando
2 - can? :create, Memorando
3 - can? [:read, :update]
4 - can? :remitir, Memorando
5 - can? :read, :admin_dashboard
6 - can? :manage, Memorando
7 - can? :manage, :all
```

Ilustración 2-18 Ejemplos de Consulta de Permisos con CanCanCan

Respecto a los Helpers especializados para Controladores, se tienen principalmente los siguientes:

Authorize_resource: por cada método del Controlador, llama a

```
authorize! action_name, @objeto

# Ejemplo

authorize! action_name, @memorando
```

Ilustración 2-19 Helper especializado Authorize

Es decir, asocia el nombre de un método en el Controlador a una etiqueta de *action*, y aplica tal etiqueta a la variable de instancia *@objeto*, correspondiente. P. ej. En el caso del *MemorandoController*, los permisos se aplican sobre la variable de instancia *@memorando*. De este modo, no se tiene que especificar explícitamente para cada uno de los métodos su declaración *authorize!* o *can?* correspondiente para consultar el permiso.

Load_resource: carga la variable de instancia *@objeto* dependiendo del Controlador. Por ejemplo, en el caso de *MemorandosController*, para el método *new*, *edit*, o *show*, cargara la variable de instancia con nombre *@memorando*, obteniéndola desde el Modelo *Memorando* automáticamente

Load_and_authorize_resource: Hace ambas acciones dependiendo del Controlador y el método invocado. En el caso del *MemorandoController*, para los métodos *new*, *edit* o *show*, primero cargará la variable de instancia *@memorando*, obteniéndola del Modelo *Memorando*. Una vez obtenido, evaluará si los permisos del usuario actual le permiten acceder a aquellos métodos. Si se desea acceder a *show*, se evaluará: *can? :show, @memorando* o *authorize! :show, @memorando*

Adelantándonos un poco, se muestra a continuación, una porción del MemorandosController

```
class MemorandosController < ApplicationController
  include MemorandosHelper # Funciones del Helper de memorandos.

  before_action :logged? # Requerir que el usuario esté logueado en todas las acciones.
  # Se busca el memo con :id recibido
  before_action :ensure_js_format, only: [:confirm, :store, :authorize]

  load_and_authorize_resource :memorando, includes: :memo_participantes, only: [:show, :edit, :update, :destroy,
  :templating, :creating_from_template]

  load_and_authorize_resource :memorando, only: [:confirm, :confirming, :disconfirm, :store, :storing, :authorize, :authorizing]

  before_action :check_memorando_status, only: [:edit, :update, :destroy, :confirm, :confirming] # Revisa si el valor de statu
  # Estado 0: memo aún
  # funciones listadas
  # (por un usuario co
  # En caso contrario,
  # funciones listadas

  # Permisos y autorización de CanCanCan de Memos. Para todas las acciones.
  authorize_resource :memorando
```

Ilustración 2-20 Helpers Load_and_authorize_resource de CanCanCan

Se puede visitar su repositorio de Github para tener acceso a la documentación y código fuente.

2.2.6 El Portal de la Intranet del Departamento de Informática

Corresponde a una aplicación web diseñada para ser usada de manera interna por los miembros del DI. Ha sido progresivamente elaborada por diferentes miembros del DI, en especial alumnos de la UTFSM, desde alumnos en práctica, ayudantes, desarrolladores y administradores del Data Center del DI, profesores, etc.

Se encuentra disponible en <https://intranet.inf.utfsm.cl>. Se puede acceder a ella desde fuera de la red interna del DI, pero se requiere el registro de una cuenta, proceso en manos del administrador del Data Center del DI. Además, muchas de sus funcionalidades solo están disponibles si se posee el nivel de autorización adecuado.

Está orientada a mejorar la productividad, la comunicación interna y el acceso a recursos compartidos del DI.

Su objetivo es facilitar el trabajo, la gestión y la comunicación en el Departamento, mejorando la productividad de sus miembros.

Dentro de las funciones que permite están la revisión y autorización de las solicitudes de salida fuera de la UTFSM; la creación de pases de visita para la red WiFi del DI y un sistema de gestión de Tickets de asistencia.

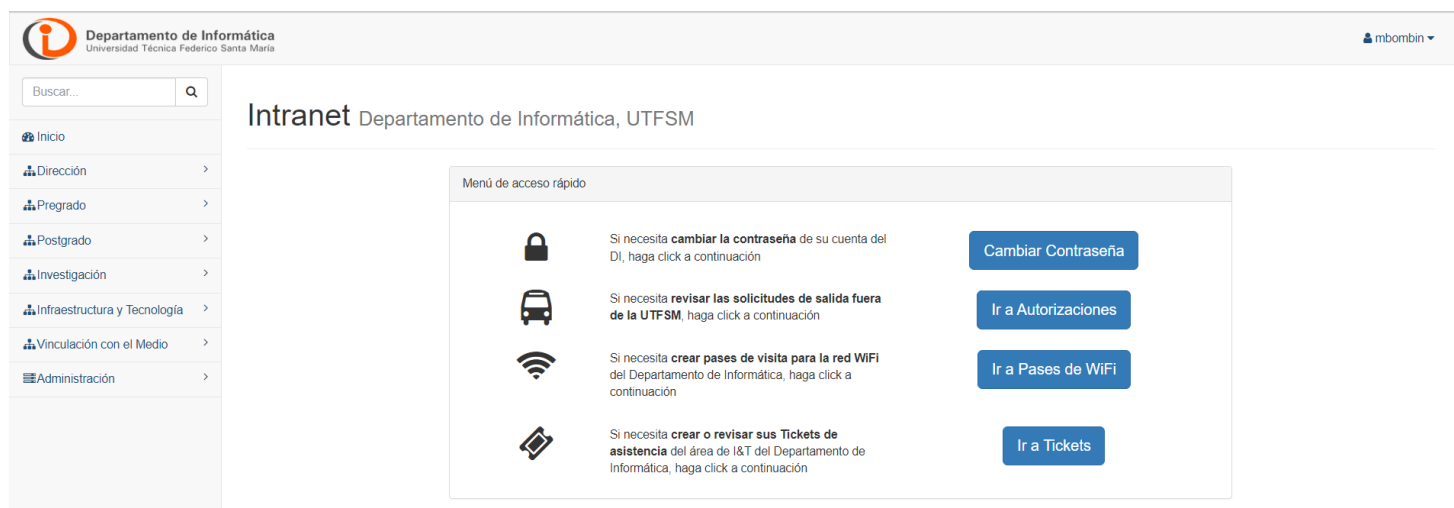


Ilustración 2-21 Página de Inicio de la Intranet, Fuente: La Intranet del DI

Además, habilita y restringe su funcionalidad y recursos compartidos dependiendo de la Unidad a la que pertenece el Usuario actual y su nivel de permisos. La Sgte. imagen pertenece al Menú Lateral del Administrador.

También, permite la gestión, acceso y compartición de documentos internos importante para que sus miembros puedan llevar a cabo sus tareas y para el funcionamiento del DI.

Es un sistema que ya posee más de 10 años de existencia y uso, con el cual sus miembros se encuentran familiarizados.

Debido a ser una Intranet a cargo de la gestión de las operaciones del DI, encargándose de la creación de Tickets y de Autorizaciones, y de sus recursos compartidos, como sus documentos, además de ser usada diariamente por sus miembros, quienes están familiarizadas con ella, es que se decidió el analizar e implementar un prototipo como un Módulo para este portal web. Centralizando y digitalizando la gestión de Memorandos, no solo en el Módulo, pero también en conjunto con las demás tareas administrativas

Mostrando un poco del prototipo, en este, si se poseen los permisos de Gestión de Memorandos, entonces aparecerán las funcionalidades en el Menú Lateral:



Ilustración 2-22 Página de Inicio de la Intranet con Sección de Memorandos



Ilustración 2-23 Zoom a Sección de Memorandos del Menú Lateral de la Intranet

Capítulo 3: PROPUESTA DE SOLUCIÓN

La solución corresponde a la implementación de un módulo de extensión del portal de la Intranet del Departamento de Informática, que se encarga de la gestión general de los Memorandos.

Otra propuesta de solución involucraba la creación de un nuevo sistema independiente, a diseñar e implementar desde cero.

Se decantó por desarrollar un módulo para la Intranet, por:

Un nuevo sistema independiente desde cero representaba un proceso de diseño y desarrollo más extenso. Ya que involucraba crear la arquitectura e infraestructura base. Al realizar un módulo, puede concentrarse solo en el diseño y desarrollo del módulo específico, sin pensar en otras funciones necesarias, como el sistema de autenticación, permisos, ventanas, etc. que ya se encuentran desarrolladas y disponibles.

Por otra parte, ya existía en el DI el plan de ir digitalizando procesos relacionados a los miembros del DI extendiendo la Intranet. Siendo uno de esos proyectos propuestos deseables, el digitalizar la gestión de Memorandos.

Además, se confirmó la creación de un módulo para la Intranet porque corresponde a un portal con el cual los colaboradores del DI se encuentran familiarizados con él y ocupan diariamente, principalmente los miembros de las Secretarías, quienes se encargan de la Gestión de los Memorandos.

Por último, el sistema propuesto involucraba la tarea puntual del manejo de Memorandos, que suele estar ligado a la gestión del DI, al igual que la gestión de sus documentos, permisos, autorizaciones, etc. Por lo cual, en vez de crear un sistema completo especializado, se decidió extender un sistema existente y familiar para los miembros, agregándole esta función especializada.

3.1 Descripción General de la Solución

El módulo provee de 4 servicios principales:

- Interfaz de edición y creación;
- Validación digital por el Emisor correspondiente;
- Envío, al Emisor emisora para ser validado y autorizado,
- Envío a los destinatarios una vez autorizado; y
- Búsqueda en base de datos mediante diferentes filtros.

El módulo fue desarrollado ocupando el *framework Ruby on Rails* versión 5.2.8.1 y lenguaje Ruby versión 2.7.7.

Al ocupar *RoR*, se adhiere a la arquitectura *Model-View-Controller* (MVC). Los componentes de la solución son:

3.2 Diseño

3.2.1 Requerimientos

3.2.1.1 Criterios

Un requerimiento tiene un Tipo y una Prioridad.

El tipo puede ser Obligatorio o Deseable. Un requerimiento obligatorio se debe implementar sí o sí, mientras que uno Deseable es opcional.

La prioridad se mide cualitativamente en: Alta, Media y Baja.

Para medir la prioridad se ocuparon los siguientes criterios:

- C1. El requerimiento es primordial para el funcionamiento básico del módulo.
- C2. El requerimiento representa alguno de los Objetivos descritos.
- C3. El requerimiento era altamente recurrente al entrevistar diferentes actores.
- C4. El requerimiento levantaba interés en diferentes actores.
- C5. El tipo del requerimiento, los obligatorios poseerán generalmente una mayor prioridad que uno deseable.
- C6. El requerimiento puede significar una buena adición a largo plazo.
- C7. Dificultad de implementación. Si es fácil de implementar, se le asignará menor prioridad.
- C8. Si es un requerimiento, que incluso si representa a un Objetivo, depende de que otros requerimientos más primordiales se implementen primero o que puede esperar, verá su prioridad disminuida.
- C9. Es una extensión o está cercanamente relacionado con un requerimiento marcado con prioridad alta

C10. Representa un potencial campo obligatorio

3.2.1.2 Listado de Requerimientos

En el listado a continuación se presentan los requerimientos levantados. Se señala su clave o id, su nombre, se describe, se determina su Tipo y una medida de su Prioridad.

RF: Requerimiento Funcional

RNF: Requerimiento no Funcional

Tabla 3-1 Tabla de Requerimientos

Clave	Nombre	Descripción	Tipo	Prioridad
RF-1	Interfaz de Edición	Se debe implementar una interfaz gráfica en donde se pueda crear un Memorando, llenando distintos campos en un formulario	Obligatorio	Alta: C1, C2 (Ver Criterios)
RF-1.1	Logo de la Universidad	El logo de la Universidad se debe incluir automáticamente.	Obligatorio	Baja
RF-1.2	Número correlativo	Los Memorandos deben tener un número correlativo que los identifique. La cadena tendrá un formato asociado: “DI- $\{Siglas\}$ unidad}- $\{ \#Memorando \}$ /AA”. Siendo AA formato de año corto.	Obligatorio	Alta: C2 (Ver Criterios)
RF-1.2.1	Número correlativo autogenerado	El Número Correlativo se debe generar automáticamente.	Obligatorio	Alta: C2, C4 (Ver Criterios)
RF-1.2.2	Número o cadena correlativa manual	Temporalmente, mientras se integra el sistema al Departamento, se debe permitir agregar la cadena correlativa manualmente	Deseable	Baja:
RF-1.2.3	Reinicio de número de Memorando	Se debe reiniciar a 0 el número de Memorando al comenzar un nuevo año.	Obligatorio	Baja
RF-1.3	Fecha de creación, remisión, autorización y envío automáticas	Las fechas de creación, remisión, autorización y envío se generarán automáticamente: -La fecha de creación al crear el Memorando, como borrador.	Obligatorio	Alta: C1, C2 (Ver Criterios)

		-La fecha de remisión, al remitir el Memorando a ser autorizado. -La fecha de autorización al autorizar el Memorando por su Emisor. -La fecha de envío al enviar el Memorando al destinatario y Con Copias.		
RF-1.4	Área de Texto Enriquecido	Se debe contar con un editor de texto enriquecido para el cuerpo del Memorando.	Obligatorio	Alta: C1, C2 (Ver Criterios)
RF-1.4.1	Archivos Adjuntos	Se debe poder adjuntar archivos PDF y MS-Word.	Obligatorio	Alta: C3, C4 (Ver Criterios)
RF-1.5	Emisor	Debe existir un campo para ingresar el nombre, el cargo y el Dpto. del Emisor del Memorando.	Obligatorio	Alta: C1 (Ver Criterios)
1.5.1	Múltiples Emisores	Un Memorando puede tener más de un Emisor	Deseable	Media: C6 (Ver Criterios)
RF-1.6	Información de campus autogenerada	Se debe autogenerar dependiendo del campus, información la dirección, fono y mail de la unidad.	Deseable	Baja
RF-1.7	Destinatarios	Debe existir un campo donde se señala el destinatario, especificando su nombre completo, cargo y Departamento.	Obligatorio	Alta: C1 (Ver Criterios)
RF-1.7.1	Múltiples Destinatarios	Puede haber múltiples destinatarios (Con Copias)	Obligatorio	Alta: C3, C4, C6 (Ver Criterios)
RF-1.8	Correo electrónico	Se debe poder especificar el correo electrónico del o los destinatarios (opcionalmente). Este servirá para su posterior envío una vez autorizado.	Obligatorio	Alta: C1, C2

				(Ver Criterios)
RF-1.9	Asunto	Se debe poder especificar el asunto del Memorando	Obligatorio	Alta: C1 (Ver Criterios)
RF-1.10	Generación de PDF de Memorando	Al crear un Memorando, se debe generar un archivo PDF con la información ingresada.	Obligatorio	Alta: C1, C2 (Ver Criterios)
RF-1.10.1	Archivos adjuntos aparte	Los archivos adjuntos no serán parte del PDF principal, pero podrán ser descargados de la vista en detalle del Memorando.	Obligatoria	Alta: C1, C2, C3 (Ver Criterios)
RF-1.11	Opción de guardar PDF	Una vez generado el PDF, se da la opción de poder guardar una copia localmente.	Obligatorio	Medio: C5 (Ver Criterios)
RF-2	CRUD de Memorandos	Modelar en una entidad los Memorandos con las operaciones <i>CRUD</i> .	Obligatorio	Alta: C1, C2 (Ver Criterios)
RF-3	Autorización de Memorandos	El Emisor correspondiente debe ser capaz de firmar o validar un Memorando por medio de la interfaz.	Obligatorio	Alta: C1, C2 (Ver Criterios)
RF-3.1	Autorización Múltiples Emisores: Principal y Secundarios	En caso de múltiples emisores, todos deben autorizar para que un Memorando sea válido. Habrá un emisor principal, que no podrá autorizar hasta que todos los demás lo hayan hecho.	Deseable	Baja:
RF-4	Envíos	El sistema debe permitir el envío por correo electrónico de los Memorandos	Obligatorio	Media: C2 (Ver Criterios)

RF-4.1	Envío al Emisor	Una vez creado un Memorando, el Redactor puede remitirlo a la lista de pendientes por autorizar del Emisor, ocupando su correo electrónico del DI.	Obligatorio	Alta: C1, C2 (Ver Criterios)
RF-4.2	Envío a los destinatarios	El Emisor al autorizar un Memorando, se debe enviar por correo electrónico a los destinatarios (principal y con copia) si se les especificó uno.	Obligatorio	Media: C2 (Ver Criterios)
RF-5	Listados y Búsquedas	-Se debe tener acceso a Listados de Memorandos de diferente tipo. -La búsqueda de Memorando se podrá filtrar por distintos criterios.	Obligatorio	Alta: C1, C2, C3, C4 (Ver Criterios)
RF-5.1	Visibilidad por unidad	Redactor o Emisor debe ver solo Memorandos asociados a su Unidad. Excepción aquellos de Dirección, que deben ver Memorandos de cualquier Unidad.	Obligatorio	Baja: C3 (Ver Criterios)
RF-5.2	Listado de Memorandos Borradores	Redactor debe tener acceso a listado Memorandos Borradores, es decir, Memorandos creados, pero aún no remitidos.	Obligatorio	Media: C1, C2, C3, C7 (Ver Criterios)
RF-5.3	Listado de Memorandos Plantillas	Redactor debe tener acceso a Listado de Memorandos Plantillas, es decir, esqueletos de Memorandos guardados para facilitar la creación de nuevos Borradores.	Obligatorio	Media: C3, C7 (Ver Criterios)
RF-5.4	Listado Memorandos Remitidos	Redactor debe tener acceso a Listado Memorandos que ha remitido para ser autorizados. Permanecerán en ella hasta ser autorizados.	Obligatorio	Media: C1, C2, C7 (Ver Criterios)
RF-5.5	Listado de Memorandos Por Autorizar	Emisor debe tener acceso a listado Memorandos pendientes por autorizar, pudiendo ver sus detalles, autorizarlos o rechazarlos.	Obligatorio	Alta: C1, C2, C3 (Ver Criterios)

RF-5.6	Listado de Memorandos Autorizados	Emisor debe tener acceso a Listado de Memorandos autorizados. Estos desaparecerán de la lista de Memorandos por autorizar.	Obligatorio	Media: C2, C7 (Ver Criterios)
RF-5.7	Listado Memorandos Rechazados	Redactor debe tener acceso a Listado Memorandos que han sido rechazados por el Emisor. Estos se pueden volver a remitir o eliminar	Obligatorio	Media: C7, C8 (Ver Criterios)
RF-5.8	Búsqueda por Rango de Fechas	-En los listados de Memorandos, se debe poder buscar por rango de fechas: -En la lista de borradores, por fecha de creación. -En la lista de Remitidos y en la de Pendientes por Autorizar, por fecha de remisión. -En la lista de Memorandos emitidos (autorizados), por fecha de autorización.	Obligatorio	Media: C2, C8 (Ver Criterios)
RF-5.9	Campo de Texto de Búsqueda General	En los listados, se debe incluir un campo de texto para poder realizar una búsqueda general. Lo que se ingrese se usará para buscar Memorandos que coincidan total o parcialmente con lo ingresado en diferentes campos.	Obligatorio	Media: C2, C3, C8 (Ver Criterios)
RF-5.9.1	Búsqueda por Asunto	- Se debe poder buscar el Asunto ingresando en campo de texto de búsqueda general, devolviendo los que concuerden total o parcialmente	Obligatorio	Media: C2, C3, C8 (Ver Criterios)
RF-5.9.2	Búsqueda por Número Correlativo	-Se debe poder buscar el Número Correlativo ingresando en campo de texto de búsqueda general, devolviendo los que concuerden total o parcialmente	Obligatorio	Media: C2, C3, C4, C8 (Ver Criterios)
RF-5.9.3	Búsqueda por Emisor	Se debe poder buscar por Emisor, especificando su nombre, cargo o Departamento.	Obligatorio	Media: C2, C3, C4, C8 (Ver Criterios)
RF-5.9.4	Búsqueda por Destinatario Principal	Se debe poder buscar por destinatario principal, especificando su nombre, cargo o Dpto.	Obligatorio	Media: C2, C3, C4, C8

				(Ver Criterios)
RF-5.10	Búsqueda por Unidad	Se debe poder buscar por Unidad	Obligatorio	Media: C2, C3, C4, C8 (Ver Criterios)
RF-6	Distintos Roles de Participantes de Memorando	Se debe crear una entidad que encapsule y almacene los diferentes participantes que puede poseer un Memorando, como mínimo 2, un Emisor y un Destinatario. Un mismo usuario puede jugar diferentes roles en distintos Memorandos	Obligatorio	Alta: C1, C2 (Ver Criterios)
RF-6.1	Nombre Completo	Se debe registrar el nombre completo del participante	Obligatorio	Alta: C1 (Ver Criterios)
RF-6.2	Cargo	Se debe registrar el cargo del participante	Obligatorio	Alta: C1 (Ver Criterios)
RF-6.3	Departamento	Se debe registrar el Departamento del participante	Obligatorio	Alta: C1 (Ver Criterios)
RF-6.4	Correo Electrónico	Se debe registrar el correo electrónico del participante	Obligatorio	Alta: C1 (Ver Criterios)
RF-6.5	Rol	Se debe registrar el rol que cumple el participante en el rol en el cual participa, como mínimo si es Emisor o Destinatario	Obligatorio	Alta: C1 (Ver Criterios)
RF-6.5.1	Posibles valores de Rol	Rol debe tener valores posibles específicos. Estos son: Emisor Principal, Destinatario Principal, Destinatario Secundario (Con Copia), Emisor Secundario.	Obligatorio	Medio C8 (Ver Criterios)

RF-6.6	UID de emisor	Se debe registrar el UID de los participantes que posean algún rol de Emisor (Emisor Principal o Secundario). Esto es para vincular al usuario de informática con el Memorando en la Intranet.	Obligatorio	Alta: C1, C2 (Ver Criterios)
RF-7	CRUD de Contactos	Se debe modelar una entidad de Contactos. Todos los actores deben poder gestionar su propia lista <i>Contactos</i> para usarlos para autocompletar destinatarios y emisores rápidamente en la interfaz al crear un Memorando.	Obligatorio	Alta: C2, C3, C4 (Ver Criterios)
RF-7.1	Contactos Individuales de cada usuario	Los usuarios deben poseer su propia lista de Contactos	Obligatorio	Alta: C9 (Ver Criterios)
RF-7.2	Privilegios de Administrador Respecto a los Contactos	El administrador debe tener acceso y poder gestionar los Contactos de todos los usuarios.	Obligatorio	Baja
RF-8	Interfaz de Creación de Contactos	Se debe implementar una interfaz gráfica en donde se pueda crear un Contacto, llenando distintos campos en un formulario	Obligatorio	Alta: C2, C3, C4, C9 (Ver Criterios)
RF-8.1	Nombre Completo	Se debe poder especificar el Nombre Completo del Contacto. Debe ser un campo obligatorio.	Obligatorio	Alta: C9, C10 (Ver Criterios)
RF-8.2	Cargo	Se debe poder especificar el cargo del Contacto. Debe ser opcional.	Obligatorio	Alta C9 (Ver Criterios)
RF-8.3	Departamento	Se debe poder especificar el cargo del Contacto. Debe ser opcional.	Obligatorio	Alta C9 (Ver Criterios)
RF-8.4	UID de Informática	Se debe poder especificar el UID de informática del usuario. Debe ser opcional	Obligatorio	Alta: C9

				(Ver Criterios)
RF-8.4.1	Enlazamiento Usuario con Contacto	Al especificar un UID, se debe poder referenciar al usuario con tal UID desde el contacto y viceversa.	Obligatorio	Alta: C9 (Ver Criterios)

3.2.2 Diagrama de Entidades y Relaciones de Base de Datos

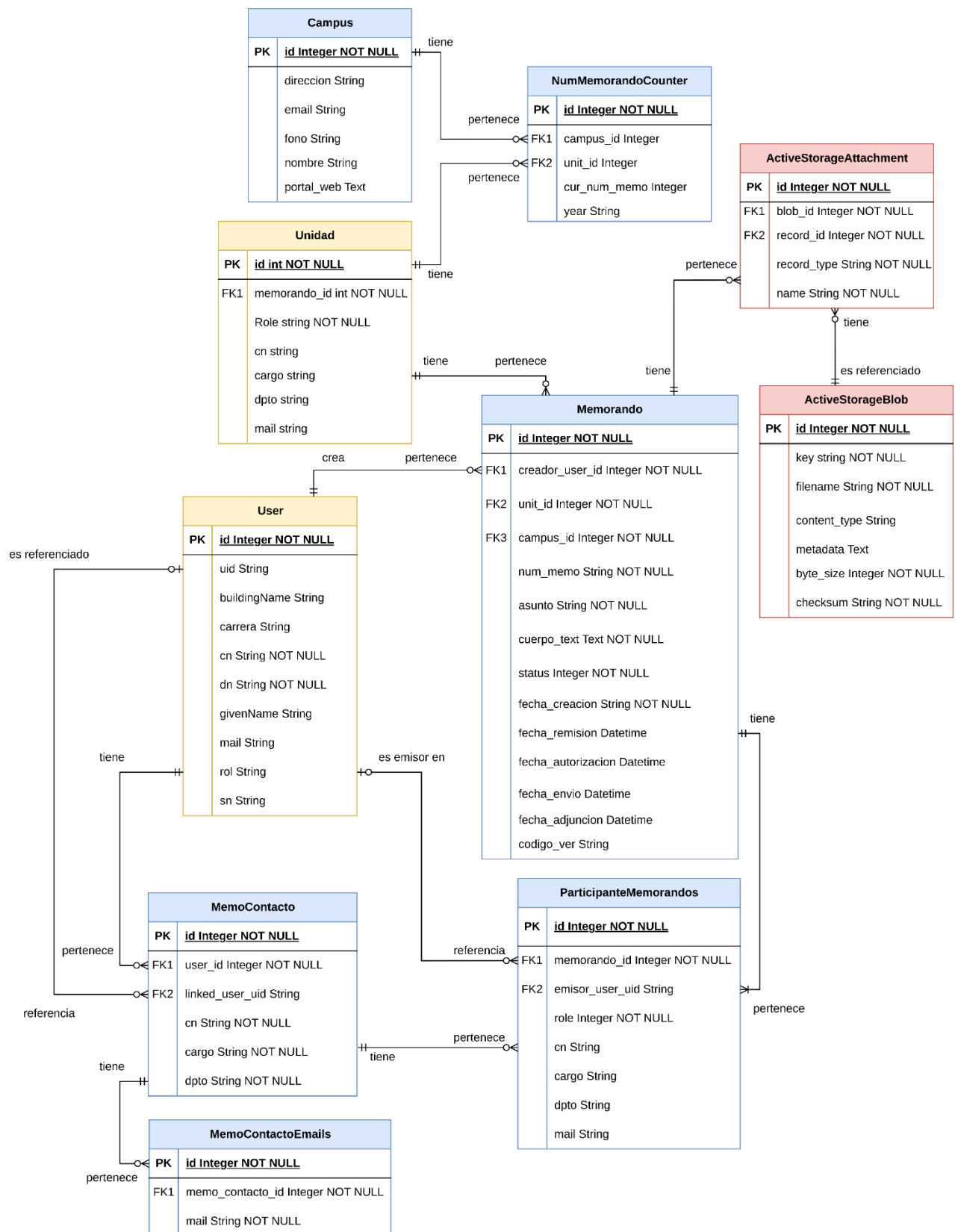


Ilustración 3-1 Diagrama de Base de Datos del Prototipo

3.2.3 Estructura y Campos de un Memorando

Al momento del levantamiento de requerimientos y para la familiarización con los documentos de Memorandos y su proceso actual de gestión. Se calendarizaron entrevistas y conversaciones con diferentes participantes del DI, con diferentes roles y pertenecientes a distintas unidades.

Para asegurar el uso, finalidad y proceso de gestión y lograr descifrar correctamente la estructura y campos de un Memorando, además de detectar oportunidades de mejora en el proceso, se configuró una pauta con las preguntas relevantes y pertinentes ([Revisar Anexo](#)).

La pauta tenía como entrevistado objetivo a los encargados de las Secretarías del DI. Siendo estos funcionarios los más informados y conectados con el tema. La pauta evolucionaba a medida que se llevaban a cabo más entrevistas, y a pesar de ser básicamente la misma para todas las Secretarías, dependiendo de la información obtenida en entrevistas previas, se modificaban o se omitían ciertas preguntas.

Luego, se les mostraron resultados parciales de implementación para obtener su retroalimentación, guiando el desarrollo.

Se entrevistó a las secretarías de todas las Unidades: Dirección, Pregrado, Postgrado, Vinculación con el Medio e Infraestructura, Tecnología & Servicios, pero el mayor involucramiento e interés provino y se mantuvo persistente, por parte de las unidades de Dirección y Pregrado.

También se consultó y obtuvo retroalimentación con otros actores, como el Ingeniero de Sistemas y la encargada de Gestión Docente, pertenecientes a Dirección.

Del análisis de las entrevistas y conversaciones, del análisis de las muestras de Memorandos otorgadas, y luego del ajuste y correcciones en respuesta a la retroalimentación al mostrar prototipos, se confeccionó el siguiente modelo de un Memorando con sus respectivas funcionalidades, como se detallarán posteriormente.

Un Memorando posee los siguientes campos: pueden ser obligatorios, el usuario al confeccionar un Memorando debe ingresarlos; opcionales, no se requiere que se ingresen o especifiquen; y autogenerados, se agregan automáticamente, pudiendo ser estos últimos, dependientes o independientes del usuario.

3.2.3.1 Logo de la Universidad

Imagen del logo de la Universidad, es agregada automáticamente a los Memorandos.

3.2.3.2 Número Correlativo

Identificación única del Memorando, se compone de las siglas del Departamento de Informática, de las siglas de la Unidad, del último número al momento de emisión de Memorandos emitidos por la unidad, y el año de emisión, en formato corto AA.

El formato corresponde a

“DI-*{Sigla Unidad}*-*{#Memorando}*/AA”

Ejemplo:

“DI-DIR-001/24”

Este corresponde a un número correlativo de un Memorando perteneciente al Departamento de Informática (DI), de la unidad de dirección (DIR), el número 001, es decir, el primer memo emitido, del año 2024.

3.2.3.3 Destinatarios

Es la persona a la que va destinado el Memorando. Este está compuesto por

- Nombre: El nombre del destinatario
- Cargo: El cargo que posee el destinatario
- Departamento: El Departamento al que pertenece el destinatario
- Correo Electrónico (opcional): se opcionalmente especificar un correo electrónico al cual se enviará el Memorando una vez autorizado por el emisor correspondiente.

Los Memorandos poseen solamente uno principal, señalado en el encabezado del documento, varios en calidad de Con Copia (CC), solo se señala su correo electrónico. El principal es obligatorio.

La implementación en la base de datos permite, si se requiere, luego de realizados los ajustes correspondientes, la inclusión de varios destinatarios con todos sus datos.

3.2.3.4 Emisor

Colaborador del DI que emite y valida el Memorando. Debe tener cuenta en la Intranet para poder autorizar. Se le debe especificar como mínimo:

- UID: el UID de informática del usuario. Se requiere para vincular el Emisor con su correspondiente *role*, permisos y Memorandos por autorizar.
- Nombre: El nombre completo del Emisor.
- Cargo: El cargo del Emisor.

- Departamento: El Departamento al cual pertenece el Emisor. Todos los emisores deben pertenecer al DI, si no, no pueden ser emisores.
- Correo Electrónico: El correo electrónico institucional de informática del usuario.

Los Memorandos poseen un emisor señalado en el encabezado, solo él puede autorizar el Memorando. La implementación permite, si se requiere, luego de realizados los ajustes correspondientes, la inclusión de varios emisores.

Se asigna a los usuarios como el director, subdirectores de las unidades, jefe de Carrera, los profesores, y cualquier otro colaborador del DI que lo requiera.

3.2.3.5 Fecha de Creación

Corresponde a la fecha *datetime* en la cual el Memorando fue creado por primera vez. Ocurre a la hora de crear un borrador o una plantilla

3.2.3.6 Fecha de Remisión o Confirmación

Corresponde a la fecha *datetime* en la cual el Memorando fue confirmado. Es decir, cuando su creador lo promovió de Borrador a Confirmado, quedando a la espera de ser autorizado, gatillando el proceso de envío de la notificación al Emisor de que posee un nuevo Memorando pendiente por autorizar.

3.2.3.7 Fecha de Autorización

Corresponde a la fecha *datetime* en la cual el Memorando fue autorizado por la correspondiente autoridad. Es decir, cuando su emisor lo promovió de Confirmado a Autorizado, volviéndose un Memorando válido y no modificable, gatillando el proceso de envío al destinatario vía correo electrónico, si es que se especificó uno, y a sus declarados como *Con Copia*, si es que fueron especificados.

3.2.3.8 Asunto

Corresponde al asunto o referencia del Memorando. Es obligatorio.

3.2.3.9 Cuerpo Mensaje

Corresponde al cuerpo con el contenido del mensaje. Este no puede ser un mensaje vacío, ni tampoco contener solamente espacios.

3.2.3.10 Documentos Adjuntos (Opcional)

Corresponden a archivos o documentos extras que complementan al Memorando y su contenido principal. Sirve para adherir los documentos extras que suelen requerirse para ciertas solicitudes que requieren ser comunicadas y aprobadas en un Memorando. Esto corresponde, por ejemplo, pero no se limita, a los siguientes documentos

- Certificados
- Resumen Académico
- PDF en general
- DOCS en general

Solamente se aceptan ciertos tipos de formatos, los cuales son los siguientes:

- PDF
- MS-Word
- MS-Power Point
- MS-Excel
- JPEG
- PNG

3.2.3.11 Información de Contacto

Información de contacto asociada al campus de origen señalado en el Memorando. Al seleccionar en el formulario de creación un campus, este busca en la tabla Campus, la información asociada y la añade automáticamente al Memorando, al pie de página.

- Dirección Campus
- Fono Campus
- Email Campus

Esta información debe ser mantenida por el Data Center y coordinada con la información actualizada del Departamento o la Universidad.

3.2.4 Actores o Perfiles de Usuario

El módulo comprende la definición de 4 tipos de actores, quienes llevan a cabo diferentes tareas y poseen diferentes beneficios.

Estos nuevos actores, junto con su definición, para implementarlos efectivamente, conllevan a la creación de nuevos roles de la Intranet. Los roles son la entidad de la Intranet que acumula y encapsula a los permisos, otra entidad de la Intranet, que representa las diferentes acciones y funciones que un usuario en particular puede ejecutar, además de definir a qué información, recursos y archivos tiene acceso.

Como la entidad Memorando es un nuevo recurso de la Intranet, entidad que posee la estructura recién mencionada, para controlar su uso y acceso por parte de los diferentes actores, involucra la creación de nuevos roles, los cuales agrupan nuevos y existentes permisos.

Los 4 grandes tipos de actores que un usuario de la Intranet puede ser para diferentes memos, asociados a diferentes roles que definen su alcance, acciones y acceso a y sobre estos recursos, son los siguientes:

3.2.4.1 Redactores

Usuario de la Intranet con permisos para crear, editar y remitir Memorandos.

Al crear un Memorando, se crea con estatus de *Borrador*. Luego, puede promoverlo a uno *Remitido*. También puede crear plantillas propias a partir de un *Borrador* o un *Remitido*. Al crear un Memorando, se incluye como su creador, indicándose su *UID* del DI en el documento.

Un Redactor solo puede operar bajo Memorandos de su Unidad. Siendo estas las siguientes: Pregrado, Postgrado, Vinculación con el Medio, Investigación, I, T & S.

El rol de Redactor está pensado para los usuarios encargados con la creación y gestión de los Memorandos, principalmente las Secretarías, las cuales poseen su propia cuenta en la Intranet, P.ej. `sec-direccion@inf.utfsm.cl`, el usuario de la Secretaría de dirección, y `sec-pregado@inf.utfsm.cl`, usuario de la Secretaría de pregrado.

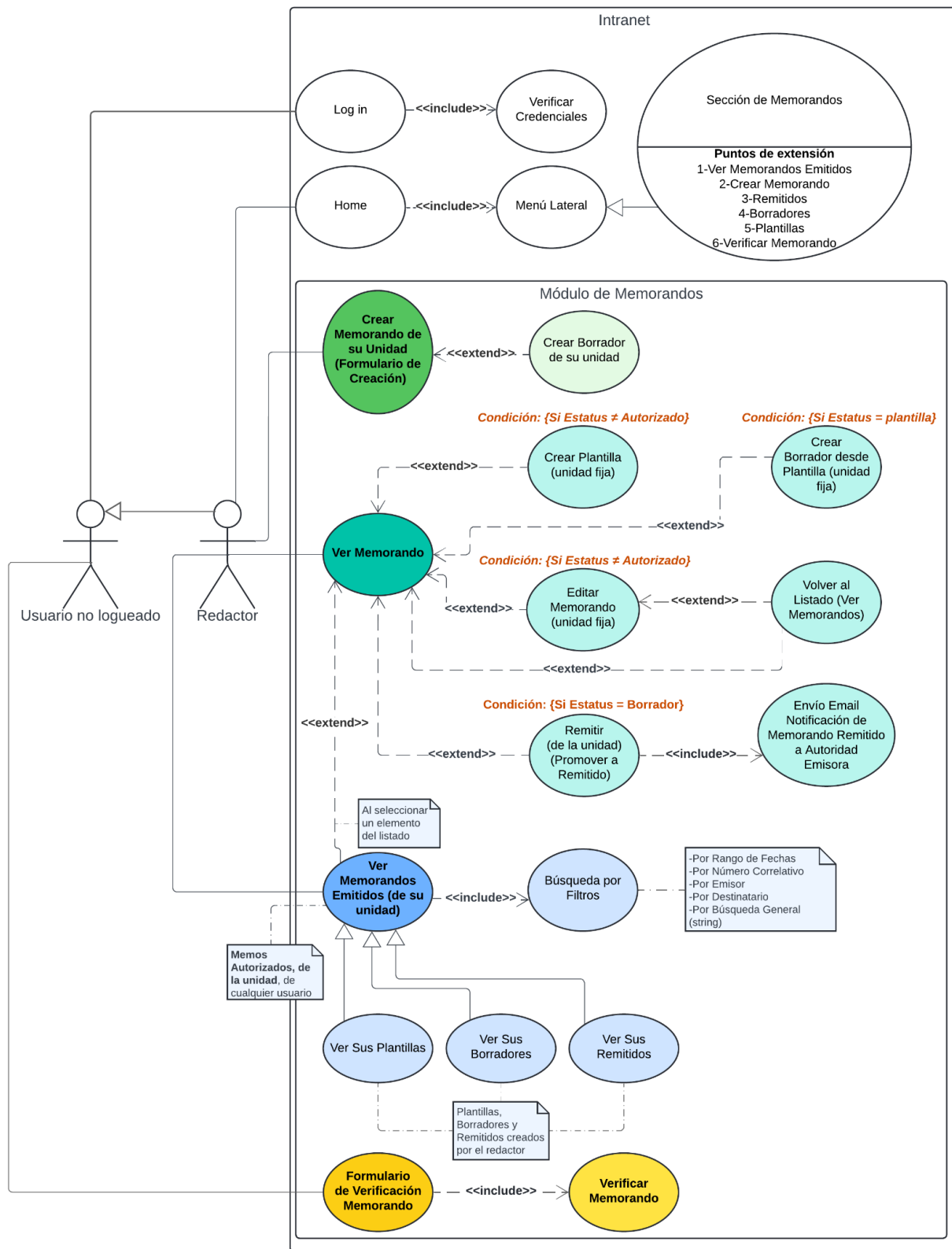


Ilustración 3-2 Diagrama de Casos de Uso de Redactor

3.2.4.2 Redactor de Dirección

Un Redactor perteneciente a la Unidad de Dirección, que goza de permisos sin restricción de Unidad. Puede crear, editar y remitir Memorandos de cualquier Unidad. También, puede ver los Memorandos emitidos por todas las unidades. P.ej. la Secretaría de Dirección, sec-direccion@inf.utfsm.cl, el Director, y el encargado de Gestión Docente, gestion@inf.utfsm.cl, entre otros.

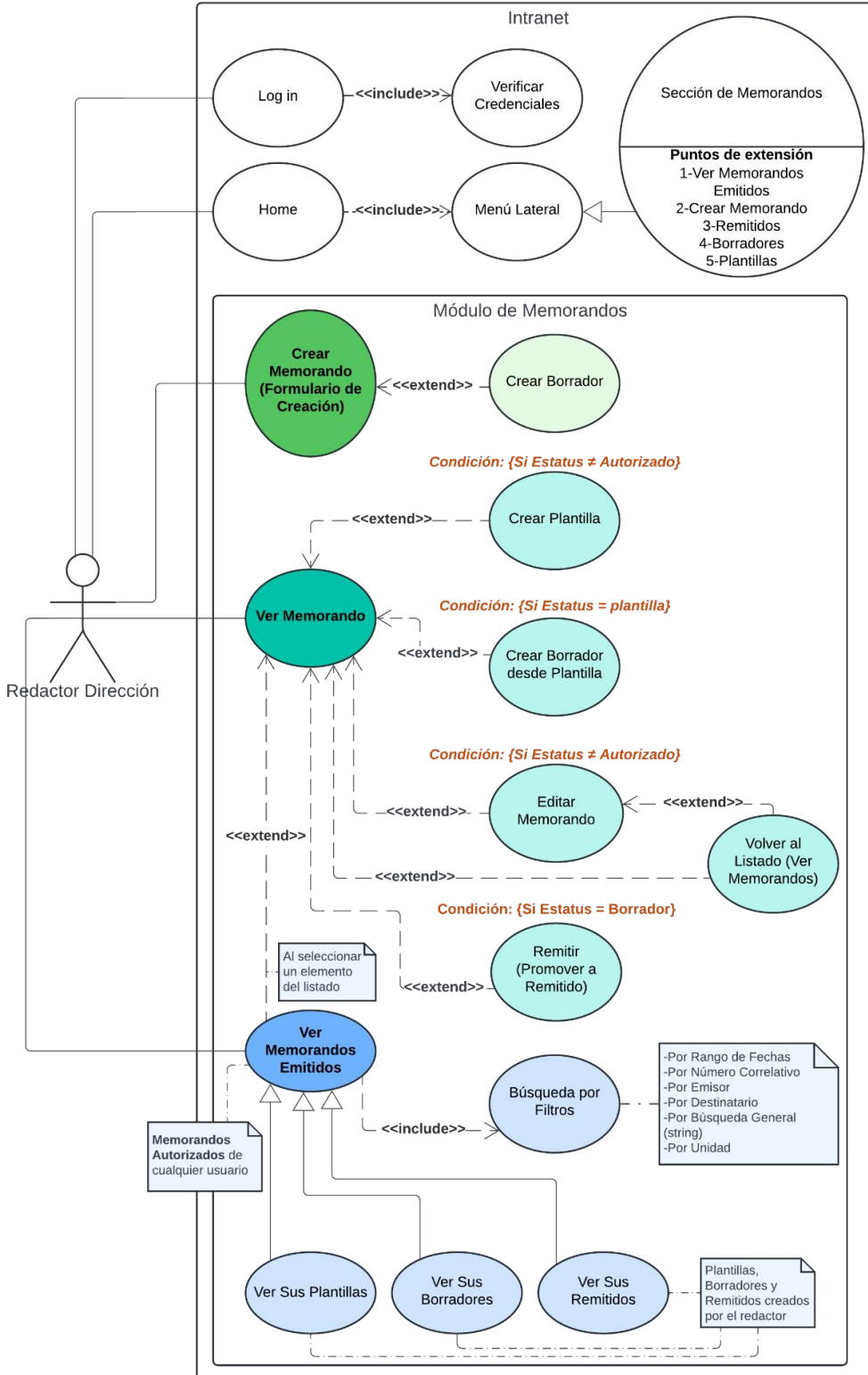


Ilustración 3-3 Diagrama de Casos de Uso de Redactor de Dirección

3.2.4.3 Emisor

Un Emisor corresponde a un usuario con cuenta en la Intranet. Es un rol de mayor rango que el Redactor, hereda los permisos de este último y además puede autorizar los Memorandos en los que aparece como emisor.

Autorizar un Memorando corresponde a la capacidad de promoverlo de *Remitido* a *Autorizado*, convirtiéndolo en un Memorando válido y no modificable, gatillando el envío al destinatario, si es que se especificó su correo electrónico, y a los Con Copia (CC), si es que fueron especificados. También, en la autorización, se calcula y asigna un código verificador único al Memorando, el cual se muestra en él.

El código se puede usar por alguna parte interesada para verificar su veracidad y validez. Esta verificación se lleva a cabo en la misma Intranet, en una ruta pública, que se señala en el Memorando.

Un Emisor solo puede serlo de un Memorando de la unidad a la cual pertenece. Estas corresponden a: Pregrado, Postgrado, Vinculación con el Medio, Investigación, I, T & S.

Este actor representa a todo miembro del DI que puede emitir Memorandos. En general, está pensado para las autoridades, como el Director, los Subdirectores y los Jefes, pero puede ser otorgado a cualquier otro usuario que lo requiera, p.ej. los profesores en Pregrado.

Estos permisos deben ser mantenidos por el administrador del Data Center de I, T & S.

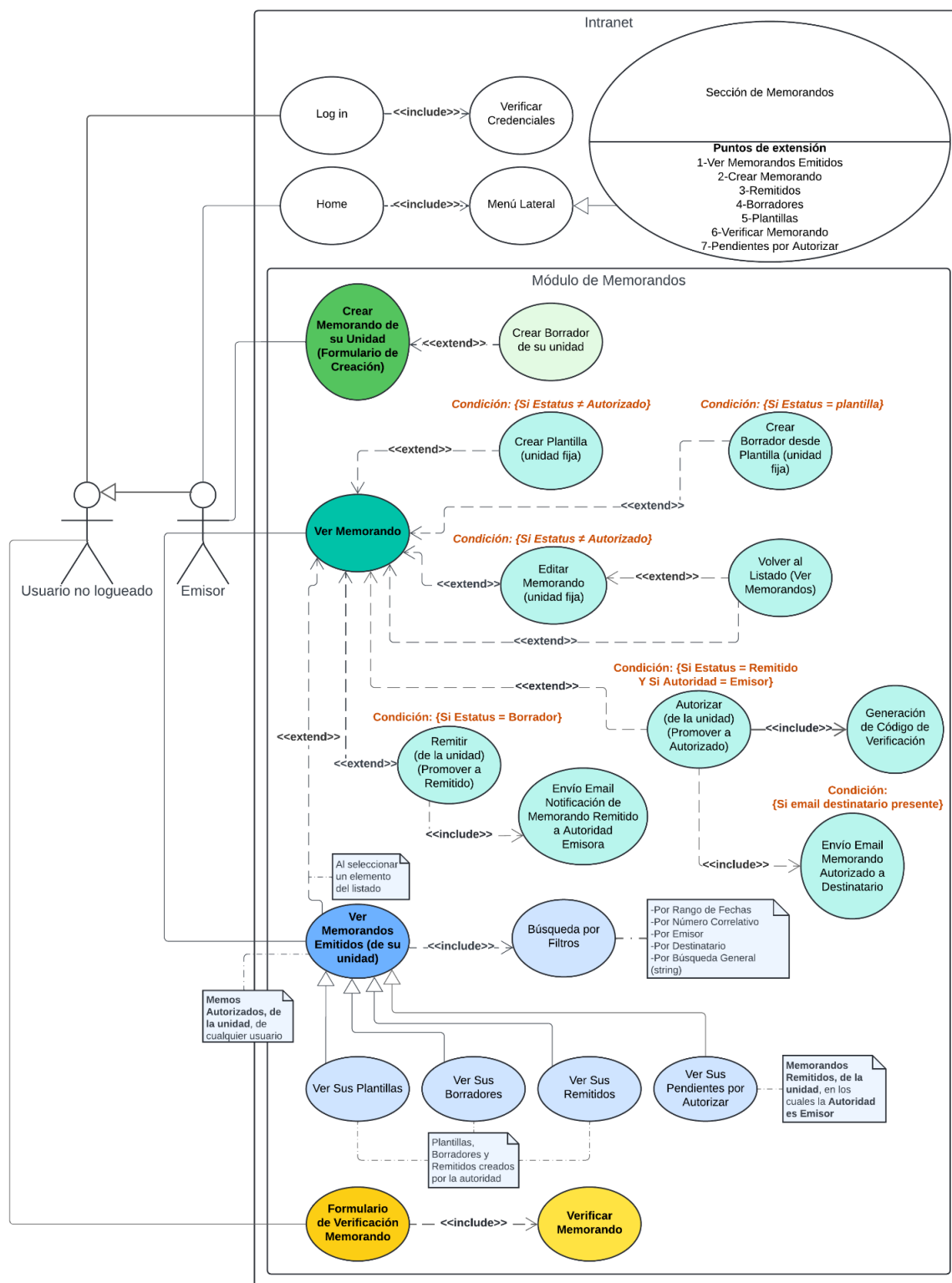


Ilustración 3-4 Diagrama de Casos de Uso de Emisor

3.2.4.4 Emisor de Dirección

Corresponde a un Emisor perteneciente a la Unidad de Dirección, goza de permisos no restringidos por unidad. Puede crear, editar, remitir y autorizar Memorandos de cualquier unidad. También, puede ver los Memorandos emitidos por todas las unidades. P.ej. el Director del DI.

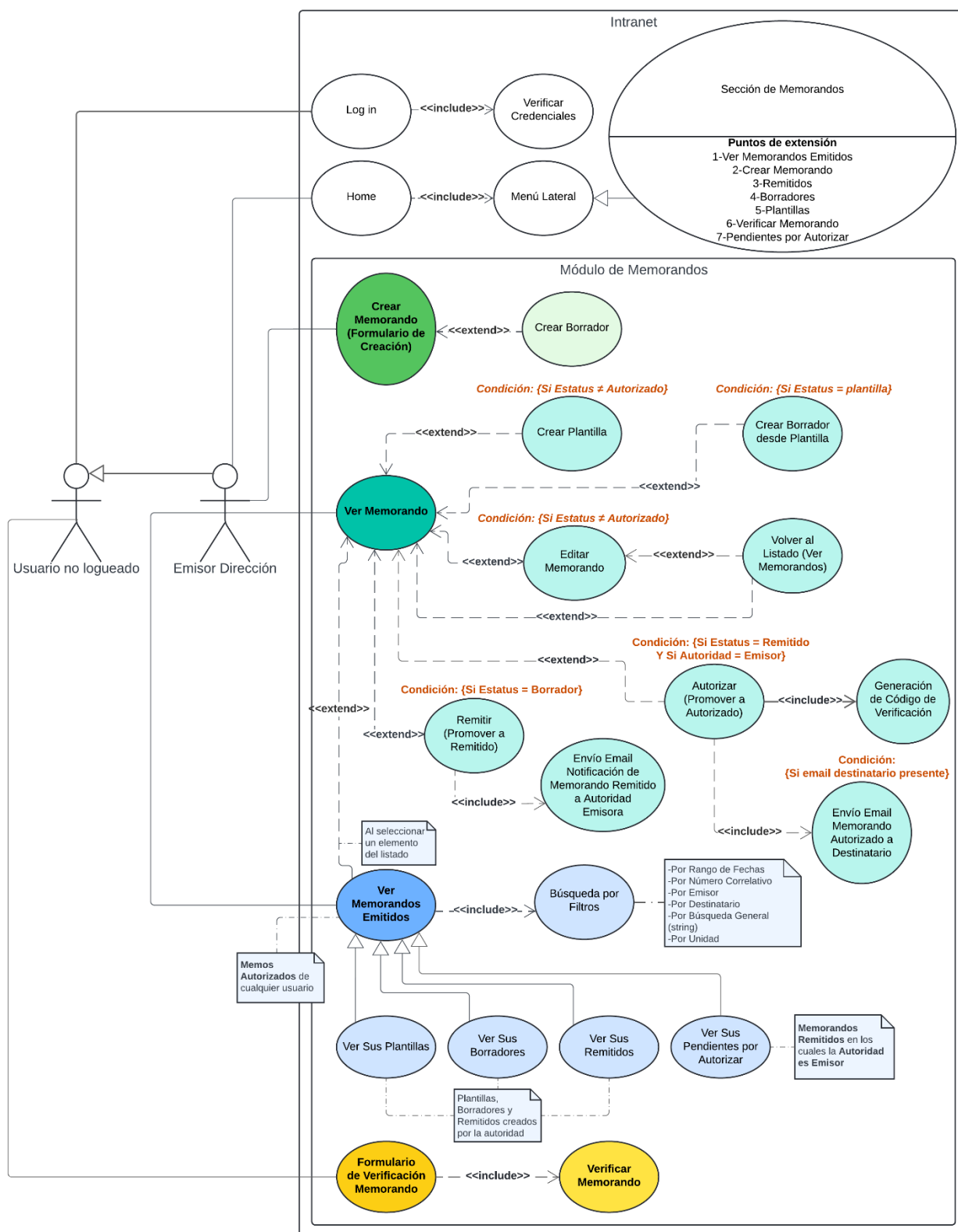


Ilustración 3-5 Diagrama de Casos de Uso de Emisor Dirección

3.2.4.5 Mantenedores y Administradores

Son los usuarios encargados de mantener el proyecto de la Intranet y su infraestructura. Este grupo está compuesto por el jefe de I, T & S, y los desarrolladores o ayudantes que estén involucrados en la mantención de la Intranet.

Se caracteriza por tener acceso completo o de administrador. Posee los permisos de un Emisor de Dirección, más la capacidad de eliminar Memorandos no autorizados y ver a los contactos (*MemoContacto*) de todos los usuarios.

3.2.4.6 Huésped

Cualquier persona o usuario que requiera verificar la existencia y veracidad de un Memorando. Esto lo logra ingresando al Formulario de Verificación de Memorandos de la Intranet. Una vez dentro, ingresa el número correlativo, la fecha de emisión y el código de verificación del Memorando por verificar.

Puede poseer o no una cuenta en la Intranet. Si posee cuenta, además de verificar un Memorando, puede también potencialmente iniciar sesión, pero no es necesario para la verificación.

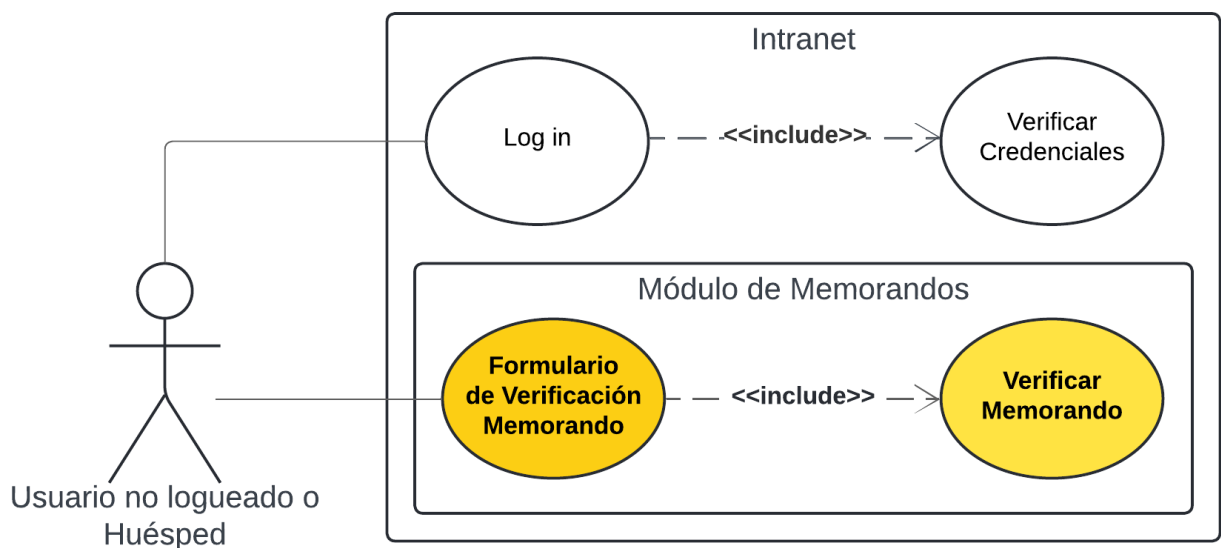


Ilustración 3-6 Diagrama de Casos de Uso de Huésped

3.2.5 Contactos (Memo Contacto)

Todos los actores pueden gestionar (CRUD) una lista propia de contactos. Estos se pueden ocupar para autocompletar los datos de un emisor o destinatario en el formulario de creación/edición de un Memorando. Se pueden ver en la pestaña *Mis Contactos* en el módulo.

Administrador tiene acceso y puede manipular los contactos de todos los usuarios.

A continuación, se detallan los campos que componen un Contacto.

3.2.5.1 Nombre Completo

Nombre completo; puede considerar cualquier combinación: nombre y apellido, ambos nombres y ambos apellidos, solo el nombre, etc. Lo importante es que no esté vacío.

3.2.5.2 Cargo

Corresponde al cargo ocupado por el contacto.

3.2.5.3 Departamento

Corresponde al Departamento al que pertenece el contacto. Para ocupar el contacto como emisor, el valor de este campo debe ser *Departamento de Informática* textual.

3.2.5.4 UID (User ID)

UID del usuario. Para ocupar el contacto como emisor, se debe especificar este campo con un UID válido y existente; de lo contrario, no se encontrará disponible.

3.2.5.5 Uso de los Contactos

Los Contactos creados, se encuentran disponibles al crear un nuevo Memorando, para autocompletar los datos de los destinatarios o de los emisores.

En el formulario de creación, arriba de los campos de emisores y destinatarios aparece un *Select*, el cual permite buscar Contactos que coincidan con los caracteres ingresados.

Si se selecciona uno de los resultados obtenidos, entonces se autocompletarán, con los datos del Contacto elegido, los campos de emisor o destinatario que estén debajo de aquel *Select*.

3.2.5.6 Permisos de Actores Sobre los Contactos

Los Actores no administradores, pueden gestionar (CRUD) solo a sus propios de Contactos. El Administrador puede gestionar a los Contactos de todos los usuarios.

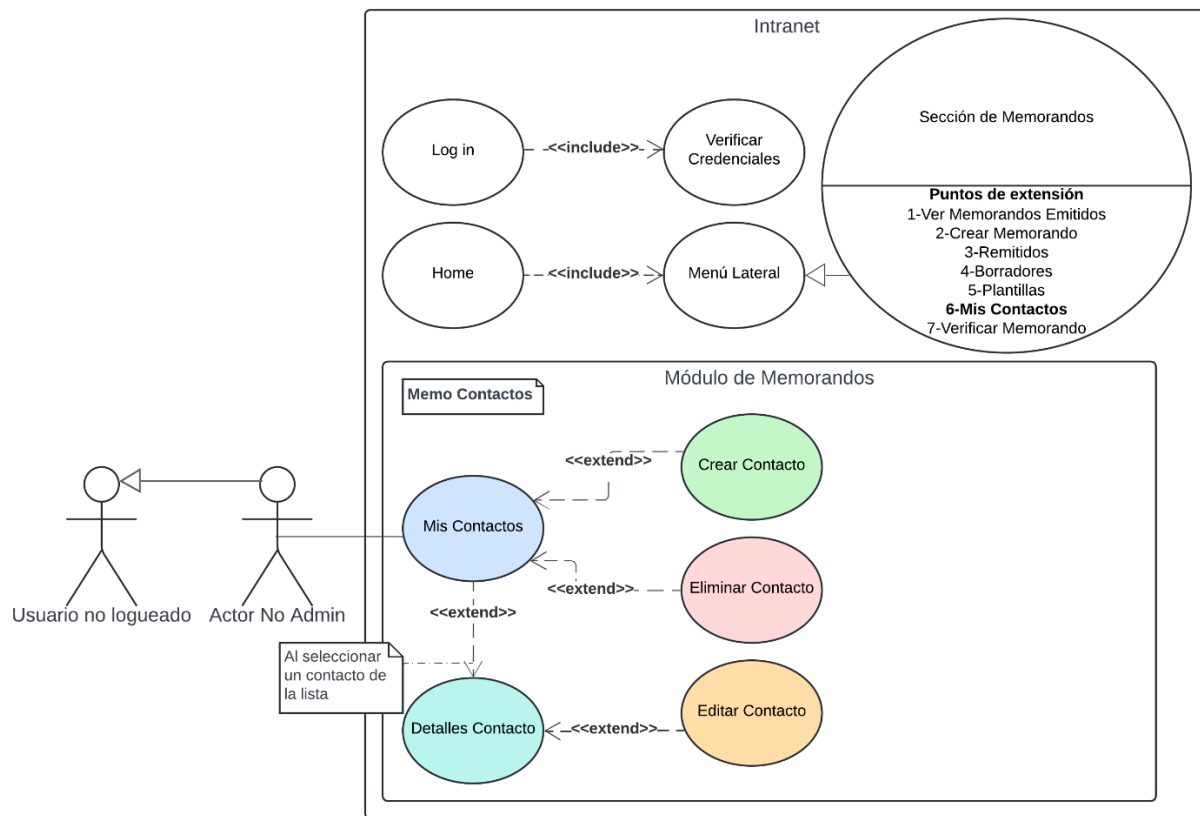


Ilustración 3-7 Diagrama de Casos de Uso de Contactos de Actor No Admin

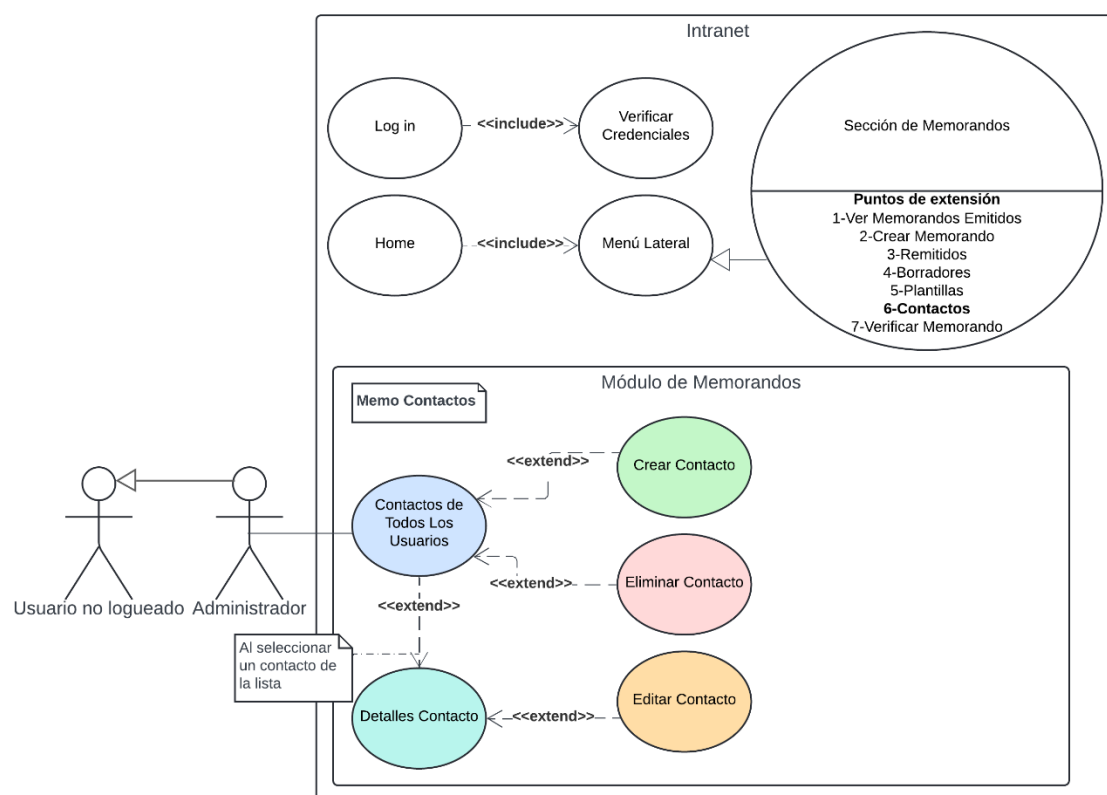


Ilustración 3-8 Diagrama de Casos de Uso de Contactos de Administrador

3.3 Implementación

En la página de inicio de la Intranet, a aquellos con permisos de Redactor o Emisor, les aparece en el Menú Lateral la sección “Memorandos”, donde se encontrarán todas las funciones asociadas al módulo a las que tengan acceso.

3.3.1 Contactos

Los Contactos

3.3.1.1 Crear y Editar Contacto

3.3.1.1.1 Caso de Uso – Crear Contacto

1. **Actores:** Redactor, Redactor de Dirección, Emisor, Emisor de Dirección. Se usará *Usuario* genéricamente.
2. **Precondiciones:** el Redactor se encuentra autenticado en la Intranet.
3. **Postcondiciones:** el Contacto es creado y registrado en la base de datos correctamente.
4. **Referencias Funcionales:** RF-7.x, RF-8.x.x ([Ver Requerimientos](#))

Flujo Principal

Tabla 3-2 Caso de Uso: Crear y Editar Contacto, Fuente: Elaboración Propia

Paso	Usuario	Sistema
1	Usuario selecciona sección Memorandos del Menú Lateral de la Intranet	Sistema despliega sección Memorandos del Menú Lateral con opciones de asociadas.
2	Usuario selecciona opción <i>Mis Contactos</i>	Sistema redirige Usuario al Listado de sus Contactos.
3	Usuario presiona el botón <i>Crear Contacto</i> presente en esta vista	Sistema redirige Usuario a la Vista con formulario o Interfaz de Creación.
4	Usuario completa obligatoriamente el Nombre Completo	
5	Usuario completa opcionalmente Cargo	
6	Usuario completa opcionalmente Dpto. Si se pretende usar el Contacto para autocompletar un Emisor se debe completar este campo con el valor literal <i>Departamento de Informática</i>	
7	Usuario completa opcionalmente Correo electrónico	
8	Usuario completa opcionalmente UID. Si se pretende usar el Contacto para autocompletar un Emisor, se debe completar este campo con el UID de informática correspondiente.	
9	Usuario envía el formulario	Sistema valida el formulario exitosamente, crea el Contacto y lo almacena en Base de Datos
10		Sistema redirige usuario a Viste Detalles del nuevo Contacto

Caminos Alternativos

Camino Error A: Sistema detecta errores en el formulario

Paso	Usuario	Sistema
9a	Usuario envía formulario	Sistema detecta errores en formulario (p.ej. datos obligatorios no ingresados o con formato incorrecto).
10a		Sistema presenta notificación con errores detectados y solicita al Usuario que los corrija.

11a	Usuario corrige los errores notificados	
12a	Se vuelve a Flujo Principal en paso 9	

3.3.1.1.2 Caso de Uso – Editar Contacto

Flujo Principal

1. **Actores:** Redactor, Redactor de Dirección, Emisor, Emisor de Dirección. Se usará *Usuario* genéricamente.
2. **Precondiciones:** el Redactor se encuentra autenticado en la Intranet.
3. **Postcondiciones:** el Contacto es creado y registrado en la base de datos correctamente.
4. **Referencias Funcionales:** RF-7.x, RF-8.x.x ([Ver Requerimientos](#)).

Tabla 3-3 Caso de Uso: Editar Contacto, Fuente: Elaboración Propia

Paso	Usuario	Sistema
1	Usuario selecciona sección Memorandos del Menú Lateral de la Intranet	Sistema despliega sección Memorandos del Menú Lateral con opciones de asociadas.
2	Usuario selecciona opción <i>Mis Contactos</i>	Sistema redirige Usuario al Listado de sus Contactos.
3	Usuario presiona botón <i>Editar</i> de un elemento de Contacto del Listado	Sistema redirige Usuario a la Vista con formulario o Interfaz de Creación.
4		Sistema completa todos los campos del formulario con los datos del Contacto específico
5	Usuario evalúa editar Contacto	
6	Se siguen los mismos pasos que en <i>Crear Contacto</i>	

Caminos Alternativos

Camino Alternativo A: Ruta a Editar desde Vista Detalles

Paso	Usuario	Sistema
3a	Usuario presiona botón Ver Detalles de un Contacto particular	Sistema redirige a Vista Detalles de Contacto
4a	Usuario presiona botón <i>Editar Contacto</i> presente en la vista.	Sistema redirige a Vista con formulario o Interfaz de Creación del Contacto específico

5a	Se vuelve a Flujo Principal en paso 4	
-----------	--	--

3.3.1.2 Caso de Uso - Ver y Listar Contacto

5. **Actores:** Redactor, Redactor de Dirección, Emisor, Emisor de Dirección. Se usará *Usuario* genéricamente.
6. **Precondiciones:** el Redactor se encuentra autenticado en la Intranet y en la Interfaz de creación.
7. **Postcondiciones:** el Contacto es creado y registrado en la base de datos correctamente.
8. **Referencias Funcionales:** RF-7.x ([Ver Requerimientos](#))

Tabla 3-4 Caso de Uso Ver y Listar Contacto, Fuente: Elaboración Propia

Paso	Usuario	Sistema
1	Usuario selecciona sección Memorandos del Menú Lateral de la Intranet	Sistema despliega sección Memorandos del Menú Lateral con opciones de asociadas
2	Usuario selecciona opción <i>Mis Contactos</i>	Sistema redirige Usuario al Listado de sus Contactos
3	Si posee Contactos, Usuario selecciona Vista Detalles de un Contacto. Si no posee Contactos, desde esta vista puede presionar <i>Crear Contacto</i> , para ir a la interfaz de creación	Sistema redirige Usuario a Vista Detalles del Contacto seleccionado
4	Usuario revisa Contacto con la opción potencial de Editarlo al presionar <i>Editar Memorando</i> , disponible en esta vista	

3.3.1.3 Eliminar Contacto

9. **Actores:** Redactor, Redactor de Dirección, Emisor, Emisor de Dirección. Se usará *Usuario* genéricamente.
10. **Precondiciones:** el Redactor se encuentra autenticado en la Intranet y en la Interfaz de creación. Existe al menos un Contacto.

11. **Postcondiciones:** el Contacto es creado y registrado en la base de datos correctamente.
12. **Referencias Funcionales:** RF-7.x.x ([Ver Requerimientos](#))

Tabla 3-5 Caso de Uso: Eliminar Contacto, Fuente: Elaboración Propia

Paso	Usuario	Sistema
1	Usuario selecciona sección Memorandos del Menú Lateral de la Intranet	Sistema despliega sección Memorandos del Menú Lateral con opciones de asociadas
2	Usuario selecciona opción <i>Mis Contactos</i>	Sistema redirige Usuario al Listado de sus Contactos
3	Usuario presiona botón <i>Eliminar Contacto</i> , al final de un elemento de Contacto del Listado	Sistema despliega una caja de diálogo solicitando a Usuario confirmación de la eliminación
4	Usuario confirma la eliminación	Sistema Elimina al Contacto de Base de Datos y notifica mensaje de éxito de la acción al Usuario.
5		Sistema reactiva Vista Listado de Contactos

Caminos Alternativos

Camino Alternativo A: Usuario cancela la confirmación de la eliminación.

3a	Usuario presiona botón <i>Eliminar Contacto</i> , al final de un elemento de Contacto del Listado	Sistema despliega una caja de diálogo solicitando a Usuario confirmación de la eliminación.
4a	Usuario no confirma cancelando la operación	Sistema cierra diálogo.
5a	Se vuelve a Flujo Principal en paso 5	

3.3.2 Creación y Edición

Los Memorandos se pueden crear al seleccionar la opción *Crear Memorando* del menú lateral.

En esta vista se presenta un formulario. Los campos a completar son:

Los destinatarios: se pueden agregar hasta 3, con el primero siendo el principal. Este último será referido en la cabecera del Memorando, en el campo A:, junto con su cargo y Dpto. Los otros

destinatarios, si se agregaron, se señalarán como CC (Con Copia) y solo incluirán su nombre y correo electrónico.

Continuando con el formulario, se debe especificar el asunto del Memorando, la unidad y el campus a los que el emisor pertenece. Luego, se pueden opcionalmente agregar archivos adjuntos.

Después de este formulario, se presenta un área de texto enriquecido para redactar el mensaje.

Una vez todo completado y redactado, presionar el botón “Crear Memorando” para enviar los datos al servidor, el que guardará un Memorando en base de datos.

Si el proceso es exitoso, se crea un Memorando con estatus *Borrador*. Memorandos con tal estatus, poseen fecha de creación, no poseen número correlativo y son solo locales para el Redactor.

3.3.2.1 Casos de Uso – Creación Memorando

3.3.2.1.1 Nombre: Creación de Memorando (*Borrador*)

13. **Actores:** Redactor, Redactor de Dirección, Emisor, Emisor de Dirección. Se usará *Usuario* genéricamente.
14. **Precondiciones:** el Redactor se encuentra autenticado en la Intranet y en la Interfaz de creación.
15. **Postcondiciones:** el Memorando es creado y registrado en la base de datos correctamente.
16. **Referencias Funcionales:** RF-1.x.x, RF-2 ([Ver Requerimientos](#))

Flujo Principal

Tabla 3-6 Flujo Principal Caso de Uso: Crear Memorando

Paso	Usuario	Sistema
1	Usuario busca Contacto ingresando caracteres en <i>Select</i> de Destinatario Principal	Devuelve Lista Contactos que coincidan parcialmente
2	Usuario selecciona un Contacto de la lista.	Autocompleta los campos Nombre Completo, Cargo, Dpto. y opcionalmente mail del Destinatario Principal con el Contacto seleccionado.
3	Usuario puede agregar hasta 2 Destinatarios secundarios extras, los <i>Con Copia</i> , opcionalmente .	

	Usuario evalúa agregar más Destinatarios	
	[Comienza Bucle]	Destinatarios Secundarios
3.1.1	Condición: {[Usuario quiere agregar otro Destinatario] Y [Número Actual de Destinatarios <= 3]} Usuario presiona botón <i>agregar Destinatario</i> .	Sistema instancia un nuevo grupo de campos para especificar Destinatario : - Campo de Texto para Nombre Completo - Campo de Texto para Cargo - Campo de Texto para Dpto. - Campo de Texto para Correo electrónico - <i>Select</i> para buscar Contactos con los cuales autocompletar los campos anteriores.
3.1.2	Usuario repite los pasos del 1 al 2 para el nuevo Destinatario secundario.	
3.2	Condición: {Else} Usuario procede a paso 4. Saliendo del Bucle	
	[Termina Bucle]	Destinatarios Secundarios
4	Usuario busca Contacto ingresando caracteres en <i>Select</i> de Emisor Principal	Devuelve Lista Contactos que coincidan parcialmente
5	Usuario selecciona un Contacto de la lista.	Autocompleta los campos Nombre Completo, Cargo, Dpto. y opcionalmente mail del del Emisor Principal con el Contacto seleccionado
6	Usuario puede agregar hasta 2 Emisores secundarios extras opcionalmente. Usuario evalúa agregar más Emisores	

	[Comienza Bucle]	Emisores Secundarios
6.1.1	Condición: {[Usuario quiere agregar otro Emisor] Y [Número Actual de Emisores <= 3]} Usuario presiona botón <i>agregar Emisor</i> .	Sistema instancia un nuevo grupo de campos para especificar Emisor : - Campo de Texto para Nombre Completo - Campo de Texto para Cargo - Campo de Texto para Dpto. - Campo de Texto para Correo electrónico - Campo de Texto para UID de Informática. - <i>Select</i> para buscar Contactos con los cuales autocompletar los campos anteriores.
6.1.2	Usuario repite los pasos del 4 al 5 para el nuevo Emisor secundario.	
6.2	Condición: {Else} Usuario procede a paso 7. Saliendo del Bucle	
	[Termina Bucle]	Emisores Secundarios
7		Si Usuario es de Dirección. Habilita todas las Unidades del DI para ser seleccionadas. En caso contrario, solo habilita la Unidad del Usuario.
8	Usuario selecciona Unidad.	
9	Usuario selecciona Campus.	
10	Usuario puede agregar archivos adjuntos opcionalmente. Usuario evalúa agregar archivos adjuntos.	
	[Comienza Bucle]	Archivos Adjuntos
10.1	Condición: [Usuario quiere agregar archivos adjuntos]	Sistema despliega explorador de archivos, solicitando a Usuario que seleccione los archivos a adjuntar.

	Usuario presiona el botón del campo de archivos (<i>file field</i>)	
10.2	Usuario selecciona 1 o más archivos a adjuntar.	Sistema guarda en caché temporalmente los archivos adjuntos. Sistema actualiza lista visual de archivos adjuntados.
10.3	Condición: [Usuario no quiere agregar más archivos] Usuario procede a paso 11, saliendo del bucle.	
	[Termina Bucle]	Archivos Adjuntos
11	El Usuario redacta el cuerpo del mensaje en área de texto enriquecido	Sistema ofrece funcionalidad de librería <i>Trix</i>
12	Una vez el Usuario está listo, presiona el botón para <i>Crear Memorando</i>	El sistema crea Memorando con los datos especificados. Añade automáticamente fecha de creación y estatus de <i>borrador</i> . Almacena los archivos adjuntos (Active Storage) si se especificaron. Guarda Memorando en base de datos. Redirige Usuario a la vista en detalle del Memorando con mensaje de éxito.

Camino Alternativos

Camino Alternativo A: Usuario completa los campos de Destinatario manualmente

Tabla 3-7 Caminos Alternativos Caso de Uso: Crear Memorando

Paso	Usuario	Sistema
------	---------	---------

1a	Usuario completa campos Nombre Completo, Cargo, Dpto. de Destinatario Principal (o secundario) manualmente	
3a	Usuario puede agregar hasta 2 Destinatarios secundarios extras, los <i>Con Copia</i>, opcionalmente. Usuario evalúa agregar más Destinatarios	
	[Comienza Bucle]	Destinatarios Secundarios
3.1.1a	Condición: {[Usuario quiere agregar otro destinatario] Y [Número Actual de Destinatarios <= 3]} Usuario presiona botón <i>agregar Destinatario</i>.	Sistema instancia un nuevo grupo de campos para especificar Destinatario: - Campo de Texto para Nombre Completo - Campo de Texto para Cargo - Campo de Texto para Departamento. - Campo de Texto para Correo electrónico - <i>Select</i> para buscar Contactos con los cuales autocompletar los campos anteriores.
3.1.2a	Usuario repite el paso 1a para el nuevo Destinatario secundario.	
3.2a	Condición: {Else} Usuario procede a paso 4a. Saliendo del Bucle	
	[Termina Bucle]	Destinatarios Secundarios
4a	Se vuelve a Flujo Principal en paso 4	

Camino Alternativo B: El Usuario completa los campos de Emisor manualmente

Paso	Usuario	Sistema
5b	Usuario puede agregar hasta 2 Emisores secundarios extras, los <i>Con Copia</i>, opcionalmente.	

	Usuario evalúa agregar más Emisores	
6b	[Comienza Bucle]	Emisores Secundarios
6.1.1b	Condición: {[Usuario quiere agregar otro Emisor] Y [Número Actual de Emisores <= 3]} Usuario presiona botón <i>agregar Emisor</i> .	Sistema instancia un nuevo grupo de campos para especificar Emisor : - Campo de Texto para Nombre Completo - Campo de Texto para Cargo - Campo de Texto para Dpto. - Campo de Texto para Correo electrónico - <i>Select</i> para buscar Contactos con los cuales autocompletar los campos anteriores.
6.1.2b	Usuario repite el paso 1a para el nuevo Emisor secundario.	
6.2b	Condición: {Else} Usuario procede a paso 7a. Saliendo del Bucle	
	[Termina Bucle]	Emisores Secundarios
7b	Se vuelve a Flujo Principal en paso 7	

Camino Alternativo C: Usuario elimina Destinatario secundario extra

	[Comienza Bucle]	Eliminación Destinatarios Secundarios
3.3c	Condición: {[Existen Destinatarios secundarios] Y [Usuario quiere eliminar un Destinatario secundario] Y [Destinatario no es el Principal]} Usuario presiona botón <i>eliminar Destinatario</i> asociado a un grupo de campos de Destinatario .	Sistema elimina el grupo de campos asociados al Destinatario secundario en cuestión. Elimina los campos: - Texto para Nombre Completo - Texto para Cargo - Texto para Departamento. - Texto para Correo electrónico - <i>Select</i> para buscar Contactos. Asociados al Destinatario secundario.

3.4c	Condición: {Else} Usuario procede a paso 4c. Saliendo del Bucle.	
	[Termina Bucle]	Eliminación Destinatarios Secundarios
4c	Se vuelve a Flujo Principal en paso 4	

Camino Alternativo D: Usuario elimina Emisor secundario extra.

	[Comienza Bucle]	Eliminar Emisores Secundarios
6.3d	Condición: {[Existen Emisores secundarios] Y [Usuario quiere eliminar un Emisor secundario] Y [Emisor no es el Principal]] Usuario presiona botón <i>eliminar Emisor</i> asociado a un grupo de campos de Emisor .	Sistema elimina el grupo de campos asociados al Emisor secundario en cuestión. Elimina los campos: - Texto para Nombre Completo - Texto para Cargo - Texto para Departamento. - Texto para Correo electrónico - Texto para UID de Informática - <i>Select</i> para buscar Contactos. Asociados al Emisor secundario.
6.4d	Condición: {Else} Usuario procede a paso 7d. Saliendo del Bucle	
	[Termina Bucle]	Eliminar Emisores Secundarios
7d	Se vuelve a Flujo Principal en paso 7	

Camino Alternativo E: El Usuario elimina archivos adjuntos.

	[Comienza Bucle]	Eliminar Archivo adjunto
10.4e	Condición: {[Se han agregado archivos adjuntos] Y [Usuario quiere eliminar uno]} Usuario presiona el botón <i>eliminar archivo adjunto</i> al costado del archivo adjunto a eliminar	Sistema elimina archivo adjunto de caché y de la lista visual.
10.5e	Condición: [Usuario no quiere eliminar más archivos adjuntos] Usuario procede a Paso 11, saliendo del bucle	
	[Termina Bucle]	Eliminar Archivo adjunto
11e	Se vuelve a Flujo Principal en paso 11	

Camino Alternativo F: El Usuario limpia lista de archivos adjuntos

10.7f	Condición: [Se han agregado archivos adjuntos] Y [Usuario quiere limpiar lista] Usuario presiona el botón del campo de <i>limpiar archivos adjuntos</i> al costado del campo de archivos (<i>file field</i>)	Sistema limpia caché temporal de archivos adjuntos y reinicia la lista visual.
11f	Se vuelve a Flujo Principal en paso 11	

Camino de Error G: Campos obligatorios de formulario no especificados

12g	Una vez el Usuario está listo, presiona el botón para <i>Crear Memorando</i>	Sistema detecta errores en formulario (datos obligatorios no ingresados o con formato incorrecto)
12.1g		Sistema presenta notificación con errores detectados y solicita al Usuario que los corrija
12.2g	Usuario corrige errores notificados	

12.3g	Se vuelve a Flujo Principal en paso 12	
--------------	---	--

3.3.2.2 Caso de uso – Editar Memorando

En el caso de Editar Memorando existente, se presenta el mismo formulario y se siguen los mismos pasos, con la excepción:

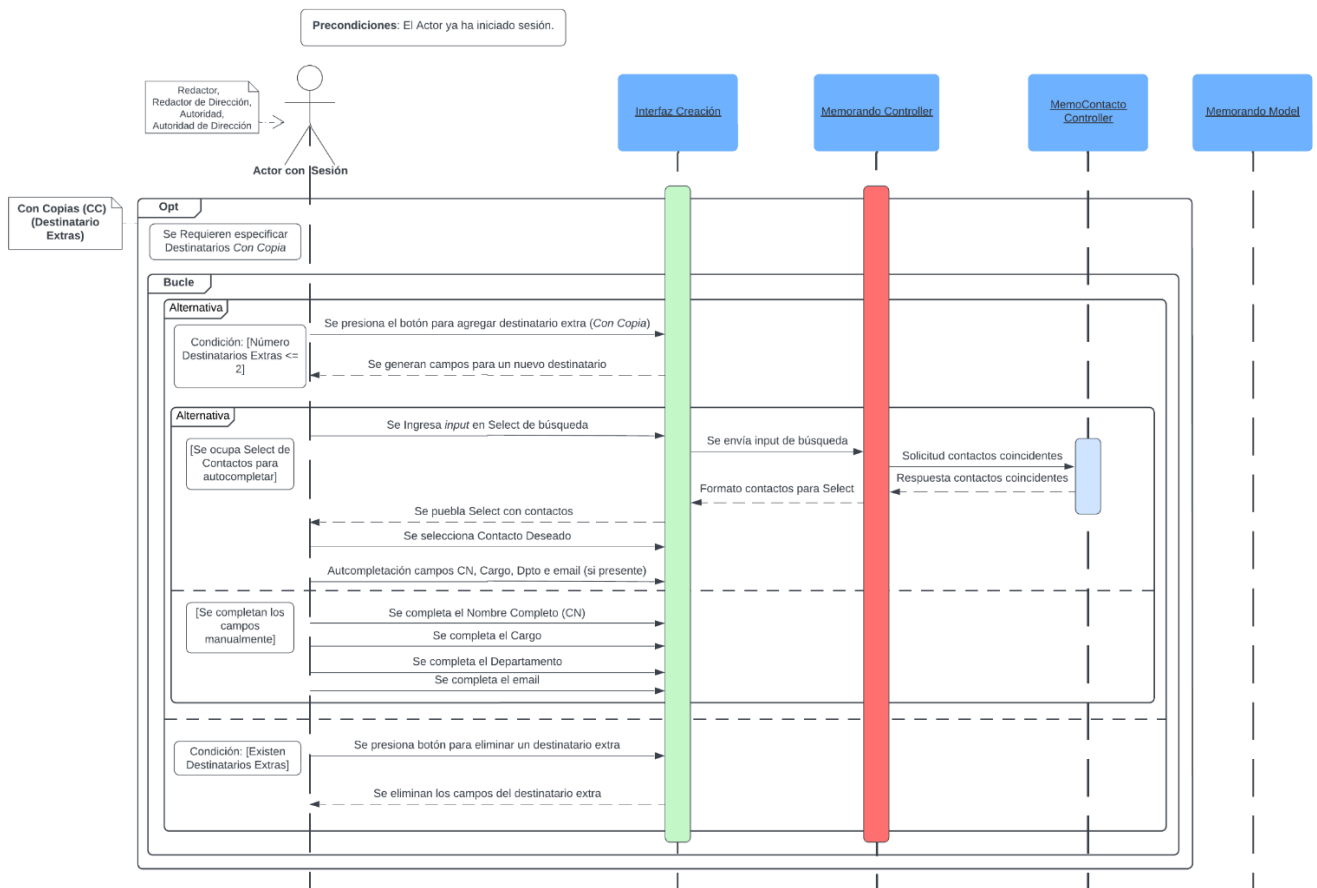
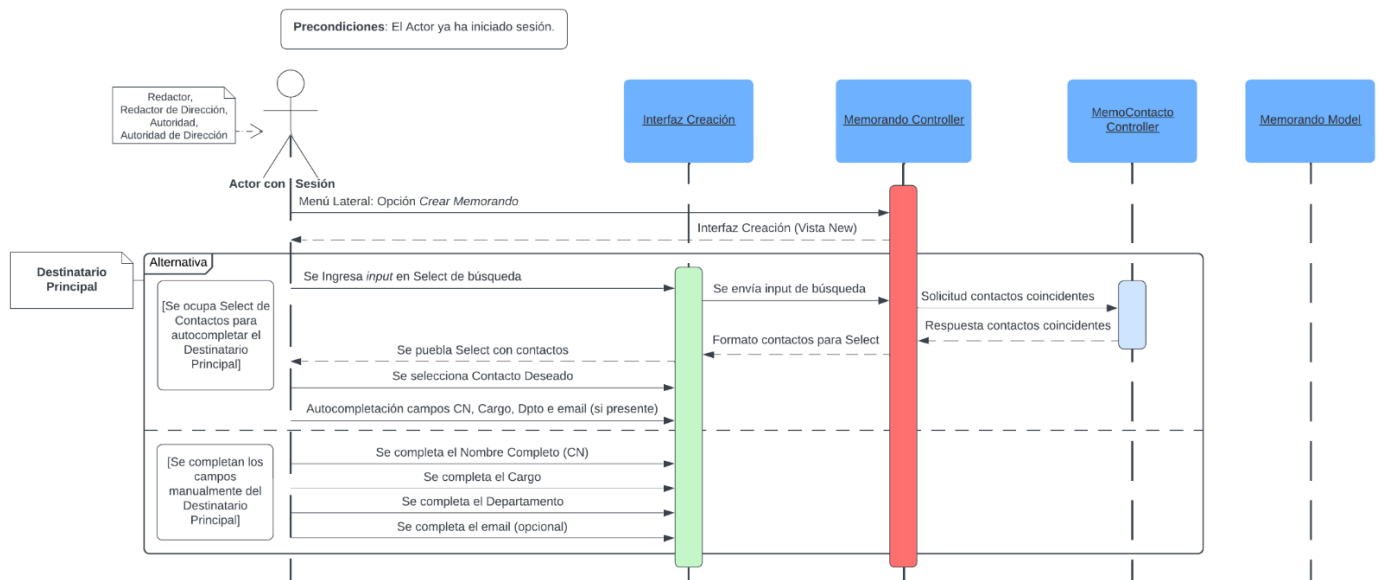
1. **Actores:** Redactor, Redactor de Dirección, Emisor, Emisor de Dirección, Administrador. Se usará *Usuario* genéricamente.
2. **Precondiciones:** el Usuario se encuentra autenticado en la Intranet.
3. **Postcondiciones:** Las modificaciones del Memorando se registran en la base de datos exitosamente.
4. **Referencias Funcionales:** RF-1.x.x, RF-2 ([Ver Requerimientos](#)).

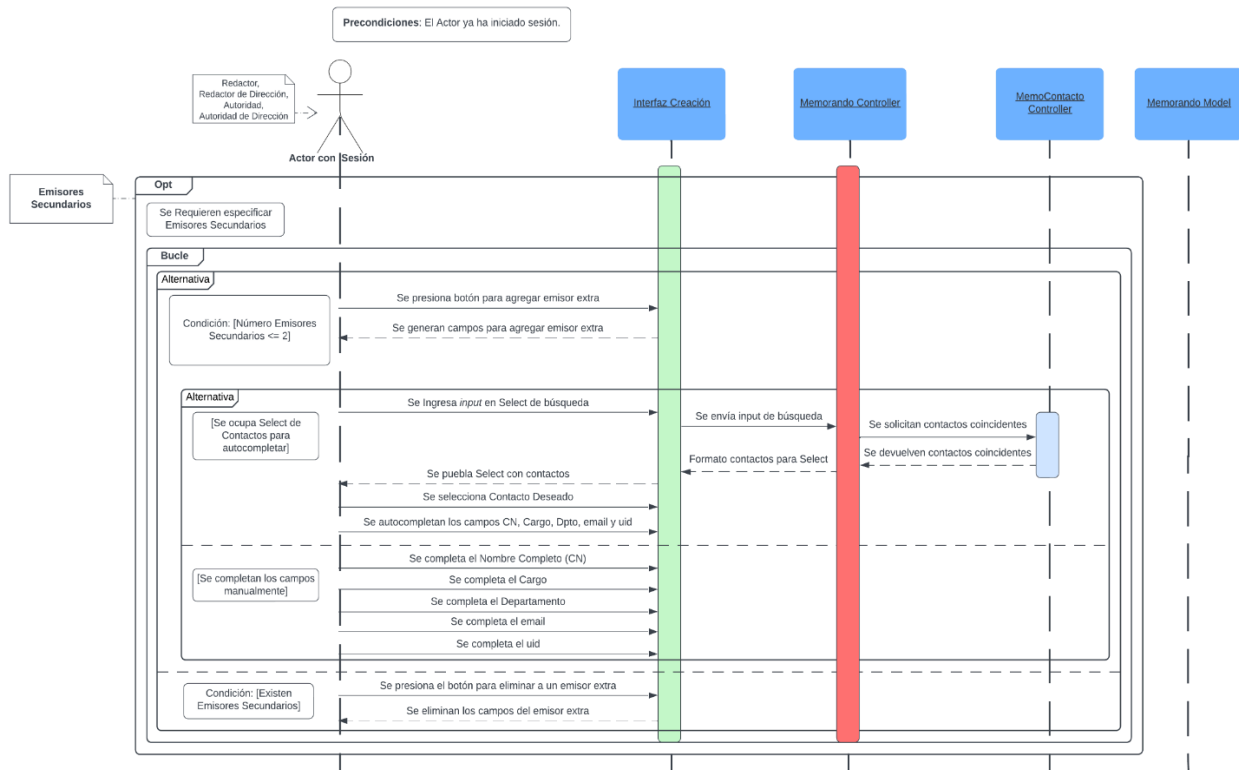
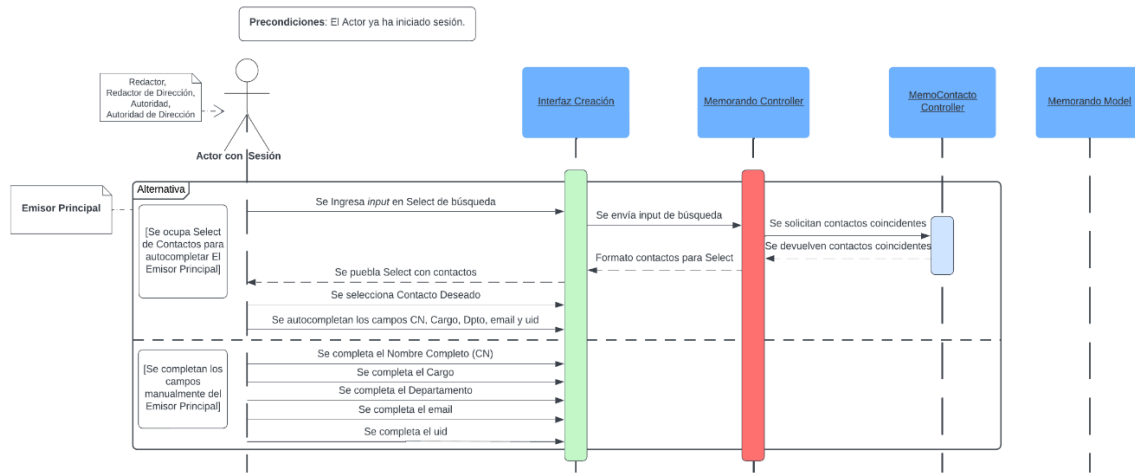
Tabla 3-8 Caso de Uso: Editar Memorando

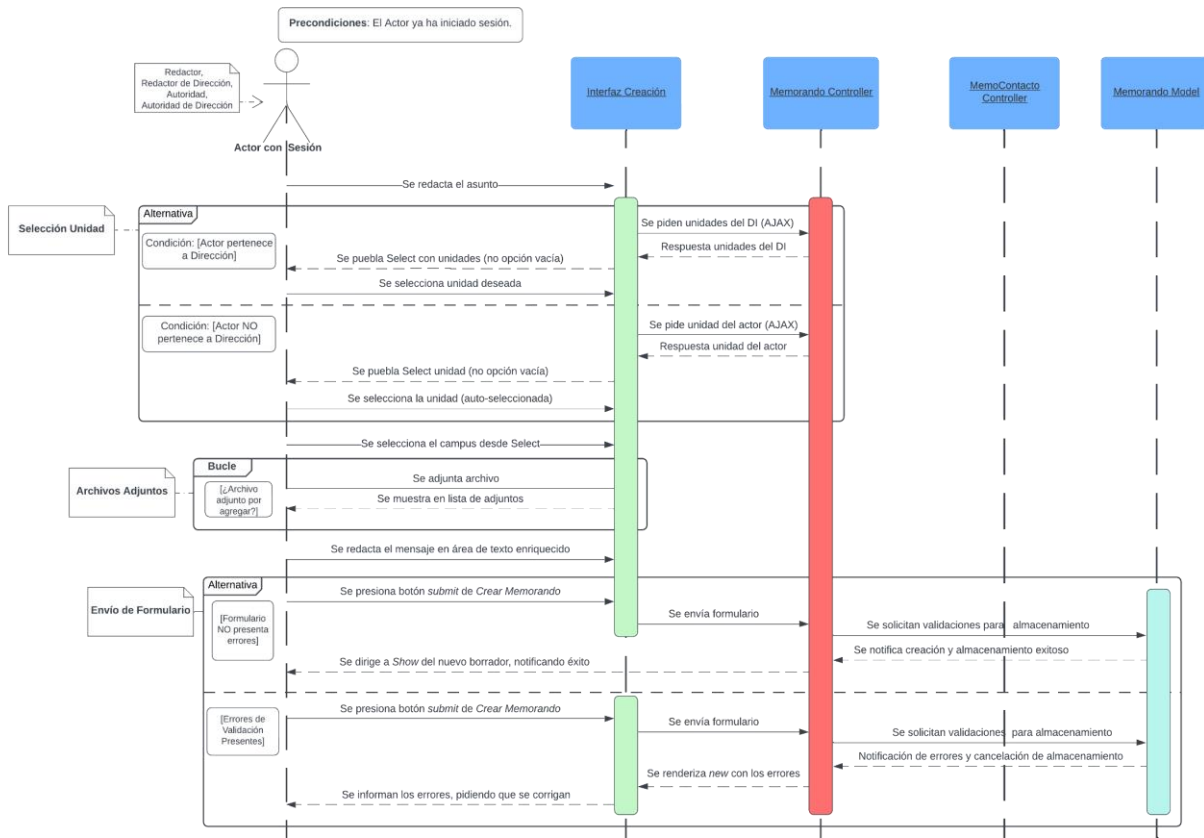
Paso	Usuario	Sistema
1	Usuario ingresa a Vista Edición Memorando de un Memorando existente, desde la Vista Detalles del Memorando o en un Listado de Memorandos propios (<i>Borradores, Plantillas, Remitidos</i>).	Sistema completa los campos del formulario con los datos del Memorando específico
2	Usuario modifica lo que necesite	

Se siguen los pasos de *Crear Memorando*...

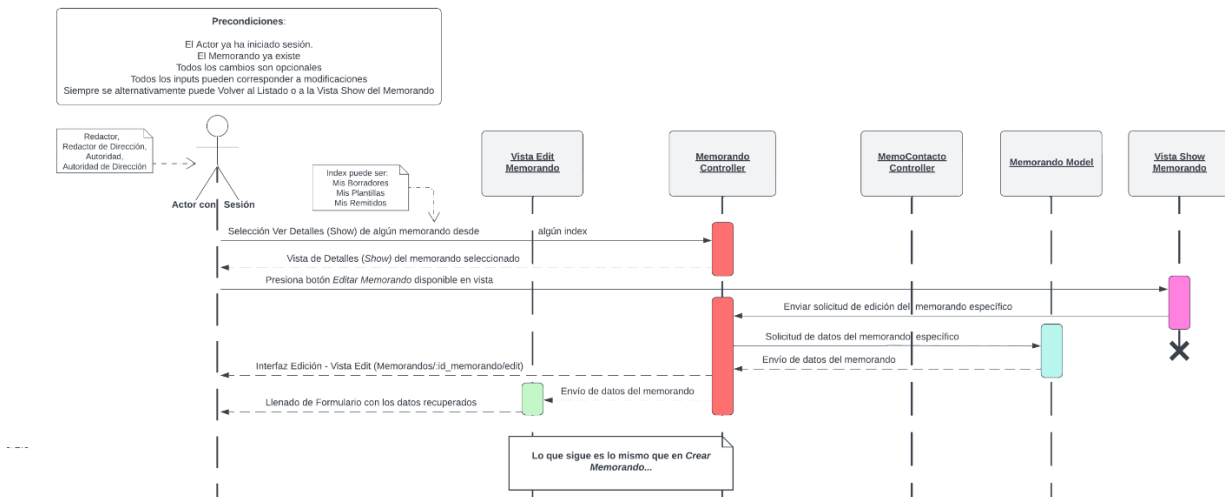
3.3.2.3 Diagrama de Secuencia – Crear Memorando







3.3.2.4 Diagrama de Secuencia - Editar Memorando



3.3.2.5 Visualización

Al crear un Memorando, se pueden sus campos y contenido en su Vista Detalles. Esta vista informa la fecha de creación y muestra una vista previa del PDF generado a partir de la información del Memorando, el cual se puede visualizar en pantalla completa. También enumera los archivos adjuntos en caso de incluirse. Revisar el Anexo B para más detalles.

3.3.2.6 Caso de Uso – Visualización de Memorando

- **Actores:** Redactor, Redactor Dirección, Emisor, Emisor Dirección, Administrador. Se usará Usuario genéricamente.
- **Precondiciones:** El Usuario se encuentra autenticado y está autorizado para ver los determinados Memorandos.
- **Postcondiciones:** El listado de Memorandos, la Vista Detalles y vuelta al Menú Principal ocurren correctamente.
- **Referencias Funcionales:** RF-2, RF-5.2, RF-5.3, RF-5.4, RF-5.5, RF-5.6 ([Ver Requerimientos](#)).

Tabla 3-9 Caso de Uso: Visualización de Memorando

Paso	Usuario	Sistema
1	Usuario ingresa a uno de los siguientes Listados de Memorandos: -Memorandos emitidos -Borradores -Remitidos -Plantillas. Si Usuario Dirección o Administrador, además: -Pendientes Autorización.	Sistema devuelve Memorandos del Listado seleccionado.
2	Usuario presiona ícono al lado de cada registro en Listado para ingresar a Vista Detalles de Memorando.	Sistema redirige al usuario a la Vista Detalle del Memorando seleccionado.

3.3.3 Plantillas

Una Plantilla en el sistema es un Memorando con estatus de *Plantilla*. Se ocupa para crear Memorandos con estatus de *Borrador* rápidamente a partir de los datos ya redactados de la Plantilla. No poseen *Número Correlativo* válido.

Todos los actores excepto el Huésped, pueden crear Plantillas a partir de Memorandos de cualquier estatus, excepto autorizados, es decir, se pueden crear Plantillas a partir de Memorandos con estatus *Borrador*, *Remitido* y *Plantilla*.

En la Vista Detalles de un Memorando apto para crear una Plantilla a partir de él, aparece la opción de *Crear Plantilla*. Al Presionar este botón, se creará inmediatamente un duplicado del Memorando, pero con estatus *Plantilla*, también, se anulan o reinician los valores de los campos *Fecha de Remisión* y *Número Correlativo* y se asigna el tiempo actual a *Fecha de Creación* de la nueva Plantilla.

Luego de la creación con éxito, se redirige a la Vista Detalles de la nueva Plantilla. Desde aquí se puede Editar Plantilla, y Crear Borrador a partir de ella.

El proceso de crear un Borrador a partir de una Plantilla es similar al reverso. En Vista Detalles de la Plantilla aparece opción exclusiva de *Crear Borrador*. Al presionar el botón, se crea inmediatamente un duplicado de la Plantilla, pero con estatus *Borrador* y se asigna el tiempo actual a *Fecha de Creación* del nuevo Borrador. Una vez creado, se redirigirá a la Vista Detalles del Borrador.

3.3.3.1 Caso de Uso – Crear Plantilla

- **Actores:** Redactor, Redactor Dirección, Emisor, Emisor Dirección, Administrador. Se usará Usuario genéricamente.
- **Precondiciones:** El Usuario se encuentra autenticado.
- **Postcondiciones:** El listado de Memorandos, la Vista Detalles y vuelta al Menú Principal ocurren correctamente.
- **Referencias Funcionales:** RF-2, RF-5.2, RF-5.3, RF-5.4 RF-5.5 ([Ver Requerimientos](#)).

Tabla 3-10 Caso de Uso: Crear Plantilla

Paso	Usuario	Sistema
1	Usuario ingresa a un Listado de Memorando, excepto el de <i>Autorizados</i> , y presiona en el botón Ver Detalles de un Memorando.	Sistema redirige Usuario a Vista Detalles del Memorando.
2	Usuario presiona botón <i>Crear Plantilla</i> en Vista Detalles de Memorando.	Sistema Crear una copia del Memorando específico: - Reemplaza estatus con <i>Plantilla</i> - Reemplaza fecha de creación con el tiempo actual.

		- Elimina valor de fecha de remisión, si existiese. - Ignora los archivos adjuntos
3		Redirige a Usuario a Vista Detalle de Plantilla con mensaje de éxito.

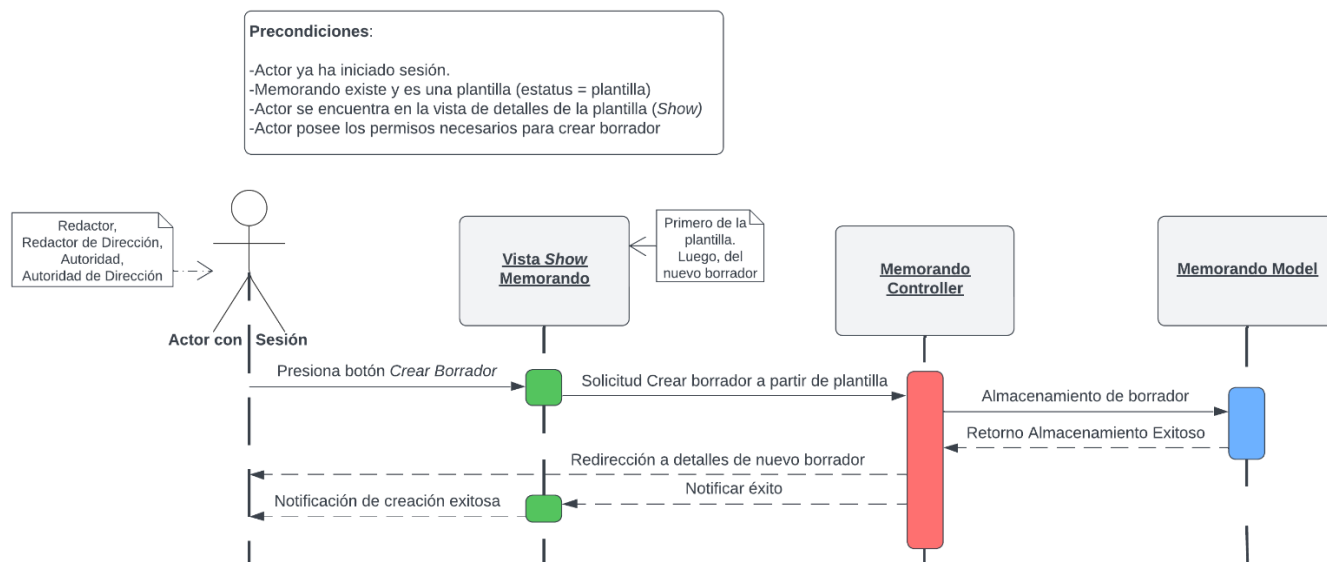
3.3.3.2 Caso de Uso – Crear Borrador a partir de Plantilla

- **Actores:** Redactor, Redactor Dirección, Emisor, Emisor Dirección, Administrador. Se usará Usuario genéricamente.
- **Precondiciones:** El Usuario se encuentra autenticado.
- **Postcondiciones:** El listado de Memorandos, la Vista Detalles y vuelta al Menú Principal ocurren correctamente.
- **Referencias Funcionales:** RF-2, RF-5.3 ([Ver Requerimientos](#)).

Tabla 3-11 Caso de Uso: Crear Borrador a partir de Plantilla

Paso	Usuario	Sistema
1	Usuario ingresa a Listado de <i>Plantillas</i> , y presiona en el botón Ver Detalles de una Plantilla.	Sistema redirige Usuario a Vista Detalles de Plantilla.
2	Usuario presiona botón <i>Crear Borrador</i> en Vista Detalles de Plantilla.	Sistema crea duplicado de la Plantilla específica: - Reemplaza estatus con <i>Borrador</i> - Reemplaza <i>Fecha de Creación</i> con el tiempo actual.
3		Redirige a Usuario a Vista Detalle de Borrador con mensaje de éxito.

3.3.3.3 Diagrama de Secuencia – Crear Borrador a partir de Plantilla



3.3.4 Eliminación de Memorando

3.3.4.1 Eliminación de Memorando

1. **Actores:** Redactor, Emisor, Administrador, se le llamará Usuario
2. **Precondiciones** El Usuario se encuentra autenticado y autorizado
3. **Postcondiciones** El Memorando es eliminado de la base de datos.
4. **Referencias Funcionales:** RF-2.1

Ilustración 3-9 Caso de Uso: Eliminación de Memorando

Paso	Usuario	Sistema
1	El mantenedor entra a la vista que enlista todos los Memorandos emitidos	El sistema le devuelve la lista de todos los Memorandos emitidos.
2	El mantenedor hace clic en el ícono para eliminar tal Memorando. También tiene acceso a la vista detallada del mismo en caso de querer comprobar más información de la presentada en la referencia a los registros	El sistema al usuario que confirme o cancele la acción de eliminación inminente
3	El mantenedor confirma la operación	El sistema elimina el Memorando y el registro de la base de datos y devuelve al usuario a la vista actualizada de registros.

3.3.5 Remisión de Memorando, Más Envío Notificación al Emisor

Un actor – como mínimo Redactor - creador de un Memorando con estatus actual de *Borrador* o *Rechazado*, una vez revisado y conforme con su contenido, puede promoverlo a un Memorando *Remitido*.

El proceso de remisión se puede gatillar desde la Vista Detalles del Memorando, presionando el botón *Remitir Borrador*.

Al presionarlo, primero se despliega una vista *modal* solicitando la confirmación de la remisión; también mostrará datos principales del Memorando, para que el usuario pueda asegurarse de que está remitiendo el Memorando correcto. Se puede cancelar la operación, lo que esconderá el *modal*, reactivando la Vista Detalles.

Una vez en el *modal*, Si se presiona el botón de *Confirmar Remisión*, se gatillará la remisión. El controlador modifica el Memorando, promoviéndolo a su nuevo estatus de *Remitido*, y le asigna el tiempo actual a su campo *Fecha de Remisión*, registrando el tiempo de la acción.

Además, luego de la actualización, se gatilla el envío de un correo electrónico al Usuario del DI señalado como Emisor, notificándole que posee un Memorando pendiente por autorizar.

Asimismo, el Emisor, al ingresar al portal, le aparecerá el Memorando remitido en su lista de *Pendientes por Autorizar*, en donde podrá ver sus detalles, editarlo y potencialmente autorizarlo o rechazarlo.

3.3.5.1 Caso de Uso – Remisión de Memorando

- **Actores:** Redactor, Redactor de Dirección, Emisor, Emisor de Dirección, Administrador. Se usará Usuario genéricamente
- **Precondiciones** el Usuario se encuentra autenticado y en Vista Detalles de Memorando existente.
- **Postcondiciones** El Memorando es enviado exitosamente al Emisor.
- **Referencias Funcionales:** RF-4.1 ([Ver Requerimientos](#))

Tabla 3-12 Tabla Caso de Uso Remisión de Memorando

Paso	Usuario	Sistema
1	Usuario presiona el botón <i>Remitir Borrador</i> en Vista Detalles	Sistema despliega vista <i>modal</i> con información principal del Memorando:

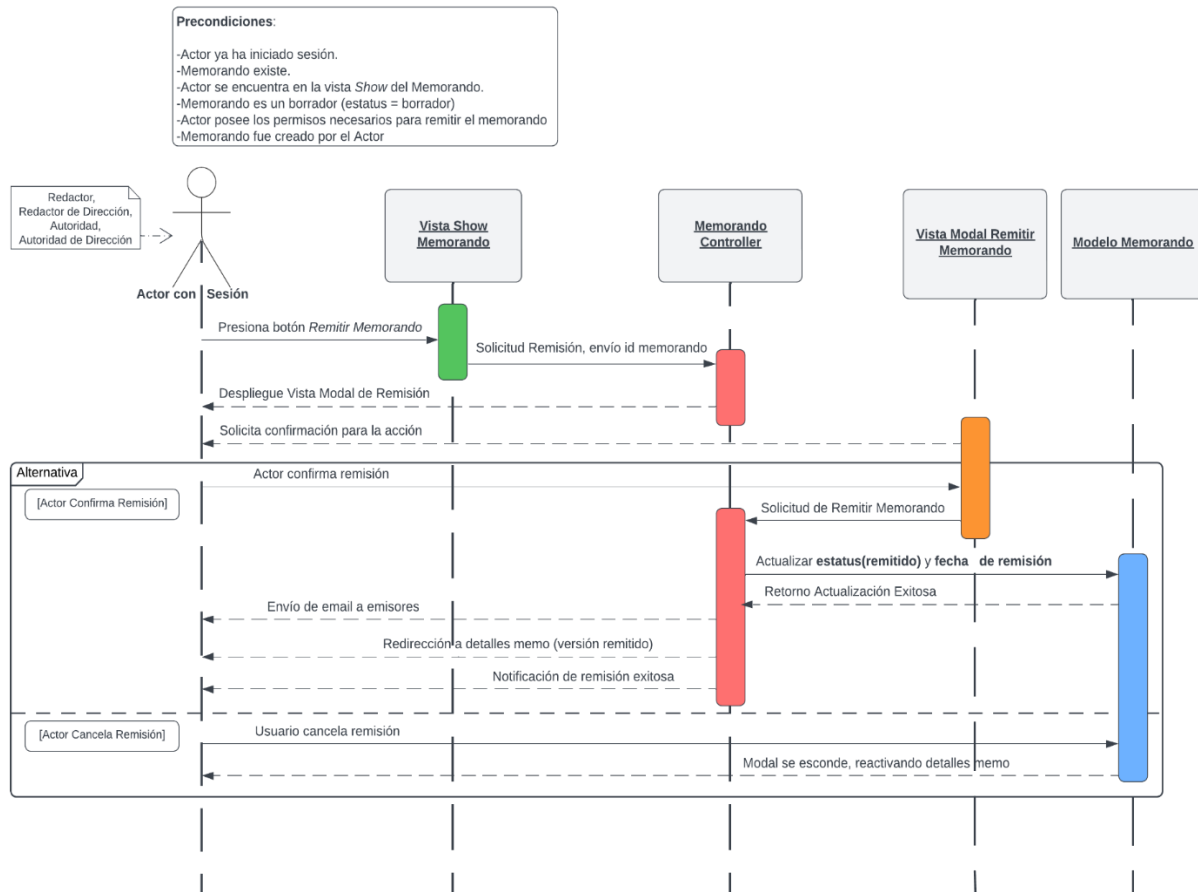
		- Número Correlativo - Emisor Principal - Destinatario Principal - Asunto
2	Usuario presiona el botón <i>Remitir Borrador</i> en <i>modal</i> ,	Sistema actualiza el Memorando: - Promueve estatus a <i>Remitido</i>. - Asigna <i>Fecha de Remisión</i> con tiempo actual. - Envía el correo electrónico con la notificación de Memorando Pendiente por Autorizar al Emisor a su correo institucional de Informática.
3		Redirige a Vista Detalles actualizada de Memorando ahora <i>Remitido</i> , con mensaje de éxito.

Camino Alternativos

Camino Alternativo A: Usuario cancela operación, cerrando el modal

Paso	Usuario	Sistema
2a	Usuario presiona el botón de <i>Cancelar</i> en <i>modal</i>	Sistema cierra el <i>modal</i> , reactivando la Vista Detalles.

3.3.5.2 Diagrama de Secuencia – Remisión de Memorando



3.3.6 Adición de Firma Electrónica Simple

Los Emisores poseen la capacidad de cargar al sistema una imagen de su firma, la cual luego se usará para firmar y los Memorandos Autorizados.

Esto crea una Firma Electrónica Simple (FES) que es aceptada por el DI, la UTFSM, respaldado por la contraloría de la Universidad, y por la legislación chilena, como un medio válido y suficiente para validar y autorizar documentación interna, como lo son los Memorandos.

En la página de Inicio, en el menú del perfil, aparecen las opciones *cambiar contraseña*, *tema oscuro*, *salir*. Se le sumará a este grupo la opción *agregar Firma Electrónica Simple*, desde donde se dirigirá al formulario de adición de FES.

Este formulario permitirá añadir imágenes de firmas en formatos png o jpg.

3.3.6.1 Caso de Uso Agregar Firma Electrónica Simple

- **Actores:** Emisor, Emisor de dirección.
- **Precondiciones** El Emisor se encuentra autenticado y autorizado. Emisor figura como el emisor en el Memorando en cuestión.
- **Postcondiciones** La FES se añade al perfil del Emisor exitosamente.
- **Referencias Funcionales:** RF-3.x, RF-5.5, RF-5.6 ([Ver Requerimientos](#))

Tabla 3-13 Caso de Uso: Agregar Firma Electrónica Simple

Paso	Usuario	Sistema
1	Emisor expande las opciones del menú de perfil de usuario y selecciona la opción <i>Agregar Firma Electrónica Simple</i>	Sistema redirige Emisor a formulario de adición de Firma Electrónica Simple. Si Emisor ya había cargado una FES, Sistema se la presenta en el formulario.
2	Emisor presiona en el campo de archivos <i>file_field</i> para buscar en la imagen de su firma a cargar.	
3	Una vez seleccionada, Emisor presiona botón <i>Guardar Cambios</i>	Sistema guarda firma en Base de Datos, asociándola al Usuario Emisor. Sistema Redirige a Emisor a formulario de adición de FES, pero ahora desplegando la imagen de la firma guardada.

Caminos Alternativos

Camino Alternativo A: Emisor quita una firma ya cargada

Paso	Usuario	Sistema
1.a	Emisor expande las opciones del menú de perfil de usuario y selecciona la opción <i>Agregar Firma Electrónica Simple</i>	Sistema redirige Emisor a formulario de adición de Firma Electrónica Simple. Sistema le presenta al Emisor la firma existente.
2.a	Emisor presiona el botón <i>Borrar Firma</i>	Sistema elimina la imagen de la firma almacenada en Base de datos.

		Sistema Redirige a Emisor a formulario de adición de FES, ahora sin firma que desplegar.
--	--	--

3.3.7 Autorización Digital

Si un actor con permisos en la Intranet de Emisor o Emisor Dirección es incluido como emisor de algún Memorando *Remitido*, se le notificará a su correo electrónico institucional de Informática, de que posee un Memorando pendiente por autorizar.

Al ingresar al portal de la Intranet, podrá ver en la sección de Memorandos el listado *Pendientes por Autorizar*, con los Memorandos remitidos y en los que participa como Emisor. Puede seleccionar uno para ver sus detalles, editarlo y potencialmente autorizarlo o rechazarlo.

Puede autorizar al presionar el botón *Autorizar Memorando* en la Vista Detalles del Memorando, este botón aparece solo si tiene permisos de Emisor y lo es para tal Memorando.

Al presionarlo, se despliega una vista *modal* solicitando confirmación para la autorización, también se muestra información principal del Memorando, para que pueda asegurarse que va a autorizar el Memorando correcto. Puede cancelar la operación, lo cual esconderá el *modal*, reactivando la Vista Detalles del Memorando.

Al presionara *Autorizar Memorando* en el *modal*, se gatilla el proceso de autorización. El controlador promueve el Memorando a estatus *Autorizado*, y le asigna el tiempo actual al campo *Fecha de Autorización*.

Además, se genera un código de verificación exclusivo del Memorando. Este código se genera ocupando:

- La Fecha de Autorización
- El Número Correlativo
- 3 componentes aleatorios de *salt* definidos por el sistema.

Este se almacena en el campo *cod_ver* (código de verificación) del Memorando

Una vez actualizado, se envía un correo electrónico con el Memorando autorizado al Destinatario principal, si se especificó su email; en caso de poseer *Con Copias*, también se les envía el Memorando a ellas.

Una vez un Memorando es autorizado, no se puede desautorizar ni editar. Tampoco se puede eliminar de la base de datos, ni siquiera por un Administrador. Los Memorandos autorizados se listan en la sección *Ver Memorandos Emitidos* del menú de Memorandos.

Para los actores no de Dirección, solo les aparecen los Memorandos emitidos (autorizados y enviados) de su Unidad. a los de Dirección, les aparecen todos los Memorandos emitidos del DI.

3.3.7.1 Caso de Uso – Autorización de Memorando

- **Actores:** Emisor, Emisor de dirección.
- **Precondiciones** la Emisor se encuentra autenticado y autorizado. Emisor figura como emisor en el Memorando en cuestión.
- **Postcondiciones** El Memorando es autorizado exitosamente.
- **Referencias Funcionales:** RF-3.x, RF-5.5, RF-5.6 ([Ver Requerimientos](#))

Tabla 3-14 Caso de Uso Autorización de Memorando

Paso	Usuario	Sistema
1	Emisor ingresa a Listado de <i>Pendientes por Autorizar</i> .	Sistema busca sus Memorando pendientes por autorizar y le devuelve el Listado.
	Emisor Ingresa a Vista Detalles de Memorando a autorizar	Sistema redirige Emisor a Vista Detalles del Memorando.
2	Emisor presiona botón <i>Autorizar Memorando</i>	Sistema despliega <i>modal</i> con información principal del Memorando: - Número Correlativo - Emisor Principal - Destinatario Principal - Asunto Solicitando confirmación a la autorización.
3	Emisor presiona botón <i>Autorizar Memorando en modal</i>	Sistema actualiza el Memorando: - Promueve estatus a <i>Autorizado</i>. - Asigna tiempo actual a campo <i>fecha de autorización</i>.

		- Genera un código de verificación exclusivo del Memorando. - Envía correo electrónico a los destinatarios (cuyos emails fueron especificados) con el Memorando autorizado.
		Sistema redirige a Emisor a Vista Detalles actualizada de Memorando

Camino Alternativos

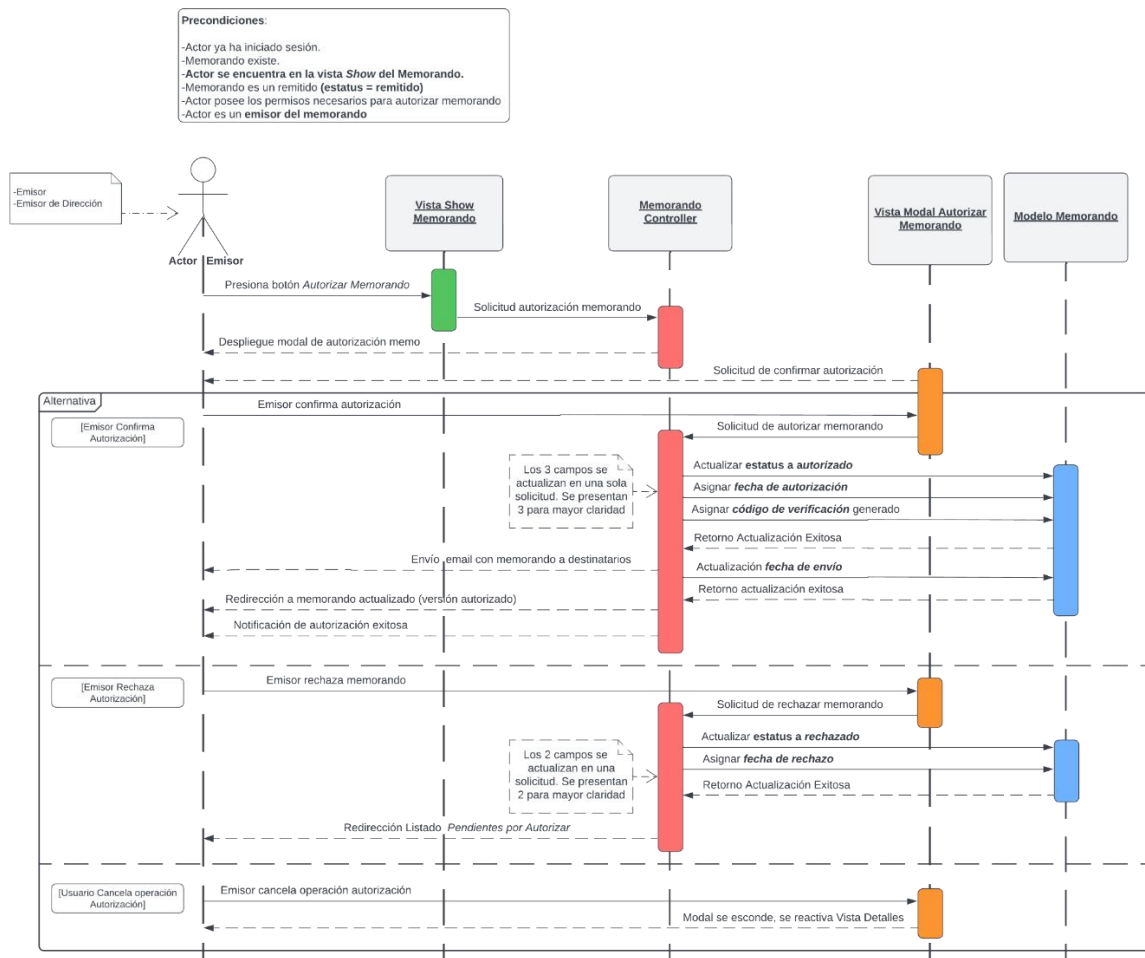
Camino Alternativo A: Emisor cancela autorización, cerrando el modal

Paso	Usuario	Sistema
3a	Emisor presiona el botón de <i>Cancelar en modal</i>	Sistema cierra el <i>modal</i> , reactivando la Vista Detalles.

Camino Alternativos B: Emisor Rechaza el Memorando

Paso	Usuario	Sistema
3b	Emisor presiona botón <i>Rechazar Memorando en modal</i>	Sistema actualiza el Memorando: - Desciende estatus a <i>Rechazado</i>. - Asigna tiempo actual a campo <i>Fecha de Rechazo</i>.
		Sistema redirige a Emisor a Listado de <i>Pendientes por Autorizar</i> con mensaje de rechazo exitoso Memorando rechazado ya no está presente en Listado.

Diagrama de Secuencia – Autorización Memorando



3.3.8 Búsqueda

3.3.8.1 Listados

Se usará la palabra Usuario, para referirse a un Redactor, Emisor o Administrador.

Los diferentes Listados de Memorandos son:

- **Ver Memorandos Emitidos**, con los Memorandos autorizados. Si es Usuario de Dirección, podrá ver los Memorandos autorizados de todas las Unidades. En caso contrario, solo verá las de su Unidad.
- **Mis Remitidos**, listado Memorandos creados y remitidos por el Usuario con estatus *Remitido*.
- **Mis Rechazados**, listado Memorandos que fueron remitidos, pero que el Emisor rechazó. estos poseen estatus de *Rechazado* y se pueden volver a remitir.
- **Mis Borradores**, listado Memorandos creados por el Usuario con estatus *Borrador*.

- **Mis Plantillas**, listado Memorandos creados por el Usuario con estatus *Plantilla*.
- **Autorización Pendiente**, solo disponible para Emisor o Emisor de Dirección. Listado de Memorandos Remitidos en los que participa como Emisor que aún no ha autorizado o rechazado.

3.3.8.2 Búsquedas

En los Listados, se pueden filtrar los resultados por los siguientes criterios:

1. **Rango de fechas**
2. **Emisor Principal**
3. **Destinatario Principal**
4. **Número Correlativo**
5. **Asunto**
6. **Unidad**

3.3.8.3 Caso de Uso – Búsqueda de Memorandos por Criterios

- **Actores:** Redactor, Redactor de Dirección, Emisor de Dirección, Administrador. Se le llamará Usuario genéricamente
- **Precondiciones:** Usuario se encuentra autenticado.
- **Postcondiciones:** Búsqueda se filtra y los datos se muestran correctamente.
- **Referencias Funcionales:** RF-5.x ([Ver Requerimientos](#)).

Tabla 3-15 Caso de Uso Búsqueda por Criterios, Fuente: Elaboración Propia

Paso	Usuario	Sistema
1	El Usuario ingresa a un listado de Memorandos de los disponibles: - Memorandos Emitidos - Remitidos - Rechazados - Borradores - Plantillas Si Usuario es Emisor	Si Usuario es de Dirección, Sistema devuelve todos los Memorandos asociados al Listado. En caso contrario, Sistema solo muestra Memorandos asociados al Listado de su unidad.

	- Autorización Pendiente	
2.1	Usuario define rango seleccionando una fecha inicial y una final.	Sistema devuelve Memorandos en el rango de fecha especificado: - Si Memorandos Emitidos, rango de <i>Fecha de Autorización</i>. - Si Remitidos, rango de <i>Fecha de Remisión</i>. - Si Rechazados, rango de <i>Fecha de Rechazo</i> - Si Borradores, rango de <i>Fecha de Creación</i>. - Si Autorización Pendiente, rango de <i>Fecha de Remisión</i>.
2.2	Usuario ingresa caracteres en campo de texto general en algún Listado:	Sistema devuelve Memorandos que coincidan total o parcialmente con: -Nombre Completo Emisor Principal. -Nombre Completo Destinatario Principal. -Asunto. -Número Correlativo
2.3	Si es Usuario de Dirección o Administrador, se muestra <i>Select</i> de Unidades del DI	
2.3.1	Usuario con Privilegios selecciona Unidad en <i>Select</i>	Sistema filtra Memorandos por Unidad seleccionada.

3.3.9 Permisos

Para implementar los diferentes actores, se crean nuevos roles, que se encuentran compuestos de nuevos permisos avanzados, en la Intranet.

Existen 2 tipos de permisos: de Redactor y de Emisor. Estos, a la vez, pertenecen a una Unidad específica, habiendo uno por cada una de las 6 Unidades del DI. Por lo que tendremos 6 Redactores y 6 Emisores distintos, completando un total de 12 nuevos roles. Estos roles se crean ocupando el formulario de creación de roles de la Intranet:

Crear nuevo Rol

[← Volver al listado](#)

Unidad

Dirección

Nombre

Memo Redactor

Crear Rol

Ilustración 3-10 Formulario de Creación de Roles de la Intranet, Fuente: Elaboración Propia

Dando como resultado los 12 roles Sgtes:

Memo redactor Dirección
Memo emisor Dirección
Memo redactor Infraestructura y Tecnología
Memo emisor Infraestructura y Tecnología
Memo redactor Investigación
Memo emisor Investigación
Memo redactor Postgrado
Memo emisor Postgrado
Memo redactor Pregrado
Memo emisor Pregrado
Memo redactor Vinculación con el Medio
Memo emisor Vinculación con el Medio

Ilustración 3-11 Roles de Usuarios de Memorandos Para la Intranet, Fuente: Elaboración Propia

Los roles de Redactor y sus permisos asociados le permiten al Usuario acceder al módulo y poder crear Memorandos. También le permiten ver aquellos que ha creado, tanto borradores como plantillas, aquellos que ha remitido y los demás Memorandos pertenecientes a la Unidad del permiso, que corresponde a la Unidad del Usuario; si es de Dirección podrá ver todos los Memorandos del DI.

El permiso de Emisor permite las acciones anteriores y además permite autorizar Memorandos, convirtiéndolos en emitidos válidos, que cuentan con su respaldo y que se pueden verificar.

Dentro de las acciones o métodos del Controlador de Memorandos (*MemorandosController*) a las que un Redactor puede acceder, tenemos:

- New & Create: crear Memorando, solo de su Unidad (formulario de creación de Borrador)
- Templating: crear Plantilla (a partir de Borrador, Remitido o Plantilla)
- Creating_from_template: crear Borrador a Partir de Plantilla
- Edit & Update: actualizar Memorando (solo de su Unidad)
- Show: ver detalles.
- Delete: eliminar Memorando, excepto aquel autorizado.
- Confirm: Remitir memorando al emisor declarado.
- Indexes: Listados.

- Index: índice de Memorandos Emitidos, solo de su Unidad
- Borradores: índice de Borradores creados por el Usuario actual
- Plantillas: índice de Plantillas creadas por el Usuario actual
- Remitidos: índice de Borradores remitidos por el Usuario actual
- Rechazados: índice de Remitidos del Usuario actual rechazados por el Emisor declarado.
- Búsquedas en los índices
 - Por Rango de fechas
 - Por Búsqueda general en campo de texto, este busca por:
 - Número Correlativo
 - Emisor (Principal)
 - Destinatario Principal
 - Asunto

Adicionalmente a lo anterior, un Redactor de Dirección puede:

- New & Create: crear Memorando, de cualquier Unidad (formulario de creación de Borrador)
- Indexes: Listados
 - Index: índice de Memorandos Emitidos (de todas las Unidades)
- Búsquedas en los índices
 - Por Unidad

Un Emisor (no de Dirección) tiene acceso a los mismos métodos de un Redactor (no de Dirección), pero también adicionalmente a:

- Authorize & Authorizing: autorizar Memorando, en los que es emisor (de su Unidad)
- Reject & Rejecting: rechazar Memorando, en los que es emisor (de su Unidad)
- Indexes: Listados
 - Pendientes: índice de Remitidos Pendientes de Autorización por Usuario actual.
 - Rechazados: índice de Remitidos rechazados por el Usuario actual.

Un Emisor de Dirección tiene acceso a los mismos métodos que un Redactor de Dirección, pero también adicionalmente a:

- Authorize & Authorizing: autorizar Memorando, en los que es emisor (de cualquier Unidad)
- Reject & Rejecting: rechazar Memorando, en los que es emisor (de cualquier Unidad)
- Indexes: Listados
 - Pendientes: índice de Remitidos Pendientes de Autorización por el Usuario actual
 - Rechazados: índice de Remitidos rechazados por el Usuario actual.

El Usuario Huésped no tiene asociado ningún permiso, este representa a un usuario que no ha iniciado sesión, puede que no tenga cuenta en el DI. El Huésped solo tiene acceso a verificar un Memorando, acción disponible para cualquiera.

A continuación, se muestra un Diagrama con los métodos permitidos por Actor o Perfil de Usuario:

Perfil Usuario	Redactor	Redactor Dirección	Autoridad	Autoridad Dirección	Administrador	Huésped
Función						
Creación	✓ Solo de su unidad	✓	✓ Solo de su unidad	✓	✓	
Edición	✓ Solo de su unidad	✓	✓ Solo de su unidad	✓	✓	
Borradores (del usuario)	✓ Solo de su unidad	✓	✓ Solo de su unidad	✓	✓	
Plantillas (del usuario)	✓ Solo de su unidad	✓	✓ Solo de su unidad	✓	✓	
Ver detalles de Memo	✓	✓	✓	✓	✓	✓ Del memo Verificado
Índex de memos	✓ Solo de su unidad	✓	✓ Solo de su unidad	✓	✓	
Remisión (de memos del usuario)	✓ Solo de su unidad	✓	✓ Solo de su unidad	✓	✓	
Autorización (si declarado emisor)			✓ Solo de su unidad	✓		
Búsqueda por Filtros	✓	✓	✓	✓	✓	
Verificación	✓	✓	✓	✓	✓	✓
Des-remisión (excepto autorizados)					✓	
Eliminación (excepto autorizados)					✓	

Ilustración 3-12 Diagrama de Permisos por Perfil de Usuario, Fuente: Elaboración Propia

Los permisos en la Intranet se implementan y revisan por medio de la gema *CanCanCan*.

Se definieron alias, por medio del método de *CanCanCan*, *alias_action*, para agrupar etiquetas, definidos, como todos los permisos de *CanCanCan*, en *models/ability.rb*. Los alias definidos son los Sgtes.

```
# Alias para Memorando -----  
  
alias_action :store, :storing, to: :can_store  
alias_action :confirm, :confirming, to: :can_remitir  
alias_action :templating, :creating_from_template, to: :can_template  
alias_action :contactos_dpto_info_search, :contactos_search, to: :can_search_contactos  
alias_action :read, :create, :update, :destroy, to: :crud  
  
alias_action :can_store, :can_remitir, :can_template, :can_search_contactos, :crud, to: :memo_redactor  
  
alias_action :borradores, :plantillas, :remitidos, :load_more_memorandos, to: :can_list_its_memos  
  
alias_action :authorize, :authorizing, to: :can_authorize  
alias_action :memo_redactor, :can_authorize, :pendientes, to: :memo_emisor
```

*Ilustración 3-13 alias_action de CanCanCan creados para los Permisos del Módulo, Fuente:
Elaboración Propia*

En alto nivel, los alias agrupan los métodos relacionados de *MemorandosController* y luego se ocupan estos grandes grupos para definir 3 etiquetas principales, el de *memo_redactor*, para los Redactores, y *memo_emisor*, para los Emisores; finalmente hay un alias *can_list_its_memos*, que agrupa permisos de Listado que es común para ambos tipos de actores anteriores. Estas etiquetas se ocupan en el formulario de creación de permisos de la Intranet.

Además, a ambos roles *Memo Redactor* y *Memo Emisor*, se les agrega la etiqueta *can_list_its_memos*, que les permite tener acceso a los Listados de sus propios memorandos, siendo estos, los *Borradores*, *Plantillas* y *Remitidos*. El permiso asociado es a los Memorandos creados (que pertenecen) al Usuario actual y restringidos por la Unidad de este.

Se ocupan los métodos helpers especializados de *CanCanCan* para llevar a cabo la autorización de los métodos en *MemorandosController*, en particular, los métodos *authorize_resource* y *load_and_authorize_resource*.

```

class MemorandosController < ApplicationController
  include MemorandosHelper # Funciones del Helper de memorandos.

  before_action :logged? # Requerir que el usuario esté logueado en todas las acciones.
  # Se busca el memo con :id recibido
  before_action :ensure_js_format, only: [:confirm, :store, :authorize]

  load_and_authorize_resource :memorando, includes: :memo_participantes, only: [:show, :update, :destroy,
  :can_template]

  load_and_authorize_resource :memorando, only: [:can_remitir, :disconfirm, :can_store, :can_authorize]

  before_action :check_memorando_status, only: [:update, :destroy, :can_remitir] # Revisa si el valor

  # Permisos y autorización de CanCanCan de Memos. Para todas las acciones.
  authorize_resource :memorando, except: [:check, :checking]
end

```

Los métodos para verificar un Memorando no se autorizan, ya que necesitan tener acceso a ellos.

Capítulo 4: VALIDACIÓN DE LA PROPUESTA

4.1 Prueba General

Luego de diferentes instancias en las cuales se entrevistó, consultó, desarrolló y se presentaron prototipos parciales. Se llevó a cabo una prueba con un prototipo funcional en las Unidades de Dirección y Pregrado.

En esta prueba general participaron Usuarios Redactores, de Dirección y Pregrado, y un Emisor de Pregrado.

Se confeccionó y se les envió un documento detallando los pasos a seguir y las funcionalidades a poner a prueba, guiándolos para probar todas las funcionalidades y el flujo de la aplicación.

4.1.1 Resultado General de las Pruebas

1. La acogida y crítica del módulo por parte de Los Redactores de Dirección y Pregrado fue positiva, mostrando gran interés en su continuo desarrollo y posible implementación. El Sistema de Gestión de Memorandos representa una bienvenida y deseada oportunidad de mejora que les facilita su trabajo y sus comunicaciones internas. En general, solo detectaron mejoras menores a realizar.
2. El Emisor de Pregrado, detectó inconveniencias en el Editor de Texto, en la Verificación de un Memorando y en la adición de una Firma Electrónica Simple. Sin embargo, obviando estos detalles fácilmente corregibles, mostró apoyo e interés por la iniciativa y el prototipo, concluyendo que toda automatización es bienvenida si representa una mejora en el servicio y acceso a la información, aportando y facilitando el trabajo del DI.
3. En general, se expresó que el formulario de creación era intuitivo y fácil de navegar, pero que se podría mejorar. El Editor de Texto en el navegador proveía de las funciones frecuentemente usadas, pero que no ofrecía ciertas comodidades que se encuentran en MS-Word, por lo que es mejorable.
4. Se logró comprobar que el formato estándar de un Memorando propuesto es el correcto, solo se requieren más pruebas para confirmar pequeños detalles asociados a la nomenclatura de los *Con Copia*, y posición del Emisor y Firma.
5. La apariencia del PDF generado también fue aceptada, pero mejorable en pequeños detalles.
6. El flujo de Redactor Redacta, Remite y Emisor Autoriza, fue percibido como intuitivo.
7. La adición de una FES fue recibida positivamente, pero con necesidad de mejoras.
8. Los Filtros de búsqueda fueron bien recibidos y funcionaron correctamente.
9. La Características más destacada fue la centralización de los Memorandos y el historial a disposición, el flujo de Redactor Redacta – Emisor Autoriza, la que ocurre en el mismo sistema. La característica que requiere más mejoras es el Editor Integrado.

En general el prototipo fue validado como un Sistema con alto interés y deseado. En gran parte cumplió con lo esperado, a excepción de ciertos arreglos. Arreglos que se esperan ya que se apoyaría la continua exploración y desarrollo.

4.1.2 Retroalimentaciones

4.1.2.1 Redactora de Dirección: Angela Miranda, secretaria de Dirección

La sugerencia de primero tener que crear Contactos para luego usarlo ocuparlos para autocompletar no fue inicialmente intuitiva.

Solo se tuvo que explicar el flujo una sola vez, luego operar las diferentes Vistas no tuvo inconvenientes.

Angela estuvo presente en todas las reuniones y fue una principal fuente de retroalimentación, por lo que su familiaridad con el prototipo le ayudó a completar la pauta sin problemas.

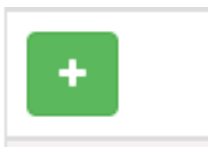
4.1.2.2 Redactora de Pregrado: Erika Guzmán, secretaria de Pregrado

No tuvo mayores problemas, siguiendo la Pauta. Además, al estar presente en las presentaciones previas, recordaba la manera de operar. Por lo que la interfaz le fue más familiar.

Detalles sobre el Editor y sus funcionalidades.

4.1.2.3 Redactor de Dirección: José Miguel González, Encargado de Gestión Financiera

1. En el prototipo actual, los Memorandos *Borradores* y *Plantillas* son solo visibles por su Redactor. Los Memorandos *Remitidos* son solo visibles por su Redactor y por el respectivo Emisor.
 - a) Se sugirió que al menos los Memorandos *Remitidos* sean compartidos por los miembros de una misma Unidad. De esta manera se puede coordinar mejor operativamente entre los miembros Redactores y Emisores de la Unidad.
2. En el prototipo actual, en el formulario de creación, el botón para agregar Destinatarios adicionales consiste en un botón verde con un símbolo de más:



- a) Se consideró no lo suficientemente intuitivo, se sugirió especificar con texto su función.

4.1.2.4 Emisor de Pregrado: Pedro Godoy, jefe de Carrera Ingeniería Civil en Informática

1. El prototipo actual fue diseñado para, una vez insertada toda la información de un Memorando, recién mostrar el PDF con la estructura y apariencia resultante.

- a) Se detectó esto como un inconveniente a mejorar, ya que no hay sensación de redacción en tiempo real, teniendo que volver entre la edición y los resultados, entregando una experiencia no mejor a la actual.
2. El prototipo actual, en el formulario de creación de Memorandos, hace uso de los Contactos para autocompletar los campos de los Destinatarios o Emisor. Pero, para usar Contactos en el formulario, se requiere crearlos previamente. Sin embargo, también los campos de Destinatarios y Emisor se pueden completar manualmente sin necesidad de un Contacto.
 - a) Se sugirió que, para facilitar y dar mayor rapidez al proceso de creación, que al momento de completar un manualmente los campos de algún Destinatario o Emisor, que se ofrezca la opción de crear en el lugar el Contacto asociado para su uso posterior.
 3. A la hora de agregar una Firma Electrónica Simple, al no incluir un archivo y enviar el formulario de creación, se gatilló un error. Un error básico que se corrigió fácil y rápidamente.
 4. El prototipo actual le muestra a un Usuario (Redactor y Emisor) no de Dirección, solo los Memorandos Autorizados de su Unidad, facilitando la búsqueda de estos por medio de diferentes criterios, como búsqueda por Asunto y Participantes, por rango de Número Correlativo, por Rango de Fechas; la búsqueda por Unidad se encuentra disponible solo para los Usuarios de Dirección.
 5. Se apreció la centralización de los Memorandos en un solo lugar, proveyendo fácil acceso y búsqueda de los Memorandos, apoyando a la comunicación del DI. Pero el Jefe de Carrera expresó su interés en también ser capaz de acceder y revisar los Memorandos de otras Unidades, por lo que la restricción inicial no aplicaría.
 6. Respecto al modo de operar al Remitir un Memorando, prefirió que un Memorando Remitido, pendiente por autorizar, no se pueda Editar ni por el Redactor ni el Emisor. Que quede en estado de revisión. Si es rechazado por el Emisor, entonces el Redactor tendrá acceso a modificarlo una vez más, para volver a remitirlo.
 7. Respecto de los Listados, sugirió una leyenda que clarifique el significado del estado de un Memorando

4.1.3 Formularios de Análisis de Experiencia

Luego de las pruebas se les envió a los participantes *Google Forms* con preguntas sobre su experiencia en los siguientes enlaces, que se pueden compartir si se requiere: Google Form para Emisor, Google Form para Redactores.

4.2 Futuro del Prototipo

El prototipo presentó dificultades para implementarse debido a que fue diseñado como un módulo parte de la Intranet. Hacia el final de esta memoria, se fue informado por el Jefe de I, T & S, que debido a problemas en la mantención y por la antigüedad del proyecto y del código de la Intranet actual, se inició un proyecto de actualización, por medio del desarrollo de una nueva Intranet V2. Debido a esta razón, no se están aceptando cambios a la Intranet actual, por lo cual el módulo no podría pasarse a producción, a pesar de estar altamente preparado para llevarse a cabo tal proceso.

Debido al avance logrado y el poco tiempo restante cuando se tomó conocimiento de esta situación, se determinó desarrollar un prototipo funcional a presentar y registrar todo el conocimiento y retroalimentación obtenidos en esta memoria.

Si existe suficiente interés, que en el transcurso de esta memoria se detectó, en el Sistema de Gestión de Memorandos, su sucesor se puede inspirar en esta memoria para no partir de cero.

Para el futuro del sistema hay 2 opciones a considerar:

1. Que el sistema se implemente en la nueva versión V2 de la Intranet.
 - a) Esto involucra un mayor plazo de desarrollo antes de que pueda estar disponible, pero será parte del mismo sistema de la Intranet, por lo cual se obtendrá mayor integración y un sistema más completo, centralizando la gestión general.
2. Que el sistema se implemente como una aplicación especializada, que se autentique con los usuarios del DI y se conecte a los sistemas que requiera.
 - a) Esto involucra más sistemas con los cuales operar y a mantener, pero involucra menor plazo de desarrollo antes de que encuentre disponible. Considerando todo el trabajo hecho, el sistema podría estar operando dentro de unos pocos meses.

Capítulo 5: CONCLUSIONES

5.1 Conclusiones por Objetivo

5.1.1 Conclusión por Objetivo Principal

Se logró diseñar y desarrollar un módulo parte de la Intranet, que expande sus funcionalidades para encargarse de la gestión digital de los Memorandos del DI. El módulo desarrollado permite:

- Creación y edición
- Almacenamiento y registro
- Envío de notificaciones de Memorandos pendientes por autorizar
- Capacidad de autorización por parte de Emisor y posterior verificación por parte de un tercero interesado o Huésped.
- Búsqueda por medio de diferentes criterios
- Centralización y disponibilidad desde cualquier dispositivo, siempre que se tenga acceso a internet y se posea una cuenta en la Intranet.

El Sistema cumple los objetivos de facilitar y acelerar el trabajo realizado por los funcionarios encargados de esta gestión, llevado a cabo principalmente por las Secretarías. En consecuencia, otorga una mejora en las comunicaciones oficiales del DI y su administración, aportando a su transformación digital.

5.1.2 Conclusión por Objetivos Específicos

Se llevó a cabo:

1. El diseño de una estructura y formato estándar para la creación de Memorandos.
2. Se ofrece una experiencia automatizada, rápida y simple de edición de Memorandos. Se ofrece un editor de texto enriquecido básico que cumple con las características de edición web necesarias. También se ofrece la autocompletación a la hora de añadir un emisor(es) y un destinatario(s) por medio de Contactos guardados.
3. Se consideran actores con diferentes permisos, que permite que los Redactores puedan crear Memorandos, pero que solo los Emisores puedan autorizarlo.
4. Se diseñó un sistema de autorización de Memorandos que permite al Emisor autorizarlo y darle validez digitalmente, pudiendo prescindir de la impresión obligatoria. Solo puede ser autorizado por el Emisor aludido. En la autorización se calcula un código de verificación único para el Memorando.
5. Se desarrolló un sistema de envíos automáticos de Memorandos:
 - a. Envío de notificación al Emisor de Memorando pendiente por autorizar
 - b. Una vez autorizado, envío de Memorando al Destinatario y a los *Con Copia* especificados

6. Se desarrollaron listados y búsquedas de Memorandos por medio de diferentes criterios, siendo estos, Usuario se refiere a Redactor, Emisor y Administrador:

a. Listados

- i. Listar *Borradores* de Memorandos del Usuario
- ii. Listar *Plantillas* de Memorandos del Usuario
- iii. Listar *Remitido* de Memorandos en espera de autorización del Usuario
- iv. Listar Memorandos pendientes por autorizar del Emisor
- v. Listar los Memorandos autorizados de la Unidad del Usuario. Si el Usuario es de Dirección, entonces, listar los Memorandos autorizados del DI.

b. Búsquedas

- i. Rango de fechas
- ii. Campo de Texto de Búsqueda General
 - 1. Emisor Principal Coincidente
 - 2. Destinatario Principal
 - 3. Asunto
 - 4. Número correlativo
- iii. Unidad

5.2 Conclusiones Generales

Se exploraron diferentes metodologías en el transcurso de esta memoria.

Inicialmente se exploró una metodología Ágil, centrándose en el desarrollo, funcionalidad, completitud y correctitud. Mucho tiempo se invirtió en investigar, buscar soluciones y se buscó desarrollarlas en casi su totalidad antes de presentarlas. Se centró en la investigación y en el sistema tecnológicamente correcto, más que el necesario.

Posteriormente, se ajustó la metodología para inspirarse en *Design Thinking* y la filosofía *Human Centered Design*, centrando el desarrollo en el usuario, en entender y profundizar en sus necesidades, deseos y en su situación. Se buscó lograr prototipar muchos más rápido, para recibir la retroalimentación rápidamente de los usuarios y las personas involucradas directa o indirectamente en la solución, guiando y adaptando los requerimientos y el desarrollo a sus sugerencias e interacciones. Se centró en el sistema deseado y necesario más que en el tecnológicamente correcto. Con este nuevo proceso, el desarrollo se aceleró considerablemente, logrando avanzar con mayor fluidez y certeza.

Para el desarrollo de la memoria fue esencial pensar en una metodología ágil, pero el mayor beneficio se obtuvo al ajustar e inspirarse en la filosofía *Human Centered Design* y en su *Framework Design Thinking*. Este no solo aumentó la certeza y aceleró el desarrollo, pero también logró impedir que el método Ágil se convirtiera en *Cascada oculta de facto*.

Por otro lado, se espera que el sistema propuesto sea un aporte para el DI y sea usado. Se concluye que tiene potencial por las siguientes razones:

Esta memoria fue inspirada al realizar mi práctica profesional en el DI. De esta experiencia se recogió la oportunidad de mejora de la digitalización de los Memorandos de planes ya existentes y de sugerencias de sus miembros. En el transcurso de la memoria, el interés en el sistema se ha mantenido. Por lo que, el sistema posee interés en los usuarios y ofrece una solución a un deseo de su parte.

La solución resuelve un tema puntual pequeño, pero de una tarea diaria común.

Al ser parte de la Intranet, no involucra nuevas tecnologías, infraestructuras o sistemas. Por lo tanto, la solución es económica de implementar y mantener.

5.3 Conclusiones Personales

Personalmente, puedo concluir que he logrado aprender una gran cantidad en el desarrollo de esta memoria, más que sobre tecnologías, sobre el proceso de desarrollo de un software en sí. Mi conclusión principal es la de tener a los usuarios siempre al centro, tener sus sugerencias en la mayor consideración y como principal guía.

Por lo anterior, también concluyo que gran parte de los esfuerzos en el desarrollo de un software deben centrarse en fomentar y mantener esta comunicación. Hay que dedicar tanto esfuerzo a la constante comunicación, como al desarrollo técnico. Hay que tener siempre presente que uno está proponiendo una solución a una organización y miembros de ella, a personas.

El desarrollo resultó ser más efectivo y eficaz entre más fuera capaz de mantener esta constante comunicación.

Por otra parte, desde una retrospectiva académica, entre los ramos que se consideran que fueron los más útiles para el desarrollo de esta memoria se encuentran:

Estructura de Datos, Base de Datos, Lenguajes de Programación, Introducción a la Ingeniería de Software, Ingeniería de Software, Taller de Producción de Software, Taller de Sistemas de Computación.

Para finalizar quiero agradecer a mi profesor guía, Don Carlos Castro, al DI en su conjunto y sus miembros y funcionarios, por otorgarme esta oportunidad de trabajo y por su preciado tiempo y disposición en el transcurso del desarrollo de esta memoria. El sistema fue hecho principalmente pensando en las encargadas de las Secretarías, para facilitarles su trabajo, lo que espero sinceramente que ocurra.

Capítulo 6: RECOMENDACIONES

Se recomienda, si el prototipo y potencial sistema posee el suficiente interés, implementarlo como una aplicación separada que se complete integrando servicios del DI ya implementados, como el sistema de usuarios del DI para la autenticación. Esta recomendación se basa en:

1. Considerando la situación y problemas de falta de personal para el mantenimiento y desarrollo presentes en el Data Center.
2. La decisión de emprender el desarrollo de una nueva Intranet, no aceptando nuevos cambios para la versión actual, congelando cualquier extensión inmediata.
3. Las recomendaciones y retroalimentaciones recibidas por parte de miembros del DI involucrados en el desarrollo y prueba del módulo
4. A pesar de depender el sistema de partes de la Infraestructura de la Intranet, de sus clases y controladores. Su simplicidad permite poder reconstruir la infraestructura necesaria sin mayores problemas. También, funcionalidad de autenticación se obtiene al integrarla con el sistema de autenticación y autorización del DI.
5. De otra entidad de la Intranet que el prototipo depende es de los Roles, pero al ser una aplicación especializada no se requieren todos los roles y permisos de la Intranet, por lo que también se podría recrear los roles y permisos necesarios sin mayores problemas.
6. En la retroalimentación de las pruebas se expresó que existe alto interés en continuar e implementar un Sistema de Gestión de Memorandos.

Por otro lado, respecto al prototipo y un futuro Sistema Gestor de Memorandos:

1. Se recomienda actualizar y rediseñar el proceso de redacción de Memorandos para permitir una idea de la estructura y formato en tiempo real, y no separar el completado del contenido con la apariencia en 2 pasos distintos. En otras palabras, crear una experiencia más cercana a MS-Word.
2. Se recomienda implementar y corregir todos los errores y sugerencias obtenidas de la prueba.
3. Se recomienda continuar su desarrollo e involucrar más personas e Unidades del DI para poder confeccionar una herramienta que facilite el trabajo y las comunicaciones de todo el DI. Además, entre más Unidades estén involucradas, se logrará concebir una mejor solución.

Capítulo 7:Anexos

7.1 Anexo A: Entrevistas

A continuación, se detallan las preguntas realizadas a las Secretarías de las unidades del DI y sus respuestas generales. A partir de estas, se entendió el contexto, y se detectó las oportunidades de mejora. A partir de estas entrevistas se confeccionó el resumen anterior y gran parte del diseño de la solución.

A continuación, se presenta una tabla con las preguntas y resumen de las distintas respuestas.

Tabla 7-1 Preguntas Entrevistas

Preguntas	Pregrado	Dirección	Postgrado	Infraestructura , Tecnología & Servicios	Vinculación con el Medio
¿Qué son los memos?	Comunicación formal multipropósito	Comunicación interna oficial validada por el director.	Comunicación interna específica de la unidad.	Comunicación interna para información o solicitudes.	Comunica-ción interna para solicitudes o información.
¿Para qué se usan?	Para enviar información formalmente dentro de la universidad, acciones relacionadas con estudiantes (ingresos, aceptación para asistir a curso, congelación, Autorización accesos a salas y en horarios especiales), envíos de pagos,	Pagos, Pagos de viáticos, transferencias de fondos, información general y validaciones.	Pagos, transferencias, acciones relacionadas con estudiantes (reincorporación, eliminación, congelación).	Solicitudes de transferencia, solicitudes de pago, información general.	Pagos, transferencias de fondos, gastos de cursos de postgrado, comunica-ción e información general.

	solicitudes de pagos				
¿Para qué se usan? - Explicación	Se respalda la comunicación de la información o solicitud de una acción. No es la acción en sí	Se respalda la comunicación de la información o solicitud de una acción. No es la acción en sí	Se respalda la comunicación de la información o solicitud de una acción. No es la acción en sí	Se respalda la comunicación de la información o solicitud de una acción. No es la acción en sí	Se respalda la comunicación de la información o solicitud de una acción. No es la acción en sí
¿Quiénes piden o emiten memos?	Jefes de Departamento (p.ej. jefe de carrera), director, vicerrectores y profesores.	Jefes de Departamento, subdirector de unidad.	Jefes de Departamento, subdirector de unidad.	Jefes de Departamento, subdirector de unidad. Jefes de la Unidad	Jefes de Departamento, subdirector de unidad.
¿Quiénes confeccionan memos?	Secretaría de Pregrado.	Secretaría de dirección y encargado de Apoyo Contable.	Secretaría de MTI.	Secretaría de I, T & S.	Secretarías de Vinculación con el Medio.
¿Qué datos componen un memo?	Número correlativo, remitente, destinatario, asunto, fecha, cuerpo del mensaje.	Número correlativo, departamento, unidad, año, fecha, remitente, destinatario, asunto.	Número correlativo, departamento, unidad, año, fecha, remitente, destinatario, asunto.	Número correlativo, departamento, unidad, año, fecha, remitente, destinatario, asunto	Número correlativo, departamento, unidad, año, fecha, remitente, destinatario, asunto.
¿Cómo se envían los memos?	Impresos y enviados en papel, a veces digitalmente.	Físicamente, a menudo con una firma física.	Impresos y a veces enviados digitalmente.	Impresos y a veces enviados digitalmente.	Impresos y a veces enviados digitalmente.
¿Quién gestiona el número correlativo?	La secretaría de Pregrado	La secretaría de Dirección	La secretaría MTI	La secretaría de I, T & S	La secretaría de Vinculación con el Medio

¿Dónde se almacenan los memos?	Máquinas locales de la secretaría.	Máquinas locales de la secretaría, pero compartidos y sincronizados con Apoyo Contable.	Máquinas locales de la secretaría.	Máquinas locales de la secretaría.	Máquinas locales de la secretaría.
¿Son los memos confidenciales?	No, pueden ser vistos por cualquiera que los solicite.	Confidenciales , típicamente accesibles solo por la secretaría, apoyo contable o la persona involucrada.	No se especifica.	No, cualquiera puede solicitar verlos.	No, pero deben ser solicitados por una persona conocida o una Emisor.
¿Cómo se transmiten los memos?	Físicamente y a veces digitalmente.	Físicamente, plegado sobre sí mismo o envuelto en un sobre si es confidencial.	Físicamente y a veces digitalmente.	Físicamente y a veces digitalmente.	Físicamente y a veces digitalmente.
¿Expiran los memos?	No en sí, pero el contenido puede indicar un evento temporal	No en sí, pero el contenido puede indicar un evento temporal	No en sí, pero el contenido puede indicar un evento temporal	No en sí, pero el contenido puede indicar un evento temporal	No en sí, pero el contenido puede indicar un evento temporal
¿Pueden tener adjuntos los memos?	Sí	Sí, respaldan solicitudes.	Sí	Sí	Sí
¿Existen categorías de memos?	Información o solicitudes de acción.	Plantillas varias dependiendo del propósito	Basado en el departamento , unidad o dirección del memo.	Depende del tipo de solicitud.	Plantillas estándar para ciertos tipos de memos.
¿Quién firma los memos?	La persona que lo solicitó, típicamente un Jefe de Departamento	Director del DI	Subdirector del MIT	Subdirector de I, T & S	Subdirector Vinculación con el Medio

	, Subdirector de Pregrado o un profesor.				
¿Quién puede ver los memos?	Cualquiera, sin restricciones de confidencialidad.	Cualquiera, sin restricciones de confidencialidad.	No especificado.	Cualquiera puede solicitar verlos.	No especificado.
¿Cuántos memos se emiten anualmente?	76 aprox.	80-100 aprox.	-	-	-

También, después de cada entrevista, se les solicitaba a los entrevistados muestras y ejemplos de Memorandos válidos emitidos

Capítulo 8: Bibliografía

- 37signals. (3 de Agosto de 2024). *Trix Editor*. Obtenido de Trix Editor Org: <https://trix-editor.org/>
- Action Controller Ruby Guide. (6 de Agosto de 2024). Obtenido de Guides Ruby On Rails: https://guides.rubyonrails.org/action_controller_overview.html
- Action View Ruby Guide. (6 de Agosto de 2024). Obtenido de Guides Ruby On Rails: https://guides.rubyonrails.org/action_view_overview.html
- Active Record Guide. (6 de Agosto de 2024). Obtenido de Guides Ruby On Rails: https://guides.rubyonrails.org/active_record_basics.html
- Active Storage - Ruby On Rails . (2 de Agosto de 2024). Obtenido de Ruby On Rails Guides v5.2: https://guides.rubyonrails.org/v5.2/active_storage_overview.html
- Basecamp-37signals. (3 de Agosto de 2024). *Trix*. Obtenido de Github Repository: <https://github.com/basecamp/trix>
- Coal. (17 de Mayo de 2023). *MVC Frameworks: A Historical Perspective on their Evolution*. Obtenido de Medium: <https://getcoal.medium.com/mvc-frameworks-a-historical-perspective-on-their-evolution-7b977cf47d77#:~:text=1990s,scalable%20and%20modular%20web%20applications>
- Company, T. S. (6 de Agosto de 2024). *TIOBE Index*. Obtenido de TIOBE - The Software Quality Company: <https://www.tiobe.com/tiobe-index/>
- Fox-Moss Rubydocs. (3 de Agosto de 2024). *Trix-Rails Gem Info*. Obtenido de Ruby Doc: <https://www.rubydoc.info/gems/trix-rails/2.4.0>
- Fox-Moss Rubygems. (3 de Agosto de 2024). *Tix-Rails Gem v2.4.0*. Obtenido de Rubygems: <https://rubygems.org/gems/trix-rails>
- <https://rubyonrails.org/>. (6 de Agosto de 2024). *Welcome*. Obtenido de Ruby On Rails: <https://rubyonrails.org/>
- Kulkarni-Truelsen. (3 de Agosto de 2024). *WK html TO pdf*. Obtenido de <https://wkhtmltopdf.org/>
- National Library of Norway - Archive-It. (25 de Abril de 2018). *Trygve M. H. Reenskaug*. Obtenido de Archive It: <https://wayback.archive-it.org/10370/20180425071111/http://folk.uio.no/trygver/themes/mvc/mvc-index.html>
- Rails Core. (5 de Agosto de 2024). *Ruby On Rails*. Obtenido de Ruby On Rails: <https://rubyonrails.org/>
- Rails-37signals. (3 de Agosto de 2024). *Ruby on Rails Framework*. Obtenido de Ruby On Rails Web Site: <https://rubyonrails.org/>
- Sam, R., & Dave, T. (2023). *Agile Web Development With Rails 7* (págs. xi-xvii). The Pragmatic Programmers.
- Sam, R., & Dave, T. (2023). *Agile Web Development With Rails 7*. The Pragmatic Programmers.
- Sam, R., & Dave, T. (2023). Capítulo - The Architecture of Rails Applications - Agile Web Development With Rails 7. En S. Ruby, & D. Thomas, *Agile Web Development With Rails 7* (págs. 37-42). The Pragmatic Programmers.
- Simon, R., & Dave, T. (2023). *Agile Web Development With Rails*. En R. Simon, & T. Dave, *Original MVC intent - Agile Web Development With Rails* (pág. 39). The Pragmatic Programmers.
- Sterrett, M. Z. (3 de Agosto de 2024). *Wicked PDF Ruby On Rails Plugin*. Obtenido de Github Repository: https://github.com/mileszs/wicked_pdf
- The Apache Software Foundation. (1 de marzo de 2024). *Struts*. Obtenido de weblegacy github.io: <https://weblegacy.github.io/struts1/>

Wilf, L. R., & John, P. R. (1990). Inside Smalltalk Volume II. En L. R. Wilf, & P. R. John, *Lalonde R. Wilf; Pugh R. John* (págs. 7-12). Englewood Cliffs: Prentice-Hall International Inc.