UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Sistema d'informació per a la creació i gestió d'automatitzacions d'operativa bancaria

Memòria

Autor:

ARNAU SANTAMARIA PIÑOL

Grau en Enginyeria Informàtica

Especialitat en Sistemes d'Informació

Facultat d'Informàtica de Barcelona

Universitat Politècnica de Catalunya

Director:

LOURDES GISSEL ARELLANO DA SILVA

everis, an NTT DATA Company

Ponent:

Joan Antoni Pastor Collado

Departament de Sistemes d'Informació

Agraïments

A la meva tutora, Lourdes Gissel Arellano, que m'ha ajudat en tot el que m'ha fet falta des de que he entrat a fer pràctiques a everis NTT Data Company i a l'equip amb qui treballo cada dia, que em fan sentir molt còmode i m'han ajudat en moments de dubtes del projecte en concret a Oleksandr Vynokur i Àlex Moll i altre cop a la Lourdes, amb qui a part de ser companys d'equip, he creat una relació d'amistat que aprecio molt.

Resum

Actualment les automatitzacions de les tasques i models de negoci del sector bancari s'han transformat com conseqüència d'utilitzar noves eines i solucions tecnològiques. Gràcies a això es preveu un augment de la productivitat en el sector en més d'un 50%. [R1]

En aquest context, everis NTT Data Company i concretament everisBPO (Business Process Outsourcing) Banking, està apostant per la transformació de processos operatius desenvolupant eines que impliqui l'ús d'automatitzacions.

La finalitat d'aquest Treball de Final de Grau (TFG) serà la construcció d'un sistema d'informació basat en un framework, anomenat Saturnus, que proporcioni les eines necessàries per a la creació, control i manteniment d'automatitzacions de processos d'operativa bancaria de manera rapida i eficaç i un panell de control per a gestionar les automatitzacions creades.

Abstract

At present, the automation of the tasks and business models of the banking sector have been transformed as a result of using new tools and technological solutions. Thanks to this, an increase in productivity in the sector is expected by more than 50%. [R1]

In this context, everis NTT Data Company and specifically everisBPO (Business Process Outsourcing) Banking, is betting on the transformation of operating processes by developing tools that involve the use of automation.

The purpose of this Final Project of Degree (TFG) will be the construction of an information system, based on a framework, called Saturnus, which will provide the necessary tools for the creation, control and maintenance of banking automation processes operations quickly and efficiently and a control panel to manage the automations created.

Resumen

Actualmente las automatizaciones de las tareas y modelos de negocio del sector bancario se han transformado como consecuencia de utilizar nuevas herramientas y soluciones tecnológicas. Gracias a ello se prevé un aumento de la productividad en el sector en más de un 50%. [R1]

En este contexto, everis NTT Data Company y concretamente everisBPO (Business Process Outsourcing) Banking, está apostando por la transformación de procesos operativos desarrollando herramientas que impliquen el uso de automatizaciones.

La finalidad de este Trabajo de Fin de Grado (TFG) será la construcción de un sistema de información basado en un framework, llamado Saturnus, que proporcione las herramientas necesarias para la creación, control y mantenimiento de automatizaciones de procesos de operativa bancaria de manera rápida y eficaz y un panel de control para gestionar las automatizaciones creadas

Taula de continguts

1. Introducció

1.1 Formulació del problema

Des de fa dècades l'esser humà i les màquines conviuen a l'àmbit laboral. Al llarg del temps diferents "revolucions" van ocasionar canvis a l'hora de la forma de fer una tasca al lloc de treball. La primera revolució va ser l'industrial amb la maquina de vapor. La segona va ser la producció en massa i l'electricitat. Una tercera relacionada amb l'aparició dels ordinadors i Internet. En aquests dies es parla que estem vivint el que els economistes li diuen "la quarta revolució industrial" que serà una interacció de canvis de tecnologies varies (digital, físiques, biològiques). [R3]

Actualment les automatitzacions de les tasques i models de negoci del sector bancari s'han transformat com conseqüència d'utilitzar noves eines i solucions tecnològiques. Gràcies a això es preveu un augment de la productivitat en el sector en més d'un 50%. [R1]

En aquest context, everis NTT Data Company i concretament everisBPO (Business Process Outsourcing) Banking, està apostant per la transformació de processos operatius desenvolupant eines que impliqui l'ús d'automatitzacions.

A everis BPO Banking porto més d'un any treballant com becari a l'Oficina de Transformació participant en diferents tasques des de la creació d'aquesta oficina. El sector d'everis BPO al que treballo porta l'externalització de l'operativa d'una entitat bancaria de prestigi a nivell nacional i internacional amb més de 300 persones treballant cada dia amb un volum d'operacions bancaries de 1000. Una vegada que l'activitat traspassada està estable s'ha observat que, dins els diferents serveis que presten, existeixen molts processos susceptibles a ser automatitzats de forma parcial o total per a millorar la productivitat i qualitat.

Quan parlem d'automatització ens referim a processos en què un software podria fer la mateixa feina que una persona, de manera que aquesta persona pugui ser redistribuïda a un altre lloc i així poder millorar l'eficiència i la productivitat de diferents serveis del projecte. La gran majoria d'aquestes automatitzacions s'han de realitzar sobre una aplicació software que proporciona l'entitat financera anomenat SoftBan (nom fictici).

1.2 Proposta de sol·lució

La finalitat d'aquest Treball de Final de Grau (TFG) serà la construcció d'un sistema d'informació basat en un framework, anomenat Saturnus, que proporcioni les eines necessàries per a la creació, control i manteniment d'automatitzacions de processos d'operativa bancaria de manera rapida i eficaç i un panell de control per a gestionar les automatitzacions creades.

Un framework es una estructura conceptual i tecnològica de suport definit, habitualment, amb mòduls concrets de software, que poden servir de base per la organització i desenvolupament d'un software. [R2]

S'han definit dos objectius principals que formaran part del conjunt del sistema:

- Desenvolupament d'un framework base per a la creació d'automatitzacions

El framework Saturnus serà la base per desenvolupar automatitzacions. Un programador utilitzant aquest framework serà capaç de crear automatitzacions a través de captura d'imatges en pantalla de manera fàcil, gràcies als mètodes que proporcionarà per a poder controlar el ratolí, el teclat, la pantalla i tot el necessari per a poder crear una automatització. aconseguir que un programador

- Desenvolupament d'un panell de control d'automatitzacions

Un cop creades les automatitzacions serà necessari controlar diferents paràmetres en tot moment. Per tal finalitat, es desenvoluparà un panell de control via web a través del qual es podran configurar, iniciar i parar les automatitzacions. A més permetrà visualitzar les automatitzacions que estan en funcionament, en quin estat es troben, consultar la càrrega de treball que han realitzat fins ara i altres opcions.

1.3 Context

En aquest apartat s'analitza el context on se situarà el sistema un cop finalitzat el projecte. Concretament, es determinen quines seran les parts interessades a les que va dirigit, quin us tindrà del sistema i quins beneficis aportarà.

1.3.1 Parts interessades

A continuació, es descriuen quines parts interessades estan implicades en el projecte que es desenvoluparà i quin us faran del sistema:

<u>everis BPO Banking:</u>

everis BPO Banking és l'empresa per la qual es desenvolupa el projecte, per el que exerceix el rol de proveïdor de processos per a que siguin automatitzats. Aquesta organització realitza diferents serveis per a l'entitat financera, com per exemple:

- Servei d'Alta d'Expedients de Risc
- Servei de Formalització d'Actiu
- Servei d'Obertura de Comptes i Clients
- Testamenteries
- Comptabilització de Factures
- Altres

Dins d'aquests serveis hi ha processos que poden ser automatitzats de forma total o parcial, per tant l'empresa també exercirà el rol d'usuari de les automatitzacions creades. La interacció amb les automatitzacions sempre es farà a través del panell de control.

<u>Entitat financera:</u>

L'entitat financera exerceix el rol de client d'everisBPO, l'empresa que dura a terme totes les tasques es van externalitzar per a l'entitat.
El grau d'influència de l'entitat és molt alt perquè per realitzar les automatitzacions, abans de començar el desenvolupament, ha d'aprovar un procés d'avaluació exhaustiu.

<u>Empleats del sector everis BPO implicat</u>

El conjunt d'empleats que es veuran afectats pel sistema i en faran ús es divideixen de la següent manera:

- <u>Empleats que realitzen operativa bancaria</u>
  Tot el conjunt de treballadors que fan l'operativa d'aquest sector d'everisBPO Banking són usuaris potencials de les diferents automatitzacions.
  Els coordinadors seran els responsables de fer la formació d'aquestes persones perquè en alguns processos s'hauria de canviar la forma de fer la feina.

- Desenvolupadors d'automatitzacions

Els programadors que desenvolupen les automatitzacions ho hauran de fer a partir del framework base Saturnus que incorpora el sistema. Utilitzaran el framework en l'àmbit d'usuari i els ajudarà a desenvolupar de manera més rapida i senzilla les noves automatitzacions.

- Coordinadors dels serveis

Els coordinadors seran els responsables de fer la formació dels seus equips a l'hora de canviar la forma de fer les tasques fins ara totalment manuals amb l'arribada de l'automatització parcial o total.

A més seran els que facin ús del panell de control ja sigui configurant els paràmetres, obtenint informes, etc.

- Directius del projecte

Per descomptat els directius del projecte estan involucrats des del minut cero d'aquesta iniciativa. Es veuran beneficiats de les automatitzacions creades indirectament des del punt de vista de millor productivitat i eficiència dels processos realitzats.

2. Estat de l'art

Entre els objectius del projecte s'ha de desenvolupar un framework i un panell de control que permeti crear als desenvolupadors automatitzacions a través d'imatges en pantalla de manera fàcil i rapida. A continuació s'ha realitzat una anàlisis de diferents tecnologies que permeten crear automatitzacions a partir del reconeixement d'imatges en pantalla, per exposar les seves carències i poder construir un producte més complet. Per a la creació del panell de control no s'ha realitzat un estudi previ perquè és vol fer a mida per a l'empresa.

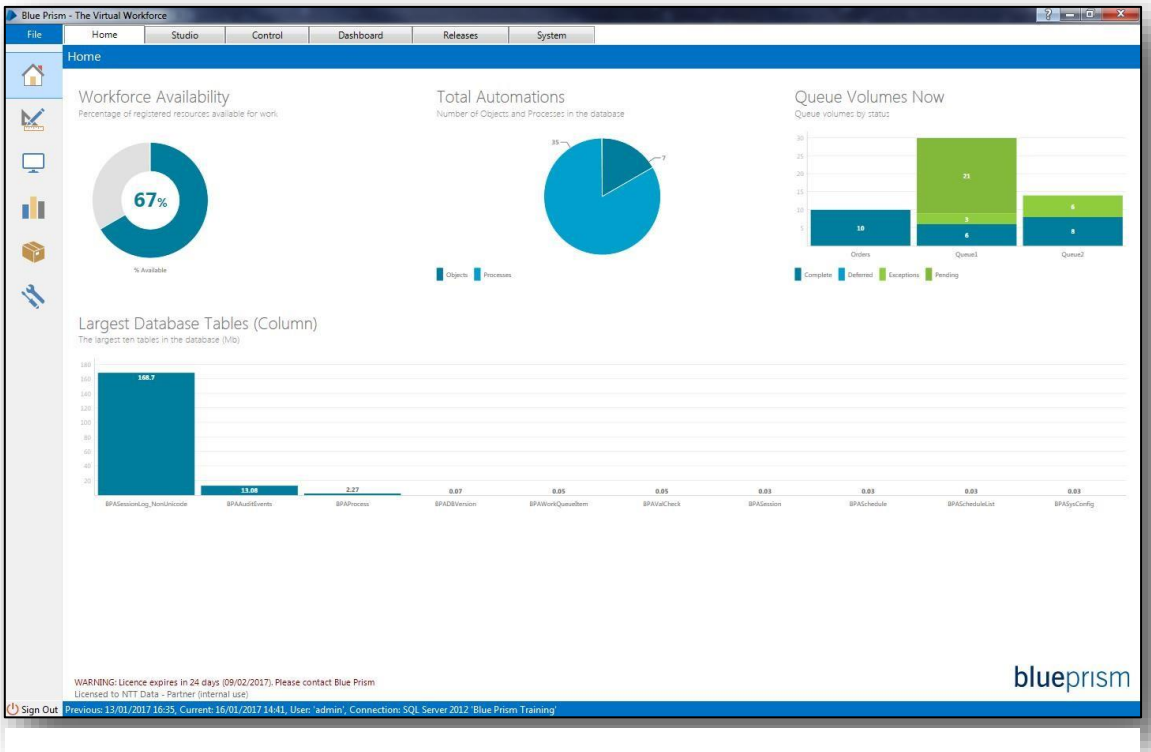
2.1 Software de creació d'automatitzacions

S'ha seleccionat l'única alternativa real per a l'empresa i una segona alternativa que es va provar i descartar pràcticament d'immediat, que és un software de creació d'automatitzacions reconegut en el mercat. Actualment diferents projectes d'everis estan fent proves amb aquest programari.

2.1.1 Blue Prism

Blue Prism es defineix com l'empresa que va inventar el terme robòtica d'automatització de processos. Aquesta plataforma de programari permet crear automatitzacions de les operacions de negoci. [R4]

Blue Prism te múltiples seccions que permeten realitzar no tan sols automatitzacions sinó que també ofereixen la possibilitat d'exercir un control i seguiment sobre aquestes. Tot a través d'una interfície intuïtiva i usable.



Il·lustració 1: Finestra Home Blue Prism

En la il·lustració 1 podem veure el punt d'entrada a l'aplicació i que mostra les estadístiques de les activitats recents. És pot configurar completament i permet tenir les dades que més interessin. En aquest cas d'exemple podem veure gràfics d'activitat.

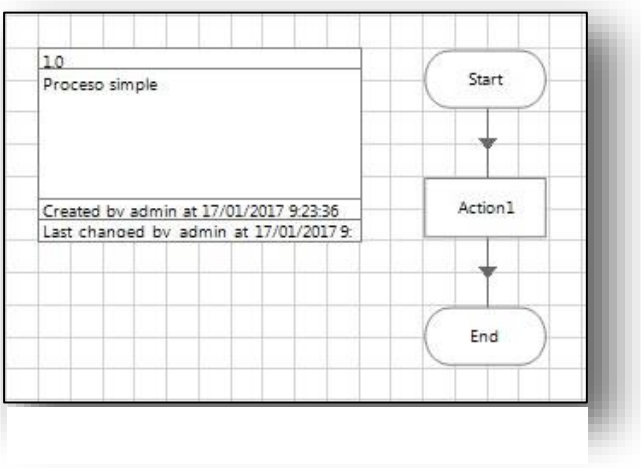
Al Blue Prism existeixen diferents eines per a la creació d'automatitzacions però és poden classificar en dues principals:

- Processos: Incorporen la lògica de negoci de l'aplicació. Aquests instancien objectes per a executar les funcionalitats desitjades.
- Objectes: Els objectes són funcionalitats comuns en diferents automatitzacions i se separen per a ser més re-usables.

Per exemple, en el cas que volguéssim automatitzar una operativa que consistís en enviar un Excel per correu electrònic, els objectes tindrien les funcionalitats d'obrir, tancar i enviar el correu, mentre que el procés indicaria els passos a seguir.

2.1.1.1 Processos

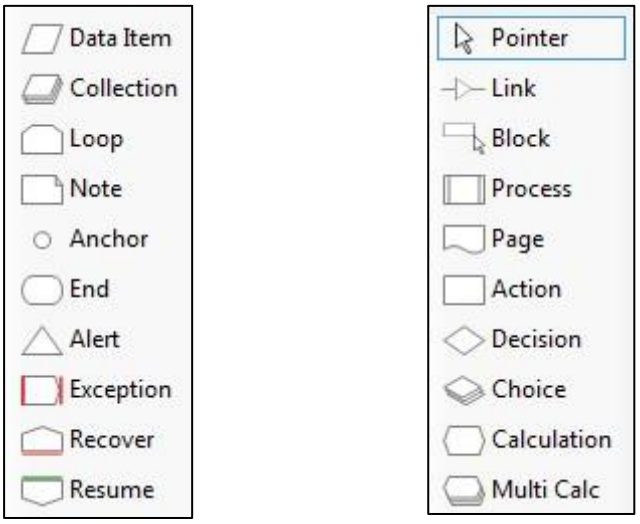
Els processos en Blue Prism tenen un format molt semblant al d'un diagrama de flux. Per a crear un procés s'utilitza simbologia i notacions de diagrames de flux estàndard.



Il·lustració 3: Diagrama de flux d'un procés

En la il·lustració 3 podem com es veu un procés simple, que sempre ha d'estar format per un element Start(Inici) i un End(Final) i una o més accions.

Per a poder crear tot tipus de diagrames de fluxos i processos Blue Prism un conjunt d'eines que permetran la creació de la lògica a seguir.



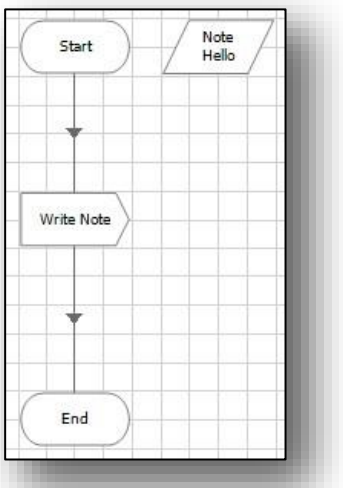
Il·lustració 4: Conjunt d'eines per a crear processos

2.1.1.2 Objectes

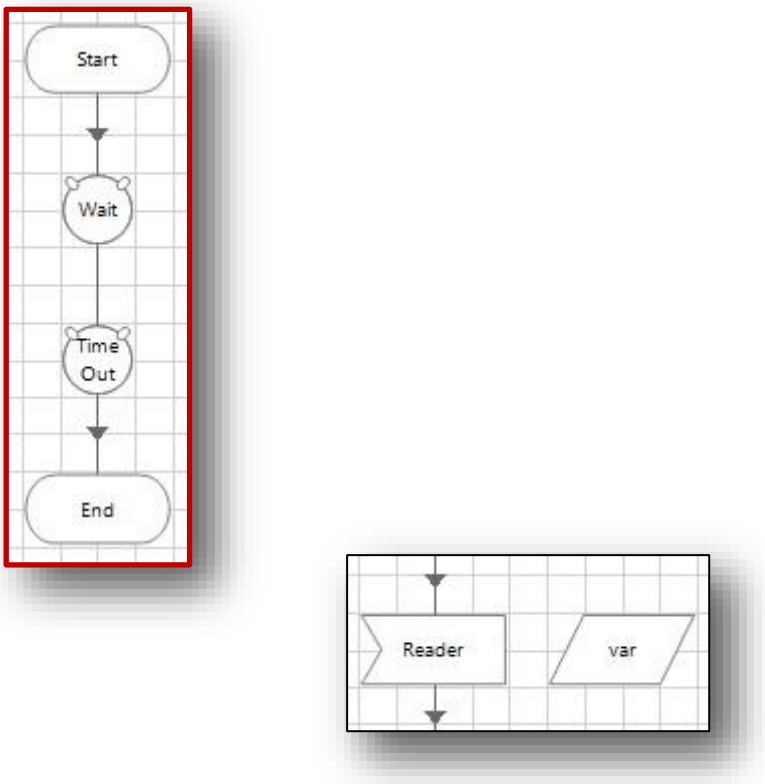
Els objectes són l'altre eina principal utilitzada per a automatitzar amb Blue Prism. Aquests ens permeten crear funcionalitats de nivell més baix. Principalment els objectes tenen la funció de ser instanciats des d'accions dins dels processos.

En la il·lustració 5 es mostra el diagrama de flux d'un objecte. Tots els objectes han de tindre un Start (Inici, constructor de l'objecte) i un End (Final, destructor de l'objecte).

Igual que en la creació de processos Blue Prism proporciona un conjunt d'eines per a poder crear Objectes.



Il·lustració 5: Diagrama de flux d'un objecte



Il·lustració 6: Eines per a la creació d'objectes.

La il·lustració 6 mostra diferents eines que es poden utilitzar per a crear objectes, com poden ser una funcionalitat per esperar cert temps (wait), una funcionalitat que et permeti llegir un camp o navegar a un altre camp. Blue Prism incorpora diferents eines útils per a crear objectes.

2.1.2 Conclusions

Del software analitzat és pot concloure que està capacitat per a realitzar automatitzacions però després d'estar provant Blue Prism i parlant amb persones més experimentades i amb el director del projecte s'han trobat les següents limitacions:

- Preu del software elevat: El preu del software és molt elevat i el projecte BPO no seria capaç de comprar-lo. En el cas que es volgués adquirir l'hauria de comprar everis de manera global.
- Corba d'aprenentatge elevada: La corba d'aprenentatge del programari és complexa i per tant la creació d'automatitzacions no la podria fer qualsevol desenvolupador en poc temps. Aquest motiu és un inconvenient molt gran doncs el que es busca és que les automatitzacions es puguin crear en poc temps i sense ser un desenvolupador experimentat amb aquest tipus de desenvolupaments.

2.2 Tecnologies candidates per a crear automatitzacions

Actualment existeixen diferents tecnologies que permetrien crear el framework i desenvolupar el panell de control. A continuació es mostren les tecnologies que es van valorar i quina va ser finalment la tecnologia escollida.

2.2.1 Llibreries per a la detecció d'imatges en pantalla

2.2.1.1 Open CV

OpenCV (Open Source Computer Vision Library) es distribueix sota una llicència BSD i per tant es lliure tant per a ús acadèmic com per a ús comercial. Té interfícies de C ++, C, Python i Java i és compatible amb Windows, Linux, Mac OS, iOS i Android.

OpenCV va ser dissenyat per a l'eficiència computacional i amb un fort enfocament en aplicacions en temps real. Escrit en C / C ++, la biblioteca pot prendre avantatge de processament multi-nucli.

Adoptat a tot el món, OpenCV té més de 47 mil persones de la comunitat d'usuaris i el nombre estimat de descàrregues superiors a 14 milions. Es utilitzant tant a l'art interactiu, com a la inspecció de mines i també en robòtica avançada. [R5]

2.2.1.2 Emgu CV

Emgu CV és una contenidor per a la plataforma .NET de la biblioteca de processament d'imatges OpenCV. El que permet és fer crides de OpenCV des de llenguatges compatibles amb .NET com C #, VB, VC ++, IronPython etc. El contenidor pot ser compilat per Visual Studio, Xamarin Studio and Unity, que pot funcionar en Windows, Linux, Mac OS X, iOS, Android i Windows Phone. [R6]

2.2.1.3 SikuliX

SikuliX automatitza qualsevol cosa que vegi en la pantalla del seu ordinador d'escriptori amb Windows, Mac o algun Linux / Unix. Utilitza el reconeixement d'imatges impulsat per OpenCV per identificar i controlar els components de la GUI. Això és útil en els casos en què no hi ha fàcil accés als components interns d'una interfície gràfica d'usuari o el codi font de la pàgina o aplicació web que desitja actuar.

A més SikuliX es pot usar en la programació Java i la programació / scripting amb qualsevol llenguatge de programació / scripting basat en Java (Jython, JRuby, Scala, Clojure, ...). [R7]

2.2.2 Llenguatges de programació candidats

2.2.2.1 C#

C# és un llenguatge de programació orientat a objectes desenvolupat per Microsoft i estandarditzat, com a part de la seva plataforma .NET.

La seva sintaxi bàsica deriva de C/C++ i utilitza el model d'objectes de la plataforma .NET el qual és similar al de Java però inclou millores derivades d'altres llenguatges. C# fou dissenyat per a combinar el control a nivell baix de llenguatges com C i la velocitat de programació de llenguatges com Visual Basic. (Font: Wikipedia)

Per al reconeixement d'imatges en pantalla C# disposa una llibre anomenada EMGU CV, una adaptació Open CV per a C#.

2.2.2.2 Java

El Java és un llenguatge de programació dissenyat el 1990 per James Gosling amb altres companys de Sun Microsystems a partir de C. Des del seu naixement fou pensat com un llenguatge orientat a objectes. Entre el 13 de novembre de 2006 i el maig del 2007 Sun va alliberar parts de Java com a programari lliure de codi obert amb llicència GPL. És un dels llenguatges de programació més utilitzats, i s'utilitza tant per aplicacions web com per aplicacions d'escriptori.

El Java és un llenguatge interpretat i, per tant, pot semblar lent en comparació amb altres llenguatges, però ofereix un índex de re-utilització de codi molt elevat, sent possible trobar moltes llibreries lliures de Java. És un llenguatge flexible i potent tot i la facilitat amb la qual es programa i dels resultats que ofereix. Un dels trets que el caracteritza i que el fa una eina molt valorada a l'hora de desenvolupar aplicacions distribuïdes, és el fet que és un llenguatge multi-plataforma. [R8]

2.3 Tecnologies candidates per a la creació del panell de control

Com que el panell de control estava creat expressament per al projecte és va decidir escollir les següents tecnologies per a desenvolupar-ho ja que aquestes ja les coneixia, degut a que vaig desenvolupar l'intranet del projecte BPO amb aquesta mateixa tecnologia.

2.3.1 Tecnologies front end

Per a desenvolupar la part visual del projecte, la part del client, es van analitzar i escollir les tecnologies que s'utilitzen habitualment en el desenvolupament d'aquest tipus de projecte.

2.3.1.1 HTML 5 + CSS 3

HTML (acrònim d'Hyper Text Markup Language, en català, "llenguatge de marcat d'hipertext"), és un llenguatge de marcat que deriva de l'SGML dissenyat per estructurar textos i relacionar-los en forma d'hipertext. Gràcies a Internet i als navegadors web, s'ha convertit en un dels formats més populars que existeixen per a la construcció de documents per a la web.

Cascading Style Sheets (CSS, en català: Fulls d'Estil en Cascada) és un llenguatge de fulls d'estil utilitzat per descriure la semàntica de presentació (l'aspecte i format) d'un document escrit en un llenguatge de marques. La seva aplicació més comuna és dissenyar pàgines web escrites en HTML i XHTML, però el llenguatge també pot ser aplicat a qualsevol classe de document XML, incloent-hi SVG i XUL.

CSS està dissenyat principalment per permetre la separació de contingut del document (escrit en HTML o un llenguatge de marques similar) de la presentació del document, incloent-hi elements com la disposició, colors, i fonts. Aquesta separació pot millorar l'accessibilitat al contingut, proporcionar més flexibilitat i control en l'especificació de característiques de presentació, permetre que múltiples pàgines comparteixin un format comú, i redueix complexitat i repetició en el contingut estructural (com per exemple al permetre disseny web sense taules). CSS també pot deixar la mateixa pàgina de marques ser presentada en estils diferents mitjançant mètodes de render diferents, com a la pantalla, en impressió, per veu (quan és llegida en veu alta per un navegador amb lector o pantalla lectora) i amb mecanismes tàctils amb sistemes Braille. Mentre que l'autor d'un document típicament associa els documents amb un full d'estil CSS, els lectors poden utilitzar un full d'estil diferent, potser un al seu propi ordinador, per invalidar aquell que l'autor ha especificat. [R9]

2.3.1.2 Angular JavaScript

Angular JS és un framework de JavaScript de codi obert, mantingut per Google, que s'utilitza per a crear i mantenir aplicacions web d'una sola pàgina. La biblioteca llegeix HTML, que conte atributs de les etiquetes personalitzades addicionals, i llavors obeeix a les directives d'aquests atributs i uneix les peces d'entrada o de sortida de la pàgina a un model representat per les variables estàndards de JavaScript. [R10]

S'utilitza per a mostrar un contingut dinàmic a la web.

2.3.2 Tecnologies del Back-End

Com que el panell de control estava creat expressament per al projecte és va decidir escollir les següents tecnologies per a desenvolupar-ho ja que aquestes ja les coneixia, degut a que vaig desenvolupar l'intranet del projecte BPO amb aquesta mateixa tecnologia.

2.3.2.1 Java Enterprise Edition

Java Platform, Enterprise Edition o Java EE (va ser conegut com a Java 2 Platform Enterprise Edition o J2EE fins a la versió 1.4), és una plataforma de programació (una de les Plataformes Java) per desenvolupar i executar programari escrit amb el llenguatge Java amb una arquitectura distribuïda amb nivells, basada en components de programari, tot plegat executant-se en un servidor d'aplicacions.

Es pot utilitzar com a servidor per a inserir o extreure informació de les bases de dades o conjuntament amb la part de client per a llançar una web completa (front-end + back-end).

2.3.2.2 Arquitectura REST

REST (Representational State Transfer) és una arquitectura de programari pensada per sistemes distribuïts basats en hipermèdia, com ara el web. Aquest terme va ser introduït l'any 2000 en una tesi doctoral sobre arquitectures de programari de xarxes.[1] Aquesta tesi va ser escrita per Roy Thomas Fielding i dirigida per Richard N. Taylor. Roy va ser un dels principals autors de les especificacions del protocol HTTP i explica en la seva tesi com es pot aprofitar aquest protocol per tal de desenvolupar aplicacions distribuïdes. Tot i que en un principi REST es referia tan sols a un conjunt de principis d'arquitectura de xarxa i la definició i adreçament dels recursos, actualment aquest concepte es fa servir per referir-se a una interfície web que utilitza XML, JSON, HTML i HTTP sense cap conjunt de capçaleres com podria ser en el cas de SOAP i XML-RPC. Segons la tesi de Roy es poden dissenyar interfícies XML+HTTP seguint la filosofia de Remote Procedure Call sense utilitzar la complexitat del protocol SOAP.

Un servei web REST és un model centrat en les dades. Els anomenats recursos vénen identificats per URIs i poden ser manipulats mitjançant accions especificades a les capçaleres HTTP. [R11]

3. Metodologia i rigor

3.1 Mètodes de treball

La metodologia utilitzada per aquest projecte és una metodologia en cascada amb modificacions. Es dividirà el procés en tres etapes i totes elles segueixen un procés seqüencial, tot i que a la segona etapa adoptarem un format iteratiu per ser que al ser més àgil permetrà el disseny, implementació i validació de les funcionalitats en cada iteració.

Hi haurà una infraestructura de tres entorns; desenvolupament, on es realitzaran totes les proves del framework, del panell de control i de les automatitzacions creades; pre-producció, entorn que proporciona el banc per a poder desenvolupar les automatitzacions en un entorn fidedigne però sense dades reals; i finalment l'entorn de producció que consta de l'aplicació del banc en la que treballaran les automatitzacions alhora que els agents del projecte.

3.1 Eines de seguiment

Per a fer un seguiment de les tasques creades i veure com va evolucionant el projecte utilitzarem un software lliure anomenat Gantter que permet crear diagrames de Gantt de manera senzilla i intuïtiva.

A part s'han realitzat reunions diàries amb la tutora del projecte per a realitzar un seguiment de les tasques del dia, veure si hi ha hagut problemes en alguna tasca i canviar la planificació si fos necessari.

3.2 Eines de validació

Per tal de validar la correcta execució del projecte es desenvoluparà una automatització necessària basada en el framework desenvolupat i utilitzant el panell de control per a la seva gestió. Aquesta automatització ha estat pactada amb el Banc i el seu comportament simula el d'un agent normal del projecte.

A través d'aquesta automatització s'ha pogut comprovar que la implementació del framework sigui correcte i que la gestió s'hagi pogut realitzar correctament a través del panell de control.

4. Abast del projecte

L'abast d'aquest projecte és definir, dissenyar i implementar un sistema que permeti la creació i gestió d'automatitzacions de l'empresa de manera rapida i senzilla.

El que es vol aconseguir és construir un sistema en què a partir d'ell es creïn noves automatitzacions funcionals i rapides en poc temps, però per a realitzar un sistema d'aquestes dimensions es requereix molt de temps i per tant hem hagut d'aplicar les següents limitacions:

- **Desenvolupament del framework base sobre un únic llenguatge de programació, Java**

  Si fos possible el framework base s'hauria de desenvolupar en diferents llenguatges de programació per així adaptar-se a totes les possibilitats que poguessin sortir en el futur, ja que potser certes automatitzacions són més fàcils de realitzar en altres llenguatges de programació. Però degut a diferents obstacles que ens hem trobat en una anàlisis previ i que veurem en posteriors apartats, es desenvoluparà només en Java.

- **Posada en funcionament de funcionalitats bàsiques**

  En els objectius presentats anteriorment s'ha mencionat de forma resumida les funcions bàsiques que tindrà el sistema i que es desenvoluparan. Hi ha altres funcionalitats interessants com l'extracció i anàlisis de dades generades per les automatitzacions que no es desenvoluparan degut a la limitació temporal del projecte.

- **Execució de les automatitzacions en una granja de servidors/ordinadors**

  L'execució de les automatitzacions s'han de realitzar en una granja de servidors a causa de diferents obstacles que veurem en el següent apartat. Per aquest motiu es construirà una granja de servidors on cada entorn executarà una automatització.
  Degut a que el projecte everis BPO no te una infraestructura informàtica a aquest nivell, els servidors es contractaran de manera interna al departament d'infraestructures d'everis i seran proporcionats i administrats en ultima instància per ells. Per tant la construcció d'aquestes granges no ha pogut formar part del projecte de final de grau.

Tal com s'ha explicat abans, l'objectiu és crear un sistema que permeti desenvolupar automatitzacions de manera ràpida i amb poc temps, i a part proporcionar un panell de control des d'on poder controlar i administrar totes les automatitzacions.

4.1 Fases previstes

Tal i com hem mencionat anteriorment, el projecte es dividirà en tres grans fases, que posteriorment seran subdividides en tasques més concretes.

Les tres fases en que es dividirà el projecte són:

- Fase 1: Anàlisis de requisits i preparació/gestió del projecte.
- Fase 2: Disseny del sistema, implementació i validació.
- Fase 3: Posada en marxa i últimes validacions.

En la primera fase es vol realitzar l'anàlisi de requisits del framework base que es vol desenvolupar i del panell de control, per a saber exactament quines son les funcionalitats que han d'incorporar el projecte.

En la segona fase es dissenyarà el framework i el panell de control i les bases de dades. S'implementaran les funcionalitats.

En la tercera fase és farà el pas a producció del sistema d'informació, pujant el panell de control al servidor utilitzat i posant a disposició dels programadors del projecte el framework per a que qualsevol programador pugui usar-lo per a desenvolupar automatitzacions.

4.2 Possibles obstacles

Les automatitzacions s'han de crear a partir de reconeixement d'imatges en pantalla.

El programari que proporciona l'entitat financera, anomenat SoftBan, que és sobre el qual es realitzaran gran part de les automatitzacions és una aplicació d'escriptori i no tenim accés al seu codi font.

Actualment existeixen tecnologies que permeten crear automatitzacions basats en el reconeixement d'objectes (ja siguin camps per introduir dades, imatges, etc) a través de les seves propietats (sobretot en aplicacions web).

Com no tenim accés al codi font les automatitzacions s'hauran de crear a partir de reconeixement d'imatges en pantalla i simulant el moviment de ratolí i el teclat. Això pot comportar problemes sobretot a l'hora de comparar imatges, ja que hi influiran factors com la resolució de la pantalla o la posició i dimensió de la imatge.

SoftBan s'ha desenvolupat en Java.

El programari que proporciona l'entitat financera, anomenat SoftBan, que és sobre el qual es realitzaran una gran part de les automatitzacions, s'ha desenvolupat en Java. Aquest fet descarta altres llenguatges de programació com C# ja que funcions de teclat com les de tabular, pitjar les fletxes, enter, etc no funcionen sobre aquesta aplicació, i aquestes són bàsiques per crear automatitzacions.

No es poden executar dues simulacions alhora.

Com que les automatitzacions és crearan a partir del reconeixement d'imatges i simulant el moviment del ratolí i el teclat, no és podran executar dues automatitzacions alhora, ja que interferirien entre elles i no funcionaria correctament. Per això s'instal·laran en la granja de servidors explicada prèviament

4.3 Fases establertes

Les fases establertes han sigut les mateixes que es van preveure amb una petita modificació que explicaré a continuació.

En la segona fase és va preveure dissenyar i implementar el framework i el panell de control, però s'ha afegit una tercera implementació a realitzar, una automatització d'una operativa del projecte que servirà per a realitzar la validació del correcte funcionament del framework i panell de control dissenyats prèviament.

Les fases establertes finals són:

- **Fase 1: Anàlisis previ**

Es realitzarà un estudi de l'estat de l'art de les aplicacions similars ja existents, i una visió global del sistema amb totes les funcionalitats que ha d'incorporar, de les quals algunes seran descartades per la limitació temporal del projecte.

- **Fase 2: Gestió de projectes**

Realització de l'assignatura Gestió de Projectes.

- **Fase 3: Anàlisis de requisits**

En aquesta fase es defineixen les parts interessades del sistema, s'estableixen els requisits no funcionals, les propietats i les hipòtesis del domini i les restriccions del projecte.

- **Fase 4: Especificació**

Descripció dels casos d'ús i realització de l'esquema conceptual i el model de comportament del sistema.

- **Fase 5: Disseny, implementació i proves.**

Respecte al disseny, es definirà l'arquitectura del sistema, la distribució i funció de cada capa, , el diagrama de classes i l'estructura de la base de dades.

La implementació es dividirà en tres parts, per un costat, s'implementarà el framework i un cop acabada es desenvoluparà el panell de control web. En les dues tasques la construcció serà iterativa, fent proves al final de cada iteració.

Finalment es desenvoluparà una automatització a través de la qual es validarà el funcionament conjunt del panell de control i el framework.

- **Fase 6: Redacció de la memòria del treball.**

Finalment, es revisarà tot el treball, es redactaran les conclusions i es mostraran les desviacions de la planificació inicial.

5. Objectius del projecte

En aquest projecte s'han assolit els següents objectius:

Objectiu 1: Desenvolupar un *framework* base per a la crear automatitzacions

Per tal de que qualsevol programador pugui crear automatitzacions de manera rapida i eficaç s'ha de desenvolupar un framework que proporcioni les eines necessàries per fer-ho.

Aquestes eines són les següents:

- *Control d'escriptori*: Funcions que permetin simular l'ús del teclat o el ratolí per a interaccionar amb les imatges que surtin en pantalla.
- *Reconeixement d'imatge en pantalla*: Per a poder navegar entre les aplicacions del banc s'haurà de construir unes funcionalitats que permetin buscar imatges en pantalla, en regions concretes de la pantalla, etc.
- *Connexió amb la base de dades*: Funcions que permetran connectar amb les bases de dades per a poder manipula la informació necessària.
- *Traces d'error*: Conjunt de funcions que permetran gestionar les tracés d'error de manera senzilla.
- *Utilitats:* Funcions útils que seran comunes en la majoria d'automatitzacions i que permetran estalviar temps a l'usuari que hagi de desenvolupar-les ja que estaran fetes per defecte.

Objectiu 2: Desenvolupar un panell de control d'automatitzacions web

Per a poder gestionar les automatitzacions que s'han creat es necessità una eina que et permeti fer-ho. Per aquest motiu s'ha desenvolupat un panell de control web que permet realitzar diferents accions i obtindré informació sobre les automatitzacions que estan en funcionament.

- Control de les automatitzacions: Configurar, iniciar, parar i reiniciar una automatització.
- Monitorització de les automatitzacions: Veure en quin estat estan les automatitzacions, quines són les ultimes accions que han realitzat o si han sorgit errors i permetre generar petits informes d'activitats de cada automatització.

Objectiu 3: Desenvolupar una automatització d'un procés d'operativa bancaria

Per a poder validar el funcionament correcte del framework base i del panell de control s'haurà de desenvolupar una automatització d'un procés d'operativa bancaria que estigui creat a través del framework base i gestionat a partir del panell de control.

Objectiu 4: Crear i gestionar bases de dades

Per tal de guardar la informació necessària s'ha hagut de crear una base de dades per al panell de control i una altre per l'automatització desenvolupada. A part s'han afegit diferents tasques de manteniment per a que el funcionament de les bases de dades sigui òptim i no hi hagi cap pèrdua d'informació (realització automàtica de Backups...).

5.1 Justificació del perquè el projecte proposat s'adequa a les característiques de l'especialitat de Sistemes d'Informació.

A continuació s'explicaran els diferents motius per els quals aquest projecte s'adequa correctament a les característiques de l'especialitat de Sistemes d'Informació.

El projecte es construirà per a donar suport a una necessitat en l'entorn empresarial d'everis BPO, mitjançant la creació, desenvolupament, implementació i manteniment d'un petit sistema d'informació que permetrà augmentar l'eficiència de molts processos de l'empresa, ja sigui gràcies a les automatitzacions, o degut a que la feina que realitzava abans una persona i ara realitza una automatització ha provocat que aquesta persona ajudi en un altre procés i per tant aquest millori la seva eficiència i productivitat.

Es realitzarà un procés per part de l'estudiant per a comprendre els diferents processos que es realitzen en aquest projecte d'everis per a poder posar en comú les diferents necessitats que haurà de satisfer el projecte, concretament per estudiar com funcionen l'operativa dels processos que realitzen persones i que seran substituís per automatitzacions, i també per a entendre quina informació és vol habitualment a l'hora de monitoritzar-los, generar informes d'activitat, etc.

Un cop entesos els processos i amb una imatge del problema actual, es realitzar una investigació de les tecnologies amb les que es pot desenvolupar una solució i es triarà la més adequada. S'analitzaran els diferents requisits que ha de complir aquesta solució de la millor manera possible.

A més s'aplicaran els coneixements obtinguts durant la carrera per al correcte desenvolupament del sistema i l'extracció i explotació dades. En aquest cas s'utilitzarà Sql Server Managment per a tal d'administrar tota la informació i per a construir diferents consultes (querys) que després s'executaran des del sistema construït.

Per tant per al desenvolupament d'aquest projecte complirem la definició d'un sistema d'informació al peu de la lletra, ja que oferirem un suport a diferents operacions empresarials a través d'un framework base que permeti desenvolupar les automatitzacions, augmentat la productivitat de l'empresa, proporcionarem una eina per a que puguin gestionar-les (panell de control) i a partir d'elles generarem informes que permetin millorar la presa de decisions de l'equip directiu del projecte, els líders dels diferents processos, etc.

5.2 Justificació de les competències tècniques

CSI2.2: Concebre, desplegar, organitzar i gestionar sistemes i serveis informàtics, en contextos empresarials o institucionals, per a millorar-ne els processos de negoci; responsabilitzar-se'n i liderar-ne la posada en marxa i la millora contínua; valorar el seu impacte econòmic i social. [En profunditat]

Justificació: El projecte consisteix justament en concebre, desplegar organitzar i gestionar un sistema de creació gestió d'automatitzacions, per tant assoliré aquesta competència en profunditat.

CSI3.3: Avaluar ofertes tecnològiques per al desenvolupament de sistemes d'informació i gestió. [Una mica]

Justificació: Per a realitzar un estudi de l'estat de l'art s'avaluaran les ofertes tecnològiques que permeten fer el que es busca en aquest projecte.

CSI4.1: Participar activament en l'especificació dels sistemes d'informació i de comunicació. [En profunditat]

Justificació: El sistema d'informació d'automatitzacions usat en aquest projecte estarà pensat i implementat per mi, amb l'ajuda de la meva tutora i el consentiment de els caps del projecte.

CSI4.2: Participar activament en el disseny, la implementació i el manteniment dels sistemes d'informació i de comunicació. [En profunditat]

Justificació : El sistema d'informació d'automatitzacions usat en aquest projecte estarà pensat, dissenyat i implementat per mi, amb l'ajuda de la meva tutora i el consentiment de els caps del projecte.

CSI4.3: Administrar bases de dades (CES1.6). [En profunditat]

Justificació: El panell de control utilitzarà diferents bases de dades que seran creades i administrades per mi.

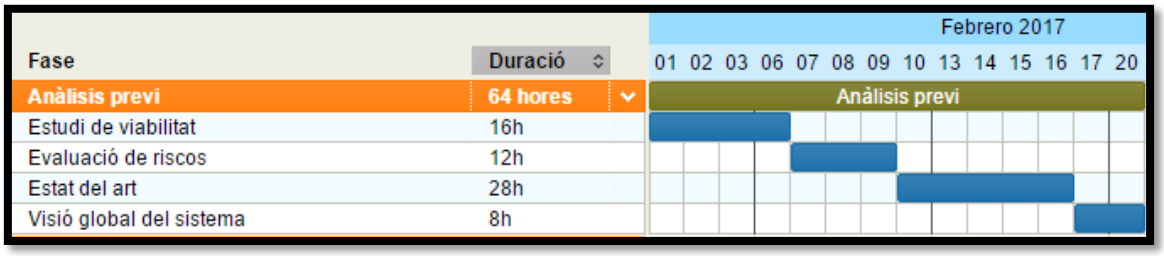
6. Planificació temporal

El desenvolupament del treball s'ha dividit en diferents fases. Per a cada d'una d'elles s'especifica la duració aproximada en hores.

6.1 Descripció de les tasques per fases

6.1.1 Fase 1: Anàlisis previ

En aquesta fase s'estudiarà la viabilitat del projecte a través de entrevistes a diferents parts interessades i estudiant les possibles limitacions legals que puguin sorgir, descrivint i avaluant els riscos que hi pot haver-hi un cop el projecte estigui en funcionament. També es realitzi un estudi de l'estat de l'art de les aplicacions similars ja existents, i una visió global del sistema amb totes les funcionalitats que ha d'incorporar, de les quals algunes seran descartades per la limitació temporal del projecte.

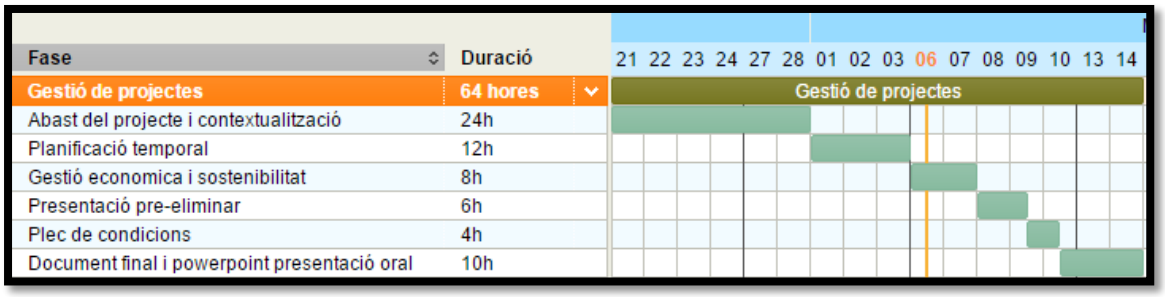


Imatge 1: Planificació Anàlisis previ

6.1.2 Fase 2: Gestió de projectes

Els continguts d'aquesta fase es van generar a mesura que és cursa l'assignatura semipresencial de Gestió de Projectes (GEP), que te una duració de cinc setmanes. S'hauran de realitzar i entregar els següents documents:

- Definició de l'abast i contextualització
- Planificació temporal
- Gestió econòmica i sostenibilitat
- Presentació preliminar
- Plec de condicions
- Document final i powerpoint presentació oral



Imatge 2: Planificació gestió de projectes

6.1.3 Fase 3: Anàlisis de requisits

En aquesta fase es defineixen les parts interessades del sistema, s'estableixen els requisits no funcionals i funcionals.

Imatge 3: Planificació anàlisis de requisits

6.1.4 Fase 4: Especificació

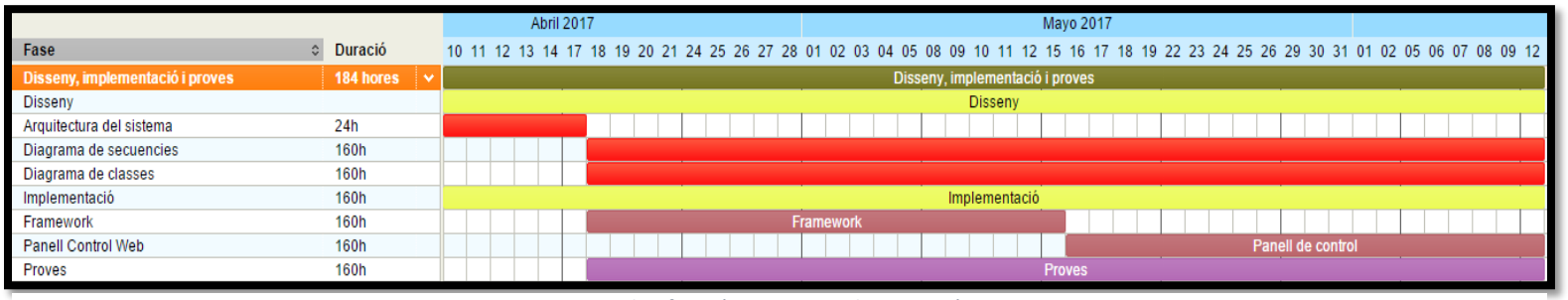
Descripció dels casos d'us i realització de l'esquema conceptual i el model de comportament del sistema.

Imatge 3: Planificació especificació

6.1.5 Fase 5: Disseny, implementació i proves

Respecte al disseny, es definirà l'arquitectura del sistema, la distribució i funció de cada capa, els diagrames de seqüencia de cada cas d'us, el diagrama de classes i l'estructura de la base de dades.

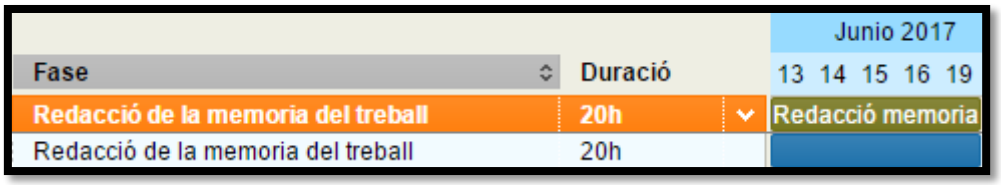
La implementació es dividirà en dues parts, per un costat, s'implementarà el framework i un cop acabada es desenvoluparà el panell de control web. En les dues tasques la construcció serà iterativa, fent proves al final de cada iteració. Al final de la fase es provarà el sistema conjuntament mitjançant un pla de proves per a depurar errors i augmentar la qualitat de sistema final.



Imatge 4: Planificació Disseny, implementació i proves

6.1.6 Redacció de la memòria del treball

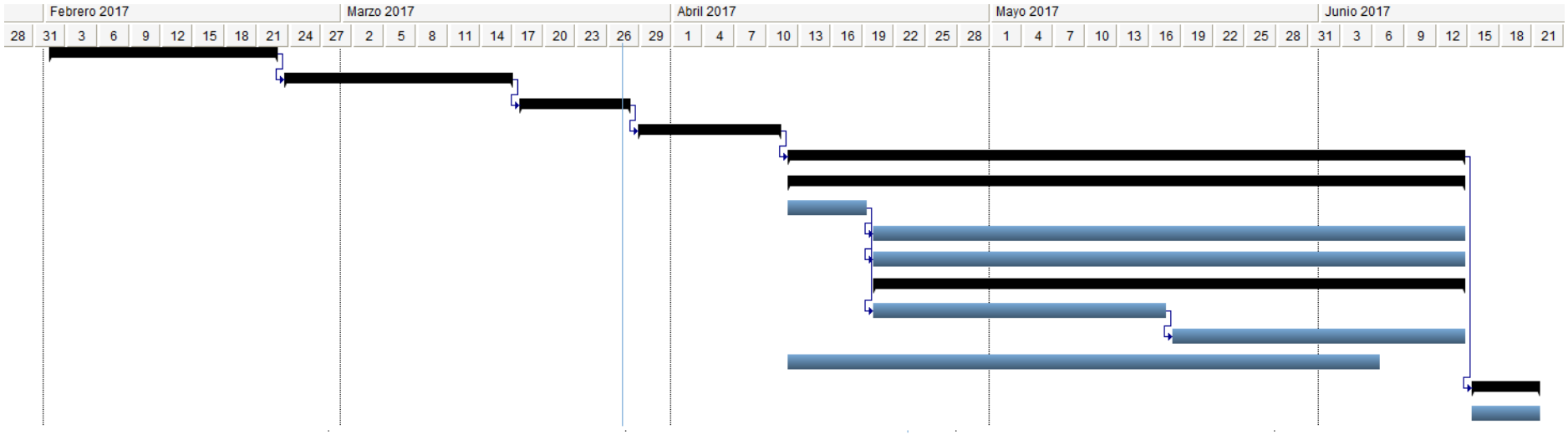
Finalment, es revisarà tot el treball, es redactaran les conclusions i es mostraran les desviacions de la planificació inicial.



Imatge 5: Planificació redacció de la memòria del treball

6.2 Diagrama de Gantt

		Nombre	Duración	Inicio	Fin	Predecesoras	Recursos
1		**1. Anàlisis previ**	**16d?**	**01/02/2017**	**22/02/2017**		
6		**2. Gestió de projectes**	**16d?**	**23/02/2017**	**16/03/2017**	**1**	
13		**3. Anàlisis de requisits**	**7d**	**17/03/2017**	**27/03/2017**	**6**	
18		**4. Especificació**	**10d**	**28/03/2017**	**10/04/2017**	**13**	
22		**5. Disseny, implementació i proves**	**46d**	**11/04/2017**	**13/06/2017**	**18**	
23		**Disseny**	**46d**	**11/04/2017**	**13/06/2017**		
24		Arquitectura del sistema	6d	11/04/2017	18/04/2017		
25		Diagrama de secuencies	40d	19/04/2017	13/06/2017	24	
26		Diagrama de classes	40d	19/04/2017	13/06/2017	24	
27		**Implementació**	**40d**	**19/04/2017**	**13/06/2017**		
28		Framework	20d	19/04/2017	16/05/2017	24	
29		Panell Control Web	20d	17/05/2017	13/06/2017	28	
30		Proves	40d	11/04/2017	05/06/2017		
31		**6. Redacció de la memoria del treball**	**5d**	**14/06/2017**	**20/06/2017**	**22**	
32		Redacció de la memoria del treball	5d	14/06/2017	20/06/2017		



6.3 Desviacions en la planificació

Durant la fase cinc, disseny, implementació i validació s'han obviat alguns dels apartats que es van definir originalment, concretament els diagrames de seqüencia, ja que un es va realitzar la anàlisis de requisits i dissenyar el sistema ens van sortir tants casos que haver creat tots els diagrames de seqüencia hagués produït una desviació en la implementació del projecte considerable.

Per tant, es va decidir, conjuntament amb la tutora i els caps de projecte, obviar els diagrames de seqüencia i començar directament les iteracions per a dur a terme la implementació. Aquestes hores destinades a la creació dels diagrames de seqüencia s'han destinat a la implementació de l'automatització que ha servit per a validar la correctesa del projecte, és a dir, les proves de la fase 5 (que s'anaven construint alhora que es desenvolupava el projecte).

A part d'aquest canvi no se n'ha produït cap canvi altre canvi destacable, i la planificació s'ha seguit acuradament.

7. Gestió econòmica

7.1 Identificació i estimació dels costos

En aquesta secció es fa una estimació dels elements que composen el pressupost del projecte tenint en compte els recursos que s'han utilitzat. Entre aquests s'inclouen les hores de feina dedicades i el cost del software i el hardware, entre d'altres.

7.1.1 Recursos humans

El projecte serà desenvolupat per només una persona, i aquesta haurà d'assumir diferents papers en el procés de creació de sistema. A continuació és mostra una taula dels diferents rols que s'adoptaran els les fases explicades prèviament i el seu cost en el mon laboral.

Recurs	Fase	Rol	€/H	Estimació hores	Total estimat
Arnau Santamaria	Anàlisis Previ	Cap de projecte	50€/h	64h	3.200€
	Gestió de projectes	Cap de projecte	40€/h	64h	3.200€
	Anàlisis de requisit	Analista	35€/h	28h	980€
	Especificació	Analista	35€/h	40h	1.400€
	Disseny, implementació i proves	Programador	20€/h	184h	3.680€
	Memòria final	Cap de projecte	50€/h	20h	1.000€
Total				400h	13.460€

Taula 1: Recursos humans

7.1.2 Recursos software

Per a realitzar el projecte han sigut necessaris l'ús d'eines de software, la majoria d'elles gratuïtes.

Producte	Cost	Vida útil	Amortització total estimada (5 mesos)
Windows 7	25,95€	3 anys	3,6€
Microsoft Office 2010	149,00€	3 anys	20,69€
NetBeans 8.1	0€	-	0€
Google Chrome	0€	-	0€
Gantter	0€	-	0€
Dia	0€	-	0€
Atom	0€	-	0€
Glassfish	0€	-	0€
Internet Information Services	0€	-	0€
TOTAL			24,29€

Taula 2: Recursos software

Aquests preus són teòrics ja que no sabem si Microsoft i Everis tenen un contracte específic per a oferir millors preus per els productes degut a la gran quantitat que compra l'empresa. Aquests són els preus de mercat de la botiga de Microsoft. (Windows 7 ja no és comercialitza per Microsoft, el preu s'ha agafat de la web de ventes Lizengo).

7.1.3 Recursos hardware

Per el desenvolupament del software i la realització de la memòria, s'ha utilitzat una ordinador portàtil.

Producte	Cost	Vida útil	Amortització total estimada (5 mesos)
Ordinador portàtil	560€	5 anys	46,67€
Total			46,67€

Taula 3: Recursos Hardware

7.1.4 Costos generals

En aquest apartat es tenen en compte costos addicionals a la realització del projecte com poden ser l'energia elèctrica consumida, l'accés a internet i el cost del transport del desplaçament a l'empresa.

Producte/Servei	Cost	Període	Total estimat
Energia elèctrica	0.11562€/kWh * ~ 0.11562kW	450h	6,01€
Transport	T-Jove 105€/Trimestre	2 trimestres	210€
Accés a internet	40€/mes	5 mesos	200€
Total			416,01€

Taula 4: Costos generals

El cost d'energia elèctrica s'ha obtingut del portal web de consumidors d'energia. [R12]

El preu d'internet es el preu mitja per a particulars ja que no sabem quin es el proveïdor d'internet a everis.

7.1.5 Costos imprevistos

L'únic cost imprevist que podria sorgir i que puguem controlar és un error en l'ordinador utilitzat per al desenvolupament del projecte; s'haurà d'afegir el cost de reparació que aproximadament seria d'uns 100€, i mentre estigues inactiu s'utilitzaria un segon ordinador per a no alentir les activitats planificades.

7.1.6 Cost total del projecte

A continuació podem veure una taula amb un resum dels costos del projecte.

Concepte	Cost
Costos directes	13.460€
Costos indirectes	416,01€
Costos imprevistos	100€
Total	13.967,01€

Taula 5: Resum de costos del projecte

8. Sostenibilitat

Per tal d'identificar i tenir en compte la sostenibilitat de projecte, s'avalua el impacte de tres aspectes: econòmic, social i ambiental. A continuació podem veure un anàlisis de cadascun d'ells, des de les fites inicials del PPP (Projecte Posada en producció) fins a la vida útil.

A continuació es mostra la matriu ponderada amb el resultat de l'avaluació.

	PPP	Vida útil
Ambiental	5/10	15/20
Econòmic	8/10	16/20
Social	7/10	18/20
Sostenibilitat	20/30	51/60
	71/90	

8.1 Impacte econòmic

Per aquest projecte s'ha realitzat una avaluació de costos de recursos i materials, tot i que no s'ha tingut en compte els costos d'ajustaments, actualitzacions o reparacions que poguessin sorgir durant la vida útil del projecte.

El temps dedicat a cadascuna de les tasques del projecte han sigut planificades segons la seva importància.

Respecte al cost total del projecte, en comparació a altres projectes de sistemes d'informació que realitzen empreses del sector, te un cost ajustat i per tant seria competitiu i viable. Respecte a la seva duració, seria possible acabar-ho en menys temps si es disposes de més recursos humans amb més experiència professional. Actualment, seria difícil reduir la utilització de recursos degut a que se n'han utilitzat molt pocs (un únic ordinador i costos indirectes baixos).

El projecte esta desenvolupat en col·laboració amb l'empresa everis i s'espera que aquest sistema es pugui aprofitar en el futur , ajudant a millorar la eficiència d'un projecte everis BPO.

8.2 Impacte social

En referencia a la situació actual del país on s'ha desenvolupat el projecte (Catalunya, Espanya) és troba ara, segons l'actual govern, sortint de la crisis econòmica que es va iniciar el 2008.

Segon la OECD (Organització per la Cooperació i Desenvolupament Econòmic) l'economia ha augmentat una taxa de 2.6% a l'estat espanyol als últims 3 anys. La taxa d'atur si bé ha millorat va ser d'un 19% a finals del 2016 i un 43% entre els més joves. (http://www.oecd.org/spain/economic-survey-spain.htm)

En aquest context en que encara existeixen taxes d'atur altes les automatitzacions a vegades són vistes com el fet que provocarà la substitució d'una persona per un ordinador.

Tot i que la humanitat sembla que esta abocada a un canvi en el paradigma actual dels tipus de feina que existeixen, ja que les màquines seran capaces de fer moltes de les feines que es duen a terme actualment en la societat, l'impacte que tindrà aquest sistema en l'empresa no serà aquest.

L'objectiu no serà el de substituir empleats per ordinadors sinó que amb l'ajuda de les automatitzacions totals o parcials dels processos, els empleats canviïn el tipus de feina per una no repetitiva i de més valor.

El impacte social serà positiu degut a que es millorarà les tasques a realitzar, a més que s'augmentarà l'eficiència i productivitat dels processos així com una reducció de costos.

8.3 Impactes ambientals

L'elaboració del projecte provoca la utilització de diferents recursos que afecten al medi ambient. En les diferents fases del projecte s'utilitzarà energia elèctrica per a poder treballar i el cost de produir l'energia té un impacte ambiental. Concretament s'haurà de mantenir alimentat l'ordinador de desenvolupament.

Apart, també s'hauria de tindre en compte l'electricitat utilitzada per everis en les seves oficines, ja sigui en la llum de l'oficina, l'Internet, etc, però això no ho hem pogut tindre en compte perquè es una dada desconeguda.

9. Especificació i implementació

9.1 Framework

9.1.1 Especificació

9.1.1.1 Anàlisi de requisits funcionals

Els requisits funcionals definiran accions fonamentals que el nostre sistema ha de realitzar al rebre informació, processar-la i, posteriorment, produir certs resultats.

	Número 1
Descripció	El framework ha de permetre simular l'ús del teclat.
Justificació del requisit	Per a poder realitzar el mateix procés que un humà, el framework haurà de proporcionar una eina per a simular l'ús del teclat i poder escriure, realitzar combinacions de tecles (Ctrl +C,Ctrl + V, etc).
Condició de satisfacció	El framework es capaç d'escriure i simular les combinacions de tecles del teclat.

	Número 2
Descripció	El framework ha de permetre simular l'ús del ratolí.
Justificació del requisit	La base de les automatitzacions sobre pantalla és la possibilitat de simular els clics d'un ratolí.
Condició de satisfacció	El framework es capaç de fer clic sobre les imatges de la pantalla o coordenades de la pantalla.

	Número 3
Descripció	El framework ha de permetre detectar imatges en pantalla.
Justificació del requisit	Per a poder crear automatitzacions a través d'imatges el framework ha de ser capaç de detectar imatges en pantalla.
Condició de satisfacció	El framwork es capaç de reconèixer imatges i clicar-les, o escriure-hi a sobre.

Número 4	
Descripció	El framework ha de permetre la comunicació amb el panell de control.
Justificació del requisit	Per a poder guardar logs al panell de control, informar de l'estat de l'execució del robot, etc el robot ha de tindre un sistema que permeti comunicar-se amb la base de dades del panell de control.
Condició de satisfacció	El framework es comunica amb la base de dades per a inserir o consultar informació.

Número 5	
Descripció	El framework ha de permetre comunicar-se amb qualsevol base de dades a través de l'arquitectura REST.
Justificació del requisit	Com hem escollit l'arquitectura REST per a comunicar les automatitzacions amb el servidor i les bases de dades, el framework ha d'oferir un client REST per a poder realitzar aquesta comunicació de manera senzilla.
Condició de satisfacció	El framework és capaç de comunicar-se amb els servidors que sigui necessari a través del client REST creat.

Número 6	
Descripció	El framework ha de permetre interactuar amb els elements de les pàgines web.
Justificació del requisit	Per a poder accedir a les aplicacions de l'entorn bancari de l'entitat financera serà necessari obrir l'extranet a través del navegador i navegar fins a obrir l'aplicació necessària.
Condició de satisfacció	El framework es capaç d'obrir qualsevol navegador i navegar a través d'ell fins a la web del banc per a obrir l'aplicació necessària.

Número 7	
Descripció	El framework ha de permetre interactuar amb els elements de les pàgines web.
Justificació del requisit	Per a poder accedir a les aplicacions de l'entorn bancari de l'entitat financera serà necessari obrir l'extranet a través del navegador i navegar fins a obrir l'aplicació necessària.
Condició de satisfacció	El framework es capaç d'obrir qualsevol navegador i navegar a través d'ell fins a la web del banc per a obrir l'aplicació necessària.

9.1.1.2 Anàlisi de requisits no-funcionals

Requisits de capacitat d'ús i humanitat

Número 8	
Tipus de requisit	11a. Requisits de facilitat d'ús
Descripció	El sistema ha de ser eficient i fàcil d'usar.
Justificació del requisit	Cal que el producte sigui el més eficient possible en el seu ús per tal que els usuaris puguin aprendre a utilitzar-lo de manera ràpida i senzilla.
Condició de satisfacció	Els usuaris no troben problemes en el moment d'utilitzar el framework i entenen totes les seves funcionalitats.

Número 9	
Tipus de requisit	11b. Requisits de personalització i internacionalització.
Descripció	El sistema s'oferirà en anglès.
Justificació del requisit	L'anglès es l'idioma internacional més parlat, i per tal de que el framework pugui ser entens per tothom estarà desenvolupat en aquest idioma.
Condició de satisfacció	El 100% dels mètodes, classes i documentació estan escrits en anglès.

Número 10	
Tipus de requisit	11c. Requisits d'aprenentatge
Descripció	El framework ha de poder ser utilitzat per persones que no rebran un aprenentatge previ abans d'utilitzar-lo.
Justificació del requisit	S'ha buscat crear un framework fàcil d'utilitzar per poder crear automatitzacions de manera senzilla i ràpida.
Condició de satisfacció	L'usuari podrà crear automatitzacions sense haver rebut formació prèvia.

Requisits d'acompliment

Número 11	
Tipus de requisit	12a. Requisits de latència i velocitat
Descripció	La resposta del framework ha de ser suficientment ràpida per evitar la interrupció del flux de les automatitzacions en funcionament.
Justificació del requisit	Un temps de resposta ràpid permetrà a l'automatització executar-se de manera correcta.
Condició de satisfacció	Les automatitzacions correctament creades usant el framework funcionaran tal i com s'espera.

Número 12	
Tipus de requisit	12d. Requisits de fiabilitat i disponibilitat
Descripció	El framework estarà disponible per a les 24 hores del dia durant els 365 dies que conformen l'any.
Justificació del requisit	Els desenvolupadors del projecte han de poder crear automatitzacions que es requereixin en qualsevol moment a través del framework.
Condició de satisfacció	La descarrega del framework estarà disponible i completament funcional tot el temps.

Requisits operacionals i ambientals

Número 13	
Tipus de requisit	13c. Requisits de producció
Descripció	El framework serà distribuït com un arxiu .jar.
Justificació del requisit	Per a poder instal·lar-se com una llibreria externa en qualsevol projecte el format més correcte es el jar.
Condició de satisfacció	El framework es pot instal·lar i usar de manera correcte en els nous desenvolupaments.

Número 14	
Tipus de requisit	13d. Requisits de llançament
Descripció	Cada nova versió no causarà la falla de característiques prèvies.
Justificació del requisit	El llançament d'una nova versió no pot fer que automatitzacions que funcionen correctament deixin de funcionar.
Condició de satisfacció	El framework no canviarà cap funcionalitat existent en versions prèvies del framework.

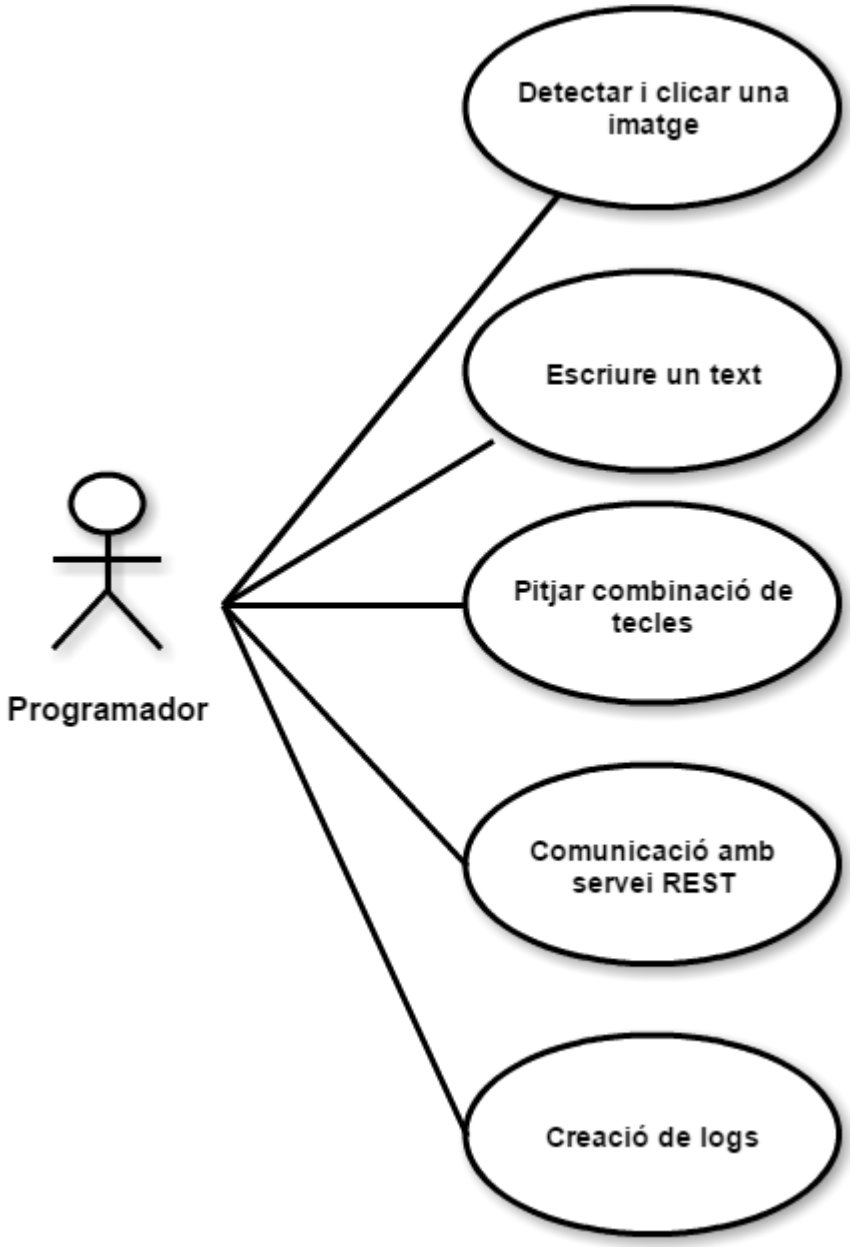
Requisits de preservació i suport

Número 15	
Tipus de requisit	14c. Requisits d'adaptabilitat
Descripció	El framework treballarà amb les ultimes versions de Windows i Linux.
Justificació del requisit	El framework ha de poder ser utilitzat en els sistemes operatius de l'empresa, Windows 7 i Windows 10.
Condició de satisfacció	El framework es utilitzat correctament sota aquests dos sistemes operatius.

9.1.2 Casos d'ús

9.1.2.1 Diagrama de casos d'ús

A continuació es mostra el diagrama dels casos d'ús mes rellevants del sistema i imprescindibles.



9.1.2.2 Descripció casos d'ús

Cas d'ús 1: Detectar i clicar una imatge

Actor principal: Usuari (desenvolupador/programador)

Disparador: L'usuari vol programar la detecció d'una imatge en pantalla i clicar-hi a sobre.

Escenari principal d'èxit:

1. L'usuari importa el framework en el seu projecte.

2. L'usuari crida a la classe que conté les funcionalitats de detectar i clicar imatges

3. L'usuari crida al mètode de la classe que li sigui necessari segons el que vulgui fer(buscar i fer clic, doble clic, clic dret, etc) i li passa els paràmetres necessàries, el més important és la ruta a la imatge a buscar en pantalla.

4. L'usuari executa el codi.

5. El framework busca la imatge en la pantalla i hi clica.

6. Continua l'execució del codi.

Extensions:

5a. El framework no troba la imatge en pantalla.

5a1. El framework retorna que no s'ha trobat la imatge en pantalla i segueix la seva execució.

5b. El framework no troba la imatge en la ruta indicada.

5b1. El framework llança una excepció d'imatge no trobada i atura l'execució.

5b2. S'acaba el cas d'ús.

Cas d'ús 2: Escriure un text

Actor principal: Usuari (desenvolupador/programador)

Disparador: L'usuari vol programar la escriptura d'un text

Escenari principal d'èxit:

1. L'usuari importa el framework en el seu projecte.

2. L'usuari crida a la classe que conté les funcionalitats d'interacció amb el teclat.

3. L'usuari crida la funció per a escriure el text i li passa els paràmetres necessàries, el més important és el text a escriure.

4. L'usuari executa el codi.

5. El framework escriu el text que ha rebut a través dels paràmetres.

6. Continua l'execució del codi.

Extensions:

5a. El framework no ha pogut escriure correctament.

5b1. El framework llança una excepció d'escriptura fallida.

5b2. S'acaba el cas d'ús.

Cas d'ús 3: Pitjar combinació de tecles (Ctrl + C, Ctrl + V...)

Actor principal: Usuari (desenvolupador/programador)

Disparador: L'usuari vol pitjar una combinació de tecles del teclat per a una finalitat concreta.

Escenari principal d'èxit:

1. L'usuari importa el framework en el seu projecte.

2. L'usuari crida a la classe que conté les funcionalitats d'interacció amb el teclat.

3. L'usuari crida la funció per a escriure el text i li passa els paràmetres necessàries, el més important la tecla modificadora (Ctrl, Alt, Shift, etc) i la segona tecla que es vol pitjar.

4. L'usuari executa el codi.

5. El framework pitja la combinació de tecles produint el efecte desitjat.

6. Continua l'execució del codi.

Cas d'ús 4: L'usuari es vol comunicar amb un servei REST

Actor principal: Usuari (desenvolupador/programador)

Disparador: L'usuari vol inserir/extreure informació d'una base de dades a través d'un servei REST.

Escenari principal d'èxit:

1. L'usuari importa el framework en el seu projecte.

2. L'usuari declara una instancia del client REST del framework.

3. L'usuari configura el webResource que utilitza el client REST amb les dades del servidor, la URL a la que s'ha de dirigir, i els paràmetres necessaris en que cas d'inserció o extracció de dades.

4. L'usuari executa el codi.

5. El framework és comunica amb les bases de dades inserint/extraient les dades necessàries.

6. Continua l'execució del codi.

Extensions:

5a. La consulta/inserció de la base de dades ha fallat.

5b1. El framework llança una excepció d'escriptura/lectura en base de dades incorrecte..

5b2. S'acaba el cas d'ús.

Cas d'ús 5: L'usuari vol crear un log a la base de dades.

Actor principal: Usuari (desenvolupador/programador)

Disparador: L'usuari vol crear un registre log a la base de dades.

Escenari principal d'èxit:

1. L'usuari importa el framework en el seu projecte.

2. L'usuari declara una instancia de la classe log del framework.

3. L'usuari crida al controlador de configuració per a obtindré la configuració del robot

al panell de control.

4. L'usuari crida a la funció depenent del tipus de log que vulgui crear i passa el missatge que vol guardar.

5. L'usuari executa el codi.

6. El framework és comunica amb les bases de dades inserint el log.

7. Continua l'execució del codi.

Extensions:

3a. L'usuari no ha cridat a la configuració del robot en el panell de control.

3a1. El framework llança un avís de configuració no carregada i atura la inserció del log.

3a2. S'acaba el cas d'ús.

3b. La configuració del robot no ha sigut creada al panell de control.

3b1. El framework llança un avís de configuració no creada i atura la inserció del log.

3b2. S'acaba el cas d'ús.

9.1.3 Tecnologia i eines usades pel desenvolupament

El framework ha estat creat en el llenguatge de programació Java amb l'ús de diverses llibreries externes que explicarem a continuació:

Java

Després de realitzar proves amb els dos llenguatges de programació conjuntament amb les llibreries mencionades en l'apartat de tecnologies candidates va decidir escollir Java per les següents raons:

- Es el mateix llenguatge en que es va desenvolupar SoftBan i això fa que moltes de les funcions usades en automatitzacions, com poden ser executar combinacions de tecles per a un propòsit (copiar, enganxar, etc) tinguin els mateixos codis.

  Fent proves amb c# van sorgir problemes a l'hora d'executar diferents combinacions de tecles, ja que el llistat de codis per interpretar una tecla no es el mateix en C# que en Java ,és a dir, que per exemple Ctrl + C no te la mateixa codificació en els dos llenguatges.

- Integra perfectament la llibreria de Sikuli.

- Coneixement extens del llenguatge.

Sikuli

SikuliX automatitza qualsevol cosa que vegi en la pantalla del seu ordinador d'escriptori amb Windows, Mac o algun Linux / Unix. Utilitza el reconeixement d'imatges impulsat per OpenCV per identificar i controlar els components de la GUI. Això és útil en els casos en què no hi ha fàcil accés als components interns d'una interfície gràfica d'usuari o el codi font de la pàgina o aplicació web que desitja actuar.

A més Sikuli presenta una API per a Java de fàcil accés i utilització, es gratuït i pot ser usat en la construcció d'un producte d'ús personal o comercial, existeix una web amb documentació abundant i amb fòrums d'ajuda i de les tres llibreries candidates es la que millor interactua amb SoftBan, l'aplicació que usa l'entitat financera per a que els agents de l'empresa realitzin totes les seves operacions, i que esta desenvolupada en Java.

Unirest

Unirest és un conjunt de biblioteques HTTP lleugeres disponibles en múltiples idiomes, construïdes i mantingudes per Mashape, que també mantenen l'API Open-Gateway Kong. [R13]

Permeten comunicar d'una manera molt senzilla amb qualsevol servei REST disponible, permeten tot el tipus d'operacions necessaris com get, put, post o delete.

Selenium

Selenium automatitza els navegadors. Principalment, és per automatitzar aplicacions web amb finalitats de proves, però certament no es limita a això.

Selenium compta amb el suport d'alguns dels proveïdors de navegadors més grans que han pres (o estan prenent) passos per fer Selenium una part nativa del seu navegador. També és la tecnologia bàsica en moltes altres eines d'automatització del navegador, API i marcs. [R14]

El framework incorpora un seguit de funcionalitats per tal de poder automatitzar també el comportament en pàgines web, ja que per accedir a l'entorn del banc s'ha de fer a través del navegador.

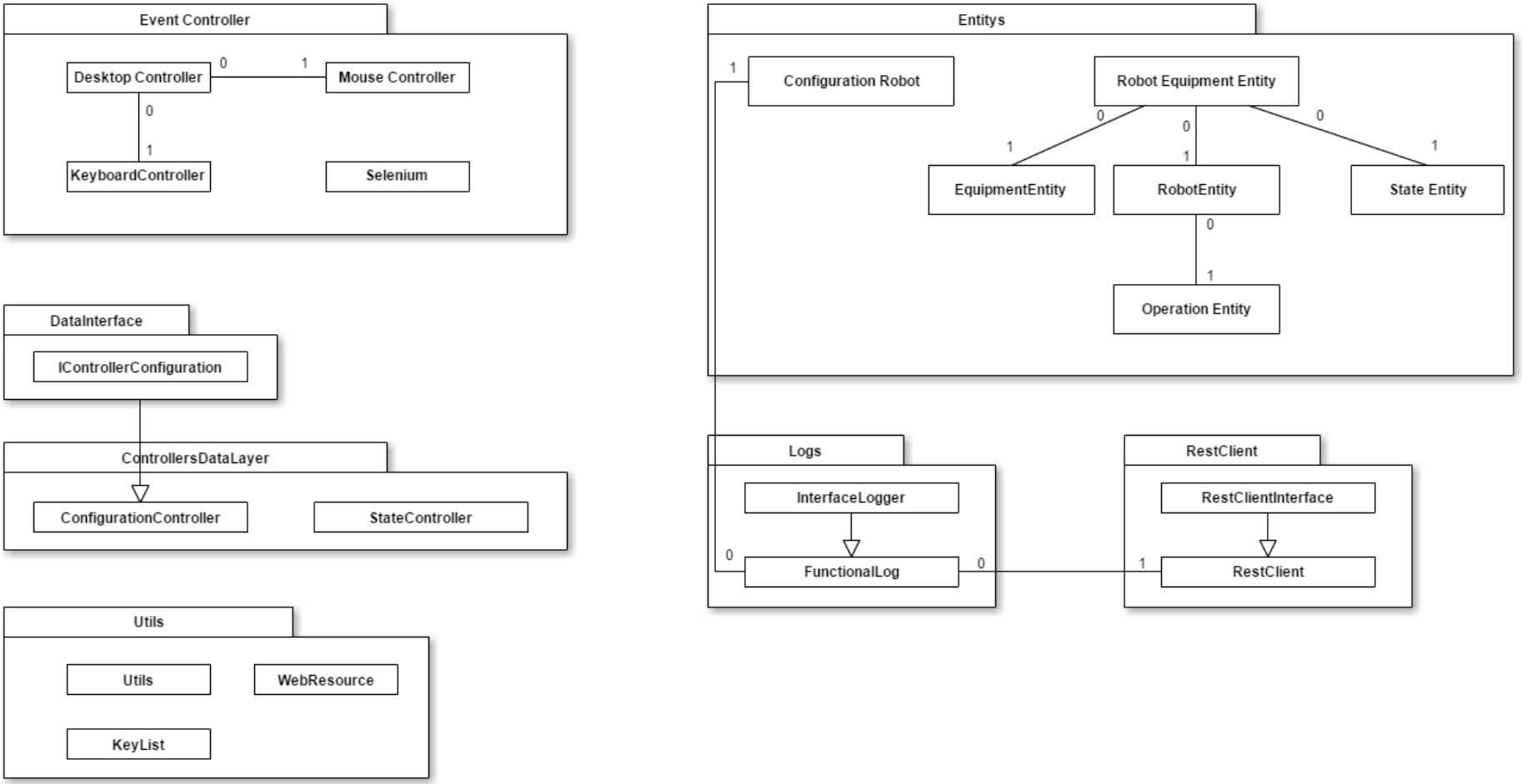
NetBeans 8.1

NetBeans és un entorn de desenvolupament integrat de codi obert, lliure i gratuït, fet principalment per al llenguatge de programació Java. Existeix a més un nombre important de mòduls per estendre'l.

NetBeans és un projecte de gran èxit amb una gran base d'usuaris, una comunitat en constant creixement i amb prop de 100 socis a tot el món. Encara que la primera versió va ser escrita en C++ orientada a Unix, com un projecte d'estudiants anomenant-se Xelfi (Delfi per uniX), posteriorment, ja va ser reescrit a Java. En el 1999 va ser adquirit per Sun MicroSystems i va fundar el projecte el juny de l'any 2000 ja concebut com IDE, continua sent-ne el patrocinador principal. [R15]

9.1.4 Model conceptual del sistema

Aquí podem veure el model conceptual del sistema. El llistat de funcionalitats s'adjuntarà a l'annex on es podrà veure el les classes amb els seus atributs i mètodes.



A continuació hi haurà una descripció de les funcionalitats de cada classe.

Package Event Controller:

El paquet Event Controller conté les classes que s'encarreguen de gestionar tots els events relacionats amb el teclat, el ratolí, la pantalla i el navegador web.

Per a fer-ho s'han creat quatre classes estàtiques que contenen les funcionalitats necessàries. El fet de que siguin estàtiques es per facilitar al programador l'escriptura del codi de les automatitzacions.

A continuació es mostrarà una explicació més detallada de cada classe.

Classe	Descripció	Funcionalitats principals
DesktopController	Classe que conte els mètodes necessaris per a interactuar amb la pantalla.	ExistImage, ClickImage, RightClickImage, WaitImage, ClickImageOffset, RightClickImageOffset...
KeyboardController	Classe que conte els mètodes necessaris per a simular l'ús del teclat.	SendText, SendKey, getClipboard, setClipboard...
MouseController	Classe que conte els mètodes necessaris per a simular l'ús del ratolí.	Click, rightClick, move, getCoords....
Selenium	Classe que conte els mètodes necessaris per a controlar el navegador web.	GoToUrl, wait, findBy, clickElement...

Package Entitys:

El paquet Entitys conté una conjunt de classes que representen les taules de la base de dades del panell de control. Són necessàries per guardar la informació de configuracions de les automatitzacions, els equips en que s'estan executant, entre d'altres. Són necessàries per a poder executar els controladors d'estats i de configuració.

Package DataInterface:

El paquet DataInterface conté les interfícies necessàries que determinen el comportament dels controladors de dades com el controlador de configuració. El paquet esta format per la interfície IControllerConfiguration que estableix els mètodes següents: get(primaryKey), getWithParams(primaryKey) i get().

Package ControllersDataLayer:

El paquet ControllersDataLayer conté els controladors de dades de del framework. El paquet consta de dos controladors, el ConfigurationController i el StateController.

A continuació es mostrarà una explicació més detallada de cada classe.

Classe	Descripció	Funcionalitats principals
ConfigurationController	Classe singleton que conte els mètodes necessaris per a extreure la configuració del robot de la base de dades. No s'ha d'instanciar i només es pot rebre la configuració una vegada, ja que es global per a tota l'automatització.	Get, getWithParams
StateController	Classe que conte els mètodes necessaris per a poder actualitzar l'estat del robot a la base de dades de manera senzilla.	updateStateActivo, updateStateEnEjecución, updateStateIniciando, updateStateParado...

Package Logs:

El paquet Logs conté la interfície que especifica el comportament dels logs i l'únic tipus de log implementat fins ara que són els logs funcionals, que guarden traces de les execucions de les automatitzacions.

La interfície s'ha creat perquè existeix la possibilitat d'afegir més tipus de logs en el futur.

A continuació es mostrarà una explicació més detallada de cada classe.

Classe	Descripció	Funcionalitats principals
InterfaceLogger	Interfície que especifica el comportament dels logs.	Verbose, debug, error, fatal, information
FunctionalLog	Classe que implementa els mètodes de la interficie InterfaceLogger i guarda logs a la base de dades.	Verbose, debug, error, fatal, information

Package RestClient:

El paquet RestClient conté les interfícies i classes necessàries per a poder comunicar-se amb qualsevol servei REST.

A continuació es mostrarà una explicació més detallada de cada classe.

Classe	Descripció	Funcionalitats principals
RestClientInterface	Interfície que especifica el comportament del client rest.	Get, getArray, post, put
RestClient	Rest que implementa la interfície RestClientInterface.	Get, getArray, post, put

Package Utils:

El paquet utils conté diverses classes que poden resultar utils per al programador.

A continuació es mostrarà una explicació més detallada de cada classe.

Classe	Descripció	Funcionalitats principals
KeyList	Conté els codis de cada tecla del teclat.	Ctrl, Alt, Shift, A, B, C...
WebResource	Classe que és usada per a comunicar-se amb serveis REST.	SetPort, SetResourceApp, getJavaResourceBase, getJavaResourceWithParams...

9.1.5 Implementació

9.1.5.1 Funcionalitats

Seguidament, es descriuran les funcionalitats més rellevants del framework, tot el codi esta pujat a un servidor privat de BitBucket.

Detecció i clic en una imatge

Per a la realització la detecció i clic de una imatge cridem a la llibreria Screen que incorpora Sikuli i ens permet cridar a aquest tipus de funció, i en tractem el resultat.

Detecció:

```
1.      /**
2.       * Check if the image exist in the screen in certain time and after a sleep time.
3.       *
4.       * @param pathImage Image Path.
5.       * @param timeOut TimeOut.
6.       * @param sleepTime Sleep time
7.       * @return True if exist, false if not exist.
8.       */
9.      public static boolean existImage(String pathImage, int timeOut, int sleepTime) {
10.         Utils.sleep(sleepTime);
11.         return SC.exists(pathImage, timeOut) != null;
12.     }
```

Clic:

```
1.      /**
2.       * Click the image in the screen in certain time and after a sleep time
3.       *
4.       * @param pathImage Image path.
5.       * @param timeOut Time out.
6.       * @param sleepTime Sleep time in function.
7.       * @return True if the image was clicked succesfully, false if not
8.       */
9.       public static boolean clickImage(String pathImage, int timeOut, int sleepTime) thro
    ws FindFailed {
10.         boolean clickOk = false;
11.         try {
12.             Utils.sleep(sleepTime);
13.             clickOk = (1 == SC.click(pathImage, timeOut)); //1 if success, 0 otherwise
14.         } catch (FindFailed findFailedException) {
15.             System.err.println("Image wasn't found." + findFailedException);
16.             throw new FindFailed("Imatge no trobada");
17.         }
18.         return clickOk;
19.     }
```

Comunicació amb un client REST

Per a comunicar-nos amb qualsevol client REST utilitzem la llibreria UNIREST, que ens permet consumir una servei rest de manera senzilla.

Aquí es mostra la classe com una enumeration, que es una de les maneres més senzilles de crear una classe singleton consumint els menors recursos possibles. El patró singleton és un patró usat quan només es vol tindre una instancia d'un objecte, que pugui ser accedit de manera global i sense possibilitat a crear-ne un de nou.

Per a retornar la informació usem una classe anomenada JSONObject creada per Oracle, que permet emmagatzemar i construir JSON.

```
1.  /**
2.   * REST Client Class
3.   *
4.   * @author asantapi
5.   */
6.  public enum RestClient implements RestClientInterface {
7.
8.      INSTANCE;
9.      private WebResource webResource;
10.
11.     /**
12.      * Consume Get on the REST API
13.      *
14.      * @return Information in JSON Object
15.      */
16.     @Override
17.     public JSONObject get() {
18.         JSONObject jsonReturnedObject = null;
19.         try {
20.
21.              HttpResponse<JsonNode> jsonResponse = Unirest.get(webResource.getJavaResour
    ce())
22.                      .asJson();
23.
24.             jsonReturnedObject = jsonResponse.getBody().getObject();
25.
26.         } catch (UnirestException ex) {
27.             // Tratar exepción de la manera correcta
28.            ex.printStackTrace();
29.         }
30.         return jsonReturnedObject;
31.     }
```

Per a configurar els paràmetres del servidor REST s'usa una classe que hem anomenat WebResource. Seguidament es mostrarà les funcions que creen el string que conte la url del servei REST.

```
1.      private String getJavaResourceBase()
2.      {
3.           return "http://" + this.ip + ":" + this.puerto + "/" + this.appName + "/webreso
    urces/" + this.resource ;
4.
5.      }
1.      private String getJavaResourceWithPathParams()
2.      {
3.          StringBuilder urlBuilder = new StringBuilder();
4.          urlBuilder.append(this.getJavaResourceBase());
5.          this.pathParams.forEach((param) -> {
6.              urlBuilder.append("/")
7.                      .append(param);
8.          });
9.
10.         return urlBuilder.toString();
11.     }
12.
13.     private String getJavaResourceWithQueryParams()
14.     {
15.         StringBuilder urlBuilder = new StringBuilder();
16.         urlBuilder.append(this.getJavaResourceBase()).append("?");
17.
18.         this.queryParams.forEach((k, v) -> {
19.             urlBuilder.append(k)
20.                     .append("=")
21.                     .append(v)
22.                     .append("&");
23.
24.         });
25.         urlBuilder.setLength(Math.max(urlBuilder.length() - 1, 0));
26.         return urlBuilder.toString();
27.     }
```

Creació d'un log

Seguidament es mostra un conjunt de funcions que serveixen per a la creació de logs. Gràcies a aquestes funcions el programador simplement ha de crear una instancia de la classe i enviar el tipus de log que vulgui crear, per exemple: FunctionalLog.verbose(Missatge).

També podem veure com es configura el webResource amb les dades que vulguem. En aquest cas les dades estan ja definides perquè els logs sempre es crearan a la mateixa base de dades independentment de l'automatització que sigui.

```
1.  /**
2.       * Create a Verbose Log
3.       * @param message Log's Message
4.       * @return True if Fatal log was inserted correctly, false if not
5.       */
6.      @Override
7.      public boolean fatal(String message)
8.      {
9.          JSONObject json = this.createJson(message, "Fatal");
10.         return this.sendLog(json);
11.     }
12. /**
13.      * Send Log to Rest Client and it be introduced in Database
14.      * @param jsonToSend Log JSON to Send
15.      * @return True if log was inserted correctly, false if not
16.      */
17.     @Override
18.     public boolean sendLog(JSONObject jsonToSend) {
19.         configureWebResource();
20.         RestClient.INSTANCE.setWebResources(webResource);
21.         return RestClient.INSTANCE.post(jsonToSend);
22.     }
23.       /**
24.      * Create Log
25.      * @param message Log's Message
26.      * @param level Log Level
27.      * @return Log JSON Object
28.      */
29.     @Override
30.     public JSONObject createJson(String message, String level) {
31.         JSONObject jsonPost = new JSONObject();
32.         ConfigurationRobot configurationRobot = ConfigurationController.INSTANCE.get();
33.         jsonPost.put("configurationID", configurationRobot.toJson())
34.                 .put("message", message)
35.                 .put("level", level)
36.                 .put("timeStamp", Utils.getTodayDateFormated())
37.                 .put("machineName", Utils.getMachineName());
38.
39.
40.         return jsonPost;
41.     }
42.
43.     /**
44.      * Set webResource with needed params.
45.      */
46.     private void configureWebResource()
47.     {
48.         webResource = new WebResource.WebResourceBuilder(Constants.HOST_BASE)
49.                 .puerto(Constants.PORT_BASE)
50.                 .appName(Constants.APP_NAME_BASE)
51.                 .resource(this.resource)
52.                 .build();
53.     }
```

9.2 Panell de control

9.2.1 Especificació

Els requisits funcionals definiran accions fonamentals que el nostre sistema ha de realitzar al rebre informació, processar-la i, posteriorment, produir certs resultats.

9.2.1.1 Anàlisi de requisits funcionals

	Número 1
Descripció	El panell de control ha de permetre iniciar i aturar les automatitzacions existents en les màquines on estiguin instal·lades.
Justificació del requisit	Per a poder executar les automatitzacions a través del panell de control es necessari afegir una opció que permeti iniciar i aturar-les des del navegador.
Condició de satisfacció	El panell de control es capaç de iniciar i aturar automatitzacions a través del navegador.

	Número 2
Descripció	El panell de control ha de de permetre registrar noves automatitzacions per a ser gestionades.
Justificació del requisit	Per a gestionar les automatitzacions aquestes s'han de poder donar de alta al panell de control.
Condició de satisfacció	El panell de control es capaç de registrar les noves automatitzacions creades.

	Número 3
Descripció	El panell de control ha de permetre veure les automatitzacions existents i en quin estat actualment (en funcionament, iniciant, parades, etc) i en quina maquina s'estan executant.
Justificació del requisit	Per a poder saber en tot moment quin és l'estat de les automatitzacions el panell de control ha d'oferir aquesta informació en tot moment.
Condició de satisfacció	El panell de control ofereix una vista amb totes les automatitzacions creades i el seu estat actual.

Número 4	
Descripció	El panell de control ha de permetre visualitzar els logs per cada automatització.
Justificació del requisit	Per a poder saber que es exactament el que ha estat fent cada automatització, el panell de control oferirà una vista que permeti visualitzar els logs.
Condició de satisfacció	El panell de control ofereix una vista amb tots els logs de cada automatització.

Número 5	
Descripció	El panell de control ha de permetre visualitzar afegir mòduls per a cada automatització creada de manera senzilla i rapida.
Justificació del requisit	Tot i que ja existeixen els logs per a visualitzar el registre del que ha estat fent cada automatització, ha d'estar organitzat de tal manera que permeti afegir mòduls nous per a les automatitzacions que vulguin mostrar nova informació que no esta contemplada inicialment en el panell.
Condició de satisfacció	El panell de control esta organitzat de manera modular i permet afegir noves vistes de manera senzilla i eficient.

Número 6	
Descripció	El panell de control ha de permetre visualitzar les configuracions de cada una de les automatitzacions creades.
Justificació del requisit	Per a saber tota la informació de cada automatització, el panell ha d'oferir la possibilitat de veure per a cada automatització la seva configuració, la maquina en que s'executa, quan es va començar a executar, etc.
Condició de satisfacció	El panell de control ofereix vistes mostrant les configuracions de les automatitzacions.

Número 7	
Descripció	El panell de control ha de permetre visualitzar la informació de tal manera que sigui possible copiar-la i enganxar-la en un Excel fàcilment.
Justificació del requisit	Per a tal de treballar en la informació generada per cada automatització s'ha de poder copiar en un Excel.
Condició de satisfacció	El panell de control ofereix vistes en format de taules que es poden copiar fàcilment a Excel.

Número 8	
Descripció	La informació del panell de control ha de ser fàcilment accessible per a les automatitzacions a través de serveis REST.
Justificació del requisit	Per tal de que les automatitzacions puguin obtindré la informació necessària per a la seva execució, han de poder consumir els serveis REST del panell de control.
Condició de satisfacció	El panell de control ofereix serveis REST consumibles des de les automatitzacions.

9.2.2.2 Anàlisi de requisits no-funcionals

Requisits de capacitat d'ús i humanitat

Número 9	
Tipus de requisit	10a. Requisits d'aparença
Descripció	El panell de control ha de complir amb els patrons de marca corporatius.
Justificació del requisit	Assegurar que l'aparença del producte s'ajusta a les expectatives d'everis.
Condició de satisfacció	Everis certificarà que el producte compleix els patrons corporatius.

Número 10	
Tipus de requisit	11b. Requisits d'estil
Descripció	Disseny conservador que mostri professionalitat
Justificació del requisit	L'estil ha de ser seriós i propi per tal de mostrar professionalitat. No ha de ser excessivament atrevit, ja que es busca la màxima funcionalitat, facilitat i intuitivitat en el seu ús.
Condició de satisfacció	Els usuaris del panell de control troben correcte l'estil dissenyat.

Requisits de capacitat d'ús i humanitat

Número 11	
Tipus de requisit	11a. Requisits de facilitat d'ús
Descripció	El sistema ha de ser eficient en el seu ús i que existeixi una bona retroalimentació a l'usuari. Complirà els criteris de temes de disseny, de contingut, d'estructura i de presentació fixat per el W3C.
Justificació del requisit	Cal que el producte sigui el més eficient possible en el seu ús per tal que els usuaris puguin aprendre a utilitzar-lo de manera ràpida i senzilla. També cal una bona retroalimentació perquè els usuaris tinguin confiança en el producte.
Condició de satisfacció	Els usuaris no troben traves en la usabilitat de l'aplicació i tenen confiança en què el producte fa el que realment s'espera.

Número 12	
Tipus de requisit	11b. Requisits de personalització i internacionalització.
Descripció	El sistema s'oferirà en anglès.
Justificació del requisit	L'anglès es l'idioma internacional més parlat, i per tal de que el framework pugui ser entens per tothom estarà desenvolupat en aquest idioma.
Condició de satisfacció	El 100% dels mètodes, classes i documentació estan escrits en anglès.

Requisits d'acompliment

Número 11	
Tipus de requisit	12a. Requisits de latència i velocitat
Descripció	La resposta del panell de control ha de ser suficientment ràpida per evitar la interrupció del flux d'accions de l'usuari.
Justificació del requisit	Un temps de resposta ràpid permetrà a l'usuari no perdre el flux o atenció del que esta fent amb el sistema.
Condició de satisfacció	El sistema respon de manera ràpida i no fa que l'usuari perdi l'atenció.

Número 12	
Tipus de requisit	12c. Requisits de precisió o exactitud.
Descripció	Totes les dates que s'incloguin en el panell de control tindran el següent format: DD/MM/AAAA
Justificació del requisit	És convenient especificar el format de la data per a que tota la informació segueixi el mateix format.
Condició de satisfacció	El format data i hora seguirà l'estàndard ISO -8601 extens d'estil Europeu.

Número 13	
Tipus de requisit	12d. Requisits de fiabilitat i disponibilitat
Descripció	El panell de control estarà disponible per a les 24 hores del dia durant els 365 dies que conformen l'any.
Justificació del requisit	Els usuaris han de poder iniciar o aturar les automatitzacions, consultar l'estat d'elles, revisar els registres, etc sempre que vulguin.
Condició de satisfacció	El sistema estarà disponible i completament funcional tot el temps.

Requisits operacionals i ambientals

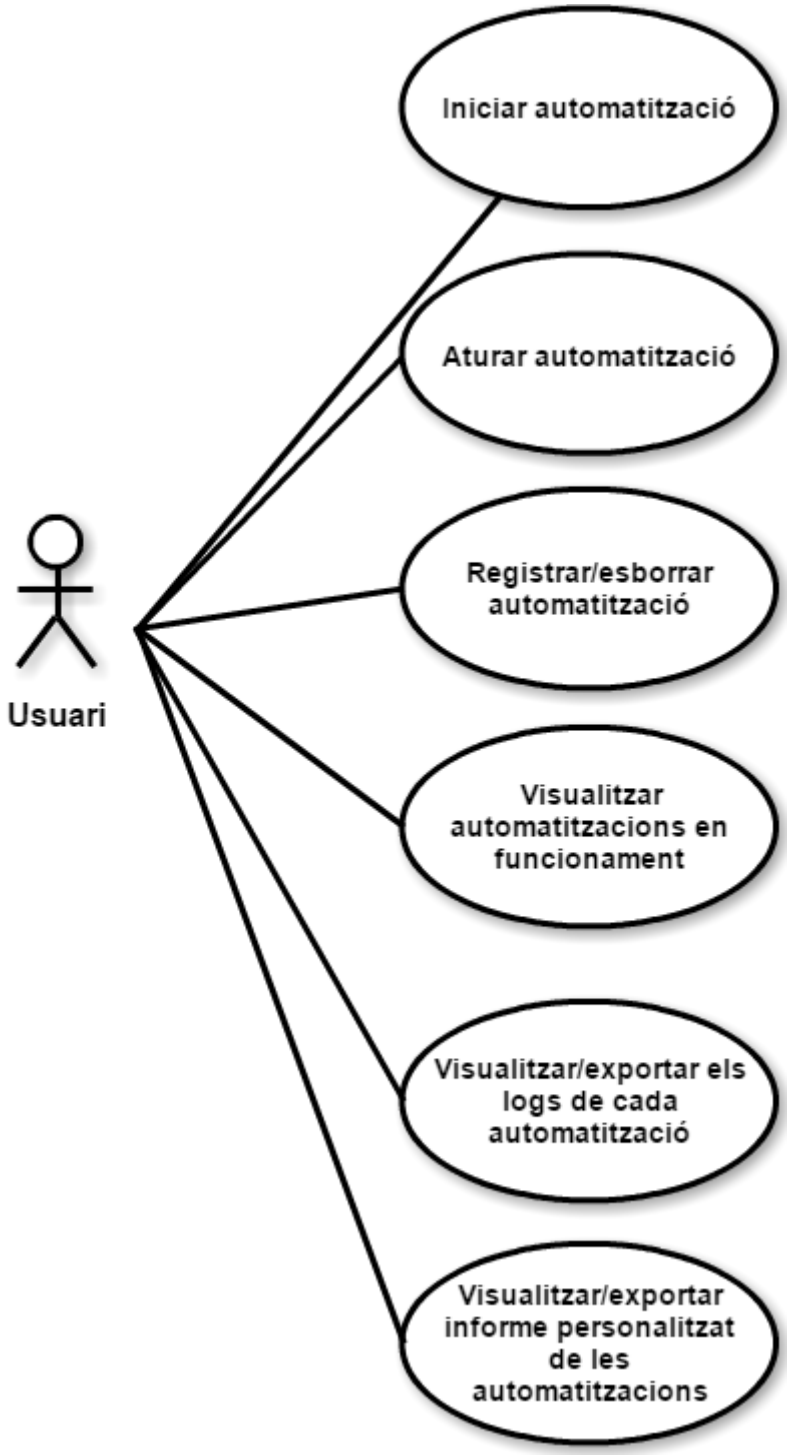
Número 15	
Tipus de requisit	13d. Requisits de llançament
Descripció	Cada nova versió no causarà la falla de característiques prèvies.
Justificació del requisit	El llançament d'una nova versió no pot fer que automatitzacions que funcionen correctament deixin de funcionar.
Condició de satisfacció	El framework no canviarà cap funcionalitat existent en versions prèvies del framework.

Requisits de preservació i suport

Número 16	
Tipus de requisit	14c. Requisits d'adaptabilitat
Descripció	El panell de control s'ha de poder veure correctament en els diferents navegadors webs més utilitzats i ser compatible amb Windows 7 i 10.
Justificació del requisit	El panell de control ha de poder ser utilitzat en els sistemes operatius de l'empresa, Windows 7 i Windows 10.
Condició de satisfacció	El panell de control es visualitzarà correctament amb el sistema operatiu Windows 7 i 10 amb els navegadors Internet Explorer, Edge, Chrome i Firefox.

9.2.2 Casos d'ús

9.2.2.1 Diagrama de casos d'ús



9.2.2.2 Descripció dels casos d'ús

Cas d'ús 1: Iniciar automatització

Actor principal: Usuari

Disparador: L'usuari vol iniciar una automatització

Escenari principal d'èxit:

1. L'usuari és dirigeix a la vista que mostra totes les automatitzacions.
2. L'usuari clica al boto d'iniciar automatització.
3. L'automatització s'inicia en l'ordinador on estigui instal·lat.
4. Fi del cas d'ús.

Extensions:

3a. L'ordinador no esta disponible per a executar l'automatització.

3a1. El panell de control mostra un missatge d'error i no s'executa l'automatització.

3a2. Fi del cas d'ús.

3b. L'execució no s'ha pogut executar correctament.

3b1. El panell de control mostrà un missatge d'error.

3b2. Fi del cas d'ús.

Cas d'ús 2: Aturar automatització

Actor principal: Usuari

Disparador: L'usuari vol parar una automatització en funcionament.

Escenari principal d'èxit:

1. L'usuari és dirigeix a la vista que mostra totes les automatitzacions existents.
2. L'usuari clica al boto de parar l'automatització que només apareix en automatitzacions iniciades.
3. L'automatització s'atura en l'ordinador on estigui instal·lat.
4. Fi del cas d'ús.

Extensions:

3a. L'ordinador no esta disponible per a parar l'automatització.

3a1. El panell de control mostra un missatge.

3a2. Fi del cas d'ús.

3b. L'automatització no s'ha pogut aturar correctament.

3b1. El panell de control mostrà un missatge d'error.

3b2. Fi del cas d'ús.

Cas d'ús 3: Registrar automatització

Actor principal: Usuari

Disparador: L'usuari vol registrar una nova automatització al panell de control.

Escenari principal d'èxit:

1. L'usuari és dirigeix a la vista que de creació d'automatitzacions.
2. L'usuari omple el formulari de registre de nova automatització.
3. El panell registra totes les dades a la base de dades.
4. Fi del cas d'ús.

Extensions:

2a. Hi ha camps buits o mal completats al formulari.

3a1. El panell de control mostra un missatge amb l'error que ha sorgit.

3a2. Fi del cas d'ús.

3a. El registre no s'ha pogut executar correctament.

3b1. El panell de control mostra un missatge d'error i es cancel·la el registre.

3b2. Fi del cas d'ús.

Cas d'ús 4: Esborrar automatització

Actor principal: Usuari

Disparador: L'usuari vol esborrar una automatització al panell de control.

Escenari principal d'èxit:

1. L'usuari és dirigeix a la vista que mostra les automatitzacions actuals.
2. L'usuari clica sobre el botó d'esborrar automatització.
3. El panell esborra totes les dades de la base de dades.
4. Fi del cas d'ús.

Extensions:

3a. Els registres no s'han pogut esborrar correctament.

3b1. El panell de control mostra un missatge d'error i es cancel·la el procés.

3b2. Fi del cas d'ús.

Cas d'ús 5: Visualitzar automatitzacions en funcionament

Actor principal: Usuari

Disparador: L'usuari vol visualitzar quines automatitzacions estan en funcionament.

Escenari principal d'èxit:

1. L'usuari és dirigeix a la vista que mostra les automatitzacions actuals.
2. L'usuari filtra per mostrar les automatitzacions amb l'estat executant-se.
3. El panell mostra les automatitzacions en execució.
4. Fi del cas d'ús.

Extensions:

3a. Els registres no s'han pogut mostrar correctament.

3b1. El panell de control mostra un missatge d'error i es cancel·la el procés.

3b2. Fi del cas d'ús.

Cas d'ús 6: Visualitzar logs de cada automatització

Actor principal: Usuari

Disparador: L'usuari vol visualitzar els logs de les automatitzacions.

Escenari principal d'èxit:

1. L'usuari és dirigeix a la vista que mostra els logs de les automatitzacions.
2. L'usuari filtra per mostrar els logs d'una automatització en concret.
3. El panell mostra els logs de l'automatització seleccionada.
4. Fi del cas d'ús.

Extensions:

3a. Els logs no s'han pogut mostrar correctament.

3b1. El panell de control mostra un missatge d'error i es cancel·la el procés.

3b2. Fi del cas d'ús.

Cas d'ús 7: Exportar logs de cada automatització

Actor principal: Usuari

Disparador: L'usuari vol exportar els logs de les automatitzacions.

Escenari principal d'èxit:

1. L'usuari és dirigeix a la vista que mostra els logs de les automatitzacions.
2. L'usuari filtra per mostrar els logs d'una automatització en concret.
3. El panell mostra els logs de l'automatització seleccionada.
4. L'usuari clica al botó d'exportar.
5. El panell genera un Excel amb els logs seleccionats.
5. Fi del cas d'ús.

Extensions:

3a. Els logs no s'han pogut mostrar correctament.

3b1. El panell de control mostra un missatge d'error i es cancel·la el procés.

3b2. Fi del cas d'ús.

5a. L'Excel no s'ha pogut generar correctament.

3b1. El panell de control mostra un missatge d'error i es cancel·la el procés.

3b2. Fi del cas d'ús.

Cas d'ús 8: Visualitzar informe personalitzat de cada automatització

Actor principal: Usuari

Disparador: L'usuari vol visualitzar l'informe personalitzat d'una automatització.

Escenari principal d'èxit:

1. L'usuari és dirigeix a la vista que mostra l'informe de la automatització en concret.
2. El panell mostra l'informe de l'automatització seleccionada.
3. Fi del cas d'ús.

Extensions:

2a. La informació no s'ha pogut mostrar correctament.

3b1. El panell de control mostra un missatge d'error i es cancel·la el procés.

3b2. Fi del cas d'ús.

Cas d'ús 9: Exportar informe personalitzat de cada automatització

Actor principal: Usuari

Disparador: L'usuari vol visualitzar l'informe personalitzat d'una automatització.

Escenari principal d'èxit:

1. L'usuari és dirigeix a la vista que mostra l'informe de la automatització en concret.
2. El panell mostra l'informe de l'automatització seleccionada.
3. L'usuari clica al botó de generar informe Excel.
4. El panell genera l'Excel amb la informació de l'automatització.
5. Fi del cas d'ús.

Extensions:

2a. La informació no s'ha pogut mostrar correctament.

3b1. El panell de control mostra un missatge d'error i es cancel·la el procés.

3b2. Fi del cas d'ús.

5a. L'Excel no s'ha pogut generar correctament.

3b1. El panell de control mostra un missatge d'error i es cancel·la el procés.

3b2. Fi del cas d'ús.

9.2.3 Arquitectura global del panell de control

En aquest apartat tractarem l'arquitectura del nostre sistema.

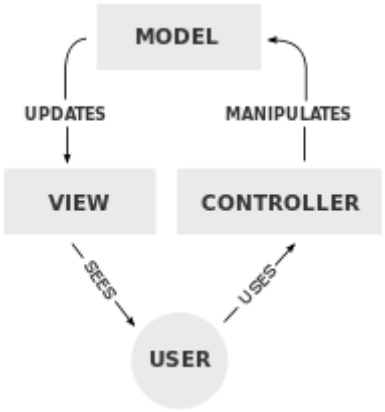
9.2.3.1 Patró de disseny escollit: MVC

El patró de disseny que s'ha escollit per a aquest projecte ha estat el patró de Model-Vista-Controlador, ja que és el patró més conegut per a desenvolupar pàgines web i a més permet obtenir un resultat molt modular.

El patró MVC consta de tres parts:

- Model: És la capa responsable de la definició de les dades.
- Vista: És la capa responsable de mostrar les dades a l'usuari.
- Controlador: És la part de codi que conté totes les interaccions entre Model i Vista.

A més, utilitzarem Services, que ens serviran per a comunicar-nos amb la base de dades del nostre sistema. Per tant, els Services ens ompliran els models definits al controlador, per a què aquest els utilitzi, internament o mostrant-los a l'usuari en qualsevol vista. [R16]



9.2.3.1 Tecnologia i eines usades pel desenvolupament

Back-End:

Per al desenvolupament de la part del servidor s'utilitzaran les següents tecnologies i eines:

- **Arquitectura REST**

  Com hem mencionat prèviament, és la tecnologia que utilitzarem per a crear els serveis que faran possible la comunicació entre el client i el servidor.

  S'ha escollit aquesta arquitectura per que permet dividir la part del client amb la del servidor i això es un punt a favor molt important perquè cap la possibilitat que en un futur canviï l'entorn on estigui instal·lat el servidor.

- **Java EE**

  Per a desenvolupar els serveis basats en una arquitectura REST usarem Java EE.

  Java EE és una plataforma de programació (una de les Plataformes Java) per desenvolupar i executar programari escrit amb el llenguatge Java amb una arquitectura distribuïda amb nivells, basada en components de programari, tot plegat executant-se en un servidor d'aplicacions. Java EE proporciona unes eines per a la creació de serveis web anomenat JAX-RS.

- **JAX-RS**

  JAX-RS: Java API for RESTful Web Services es una API del llenguatge que proporciona suport per a la creació de serveis web d'acord amb l'arquitectura Representational State Transfer (REST).

  JAX-RS usa anotacions, introduïdes des de la versió Java SE 5, para simplificar el desenvolupament y desplegament dels clients y punts finals dels serveis web.

  A partir de la versió 1.1 en adelante, JAX-RS es una parte oficial de Java EE 6. Una característica notable de ser part oficial de Java EE es que no es requereix cap configuració per començar a usar JAX-RS.

- **SQL Server Management 2014**

  Per a gestionar les bases de dades usarem el SGBD de l'empresa Microsoft anomenat SQL Server Managament que és el programari proporcionat per a everis.

  A part de la creació i visualització de les taules, permet la programació de backups automàtics, importar i exportar taules, generar scripts per la migració a producció i altres serveis també molt interessants.

Glassfish Server

GlassFish és un servidor d'aplicacions de programari lliure desenvolupat per Sun Microsystems, companyia adquirida per Oracle Corporation, que implementa les tecnologies definides en la plataforma Java EE i permet executar aplicacions que segueixen aquesta especificació. És gratuït, de codi lliure i es distribueix sota un llicenciament dual a través de la llicència CDDL i la GNU GPL [R17]

A través de glassfish llançarem els serveis REST desenvolupats.

Front-End

- **HTML + CSS**

HTML (acrònim d'Hyper Text Markup Language, en català, "llenguatge de marcat d'hipertext"), és un llenguatge de marcat que deriva de l'SGML dissenyat per estructurar textos i relacionar-los en forma d'hipertext. Gràcies a Internet i als navegadors web, s'ha convertit en un dels formats més populars que existeixen per a la construcció de documents per a la web.

Cascading Style Sheets (CSS, en català: Fulls d'Estil en Cascada) és un llenguatge de fulls d'estil utilitzat per descriure la semàntica de presentació (l'aspecte i format) d'un document escrit en un llenguatge de marques. La seva aplicació més comuna és dissenyar pàgines web escrites en HTML i XHTML, però el llenguatge també pot ser aplicat a qualsevol classe de document XML, incloent-hi SVG i XUL.

CSS està dissenyat principalment per permetre la separació de contingut del document (escrit en HTML o un llenguatge de marques similar) de la presentació del document, incloent-hi elements com la disposició, colors, i fonts. Aquesta separació pot millorar l'accessibilitat al contingut, proporcionar més flexibilitat i control en l'especificació de característiques de presentació, permetre que múltiples pàgines comparteixin un format comú, i redueix complexitat i repetició en el contingut estructural (com per exemple al permetre disseny web sense taules). CSS també pot deixar la mateixa pàgina de marques ser presentada en estils diferents mitjançant mètodes de render diferents, com a la pantalla, en impressió, per veu (quan és llegida en veu alta per un navegador amb lector o pantalla lectora) i amb mecanismes tàctils amb sistemes Braille. Mentre que l'autor d'un document típicament associa els documents amb un full d'estil CSS, els lectors poden utilitzar un full d'estil diferent, potser un al seu propi ordinador, per invalidar aquell que l'autor ha especificat.

- **Angular JS**

Angular JS és un framework de JavaScript de codi obert, mantingut per Google, que s'utilitza per a crear i mantenir aplicacions web d'una sola pàgina. La biblioteca llegeix HTML, que conte atributs de les etiquetes personalitzades addicionals, i llavors obeeix a les directives d'aquests atributs i uneix les peces d'entrada o de sortida de la pàgina a un model representat per les variables estàndards de JavaScript.

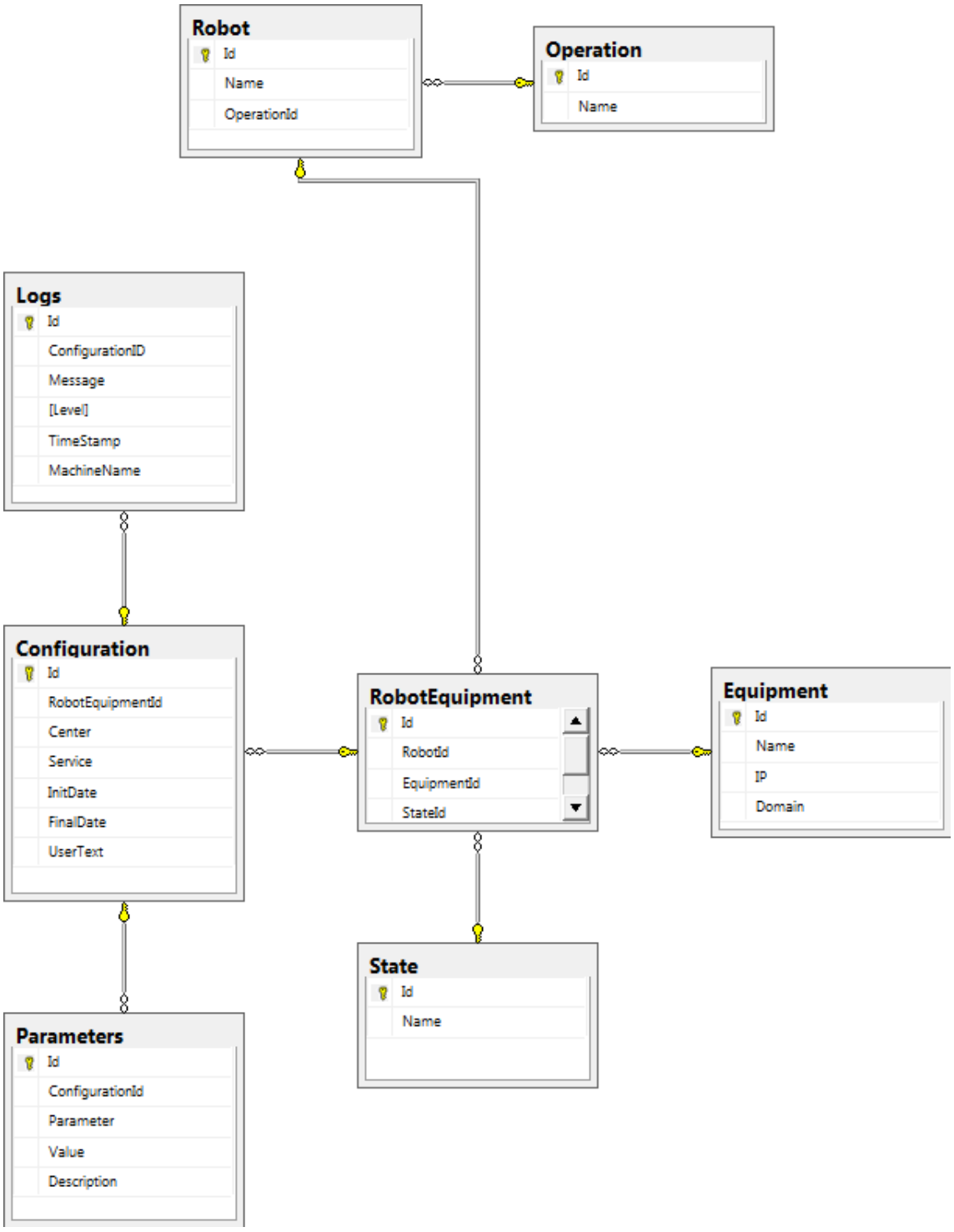
S'utilitza per a mostrar un contingut dinàmic a la web.

- **IIS (Internet Information Services)**

Internet Information Services o IIS és un servidor web i un conjunt de serveis per al sistema operatiu Microsoft Windows. Originalment era part del Option Pack per a Windows NT. 2003. Aquest serà el servidor usat per llançar el front-end del panell de control.

9.2.4 Model relacional de la base de dades

Seguidament es mostra una imatge que descriu les taules que conformen la base de dades del panell de control i una descripció de que representa cada una d'elles.



9.2.4.1 Descripció de les taules

Taula Operations

Entenem com a operació cada operativa que ofereix el projecte everis BPO de l'entitat financera. Aquesta taula representa cada un d'aquestes operatives existents en el projecte.

Taula Robot

La taula Robot representa cada automatització creada, amb el seu id, el seu nom i l'operació sobre la que treballa.

Taula State

La taula State representa els estats per els quals pot passar una automatització. Per exemple: activa, no activa, iniciant, en execució, etc.

Taula Equipment

La taula equipment representa les maquina sobre la quals es poden executar automatitzacions. Guarda les dades per a que sigui possible l'execució remota, com és la IP, el domini de la maquina i també el nom per a que sigui fàcilment identificable.

Taula Robot Equipment

La taula robot equipment representa la unió de les taules Robot, State i Equipment. És a dir, que aquesta taula relaciona una automatització amb la màquina en que s'executa i l'estat en que es troba actualment.

Taula Configuration

La taula configuration guarda la informació que configura una automatització. Els paràmetres són interns per a l'ús d'everis però volen representar la oficina des d'on s'executa l'automatització, el servei que l'executa i l'inici i fi de l'execució de les automatitzacions.

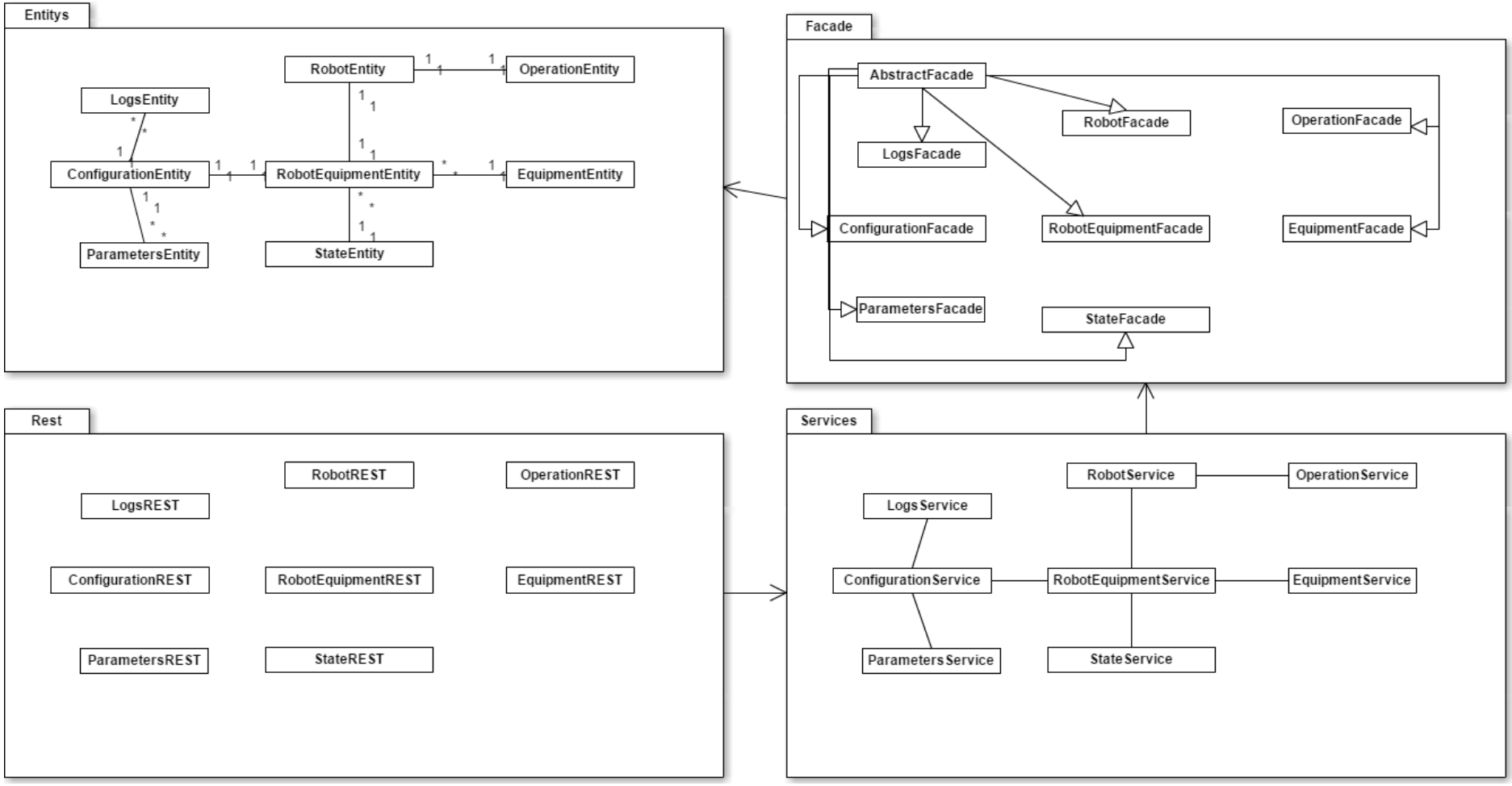
Taula Parameters

La taula parameters esta estrictament relacionada amb configuració. De vegades les automatitzacions requereixen de configuracions addicionals a les de la taula Configuration, i per això es va crear aquesta taula que representa aquests nous paràmetres.

Logs

La taula logs guarda tots els logs creats per les automatitzacions. Es relaciona amb les automatitzacions a través de la seva configuració.

9.2.5 Diagrama de classes del Back-End



9.2.5.1 Descripció del sistema

Package Entitys

El paquet Entity conté les classes que representen les taules de la base de dades mencionades anteriorment. Cada classe representa una taula i és anomenada Entity, i inclou tots els seus atributs, claus primàries i relacions.

Package Facades

El paquet Facades conté un conjunt de classes que implementen un patró de disseny anomenat Facade.

El patró facade s'aplica quan es necessita proporcionar una interfície simple per un subsistema complex, o quan es vol estructurar diferents subsistemes en capes, ja que els Facade seran els punts d'entrada a aquests nivells.

En aquest sentit el subsistema complex són les entitats i les relacions entre sí i tots els facade implementen una classe abstracta anomenada AbstractFacade que proporciona mètodes per a realitzar consultes a les entitats, crear-ne de noves, esborrar-ne, entre d'altres. És a dir, són el punt d'entrada per a la inserció/modificació de les entitats existents.

Package Services

El paquet Services son uns contenidors de funcions que fan d'intermediari entre el paquet Facade i el paquet REST. S'utilitzen perquè gràcies a una anotació de JAX-RS permeten dividir les interaccions amb la base de dades en transaccions, i això permet la opció de cancel·lar una transacció i tornar a l'estat original en cas que sorgeixin errors durant l'execució.

Package REST

El paquet REST conté les classes que usen una anotació JAX-RS per a crear serveis web. Cada REST esta associat a una entitat, tot i que poden retornar informació de més d'una. Serveix per a identificar clarament sobre quina entitat es treballa.

Tal i com hem explicat abans, el REST crida a les classes del paquet Services, que fara una crida als facades per a executar operacions sobre les entitats. El rest no conté lògica, simplement realitza crides als serveis. Això permet una separació molt diferencia de les capes, permeten una escalabilitat i una modularitat molt important.

9.2.6 Implementació del back-end

Seguidament, es mostrarà la implementació de part d'una Entity, un Facade, un service i un REST.

State Entity

Les anotacions que comencen per @ són anotacions de JAX-RS i serveixen per a especificar quin tipus d'element és, les relacions, etc.

```
1.  @Entity
2.  @Table(name = "State")
3.  @XmlRootElement
4.  @NamedQueries({
5.      @NamedQuery(name = "StateEntity.findAll", query = "SELECT s FROM StateEntity s"),
6.      @NamedQuery(name = "StateEntity.findById", query = "SELECT s FROM StateEntity s
    WHERE s.id = :id"),
7.      @NamedQuery(name = "StateEntity.findByName", query = "SELECT s FROM StateEntity s
    WHERE s.name = :name")})
8.  public class StateEntity implements Serializable {
9.
10.     private static final long serialVersionUID = 1L;
11.     @Id
12.     @GeneratedValue(strategy=GenerationType.IDENTITY)
13.     @Column(name = "Id")
14.     private Integer id;
15.     @Size(max = 50)
16.     @Column(name = "Name")
17.     private String name;
18.     @OneToMany(cascade = CascadeType.ALL, mappedBy = "stateId")
19.     private Collection<RobotEquipmentEntity> robotEquipmentEntityCollection;
20.
21.     public StateEntity() {
22.     }
23.
24.     public StateEntity(Integer id) {
25.         this.id = id;
26.     }
27.
28.     public Integer getId() {
29.         return id;
30.     }
31.
32.     public void setId(Integer id) {
33.         this.id = id;
34.     }
35.
36.     public String getName() {
37.         return name;
38.     }
39.
40.     public void setName(String name) {
41.         this.name = name;
42.     }
43.
44.     @XmlTransient
45.     public Collection<RobotEquipmentEntity> getRobotEquipmentEntityCollection() {
46.         return robotEquipmentEntityCollection;
47.     }
48.
49.     public void setRobotEquipmentEntityCollection(Collection<RobotEquipmentEntity> robo
    tEquipmentEntityCollection) {
50.         this.robotEquipmentEntityCollection = robotEquipmentEntityCollection;
51.     }
```

Abstract Facade i State Facade

La classe Abstract Facade conté les funcionalitats bàsiques que han d'implementar totes les classes Facade, i com és pot observar seguidament conté mètodes per a crear , editar, eliminar i buscar entitats. A sota es pot veure la implementació de la classe StateFacade heretant del Abstract Facade. Bàsicament el que fa es especificar sobre quina entitat s'ha de treballar.

```
1.
2.  /**
3.   *
4.   * @author asantapi
5.   */
6.  public abstract class AbstractFacade<T> {
7.
8.      private Class<T> entityClass;
9.
10.     public AbstractFacade(Class<T> entityClass) {
11.         this.entityClass = entityClass;
12.     }
13.
14.     protected abstract EntityManager getEntityManager();
15.
16.     public void create(T entity) {
17.         getEntityManager().persist(entity);
18.     }
19.
20.     public void edit(T entity) {
21.         getEntityManager().merge(entity);
22.     }
23.
24.     public void remove(T entity) {
25.         getEntityManager().remove(getEntityManager().merge(entity));
26.     }
27.
28.     public T find(Object id) {
29.         return getEntityManager().find(entityClass, id);
30.     }
31.
32.     public List<T> findAll() {
33.         javax.persistence.criteria.CriteriaQuery cq = getEntityManager().getCriteriaBui
    lder().createQuery();
34.         cq.select(cq.from(entityClass));
35.         return getEntityManager().createQuery(cq).getResultList();
36.     }
```

State Facade

```
1.  @Stateless
2.  @Path("state")
3.  public class StateFacade extends AbstractFacade<StateEntity>
4.  {
5.
6.      @PersistenceContext(unitName = "com.everis_PanelControl_war_1.0PU")
7.      private EntityManager em;
8.
9.      public StateFacade() {
10.         super(StateEntity.class);
11.     }
12.
13.    @Override
14.     protected EntityManager getEntityManager() {
15.         return em;
16.     }
17.
18. }
```

State Services

La classe State Service, tal i com hem explicat anteriorment només fa d'intermediari entre el REST i el facade, dotant de transaccions les interaccions amb les entitats gràcies a les anotacions en JAX-RS.

```
1.  /**
2.   *
3.   * @author asantapi
4.   */
5.  @Stateless
6.  public class StateService
7.  {
8.
9.      @EJB
10.     private StateFacade stateFacade;
11.
12.     public void create(StateEntity entity)
13.     {
14.         stateFacade.create(entity);
15.     }
16.
17.     public void edit(@PathParam("id") Integer id, StateEntity entity)
18.     {
19.         stateFacade.edit(entity);
20.     }
21.
22.     public void remove(@PathParam("id") Integer id)
23.     {
24.         stateFacade.remove(stateFacade.find(id));
25.     }
26.
27.     public StateEntity find(@PathParam("id") Integer id)
28.     {
29.         return stateFacade.find(id);
30.     }
31.
32.     public List<StateEntity> findAll()
33.     {
34.         return stateFacade.findAll();
35.     }
36.
37.     public List<StateEntity> findRange(@PathParam("from") Integer from,
    @PathParam("to") Integer to)
38.     {
39.         return stateFacade.findRange(new int[]{from, to});
40.     }
41.
42.     public String countREST()
43.     {
44.         return String.valueOf(stateFacade.count());
45.     }
46. }
```

State REST

Per últim tenim la classe State Rest, que defineix el servei web, com s'hi accedirà (amb l'anotació @path) i injecta el servei per a poder realitzar totes les transaccions.

També cal destacar que s'ha d'especificar quin tipus de servei rest hi haurà. Per exemple, els @GET permeten extreure informació, els @POST permeten crear noves entitats (inserir noves dades a la base de dades) i els @PUT permeten actualitzar les dades de les entitats.

A part també s'especifica els paràmetres que es rebran a través de la URL amb les anotacions @PathParam o @QueryParam i a quin tipus d'objecte s'assignaran (Integer, String...)

```
1.  /**
2.   *
3.   * @author asantapi
4.   */
5.  @Stateless
6.  @Path("state")
7.  public class StateRest
8.  {
9.
10.     @Inject
11.     private StateService stateService;
12.
13.     @POST
14.     @Consumes({MediaType.APPLICATION_JSON})
15.     public void create(StateEntity entity)
16.     {
17.         stateService.create(entity);
18.     }
19.
20.     @PUT
21.     @Path("{id}")
22.     @Consumes({MediaType.APPLICATION_JSON})
23.     public void edit(@PathParam("id") Integer id, StateEntity entity)
24.     {
25.         stateService.edit(id, entity);
26.     }
27.
28.     @DELETE
29.     @Path("{id}")
30.     public void remove(@PathParam("id") Integer id)
31.     {
32.         stateService.remove(id);
33.     }
34.
35.     @GET
36.     @Path("{id}")
37.     @Produces({MediaType.APPLICATION_JSON})
38.     public StateEntity find(@PathParam("id") Integer id)
39.     {
40.         return stateService.find(id);
41.     }
42.
43.     @GET
44.     @Produces({MediaType.APPLICATION_JSON})
45.     public List<StateEntity> findAll()
46.     {
47.         return stateService.findAll();
48.     }
```

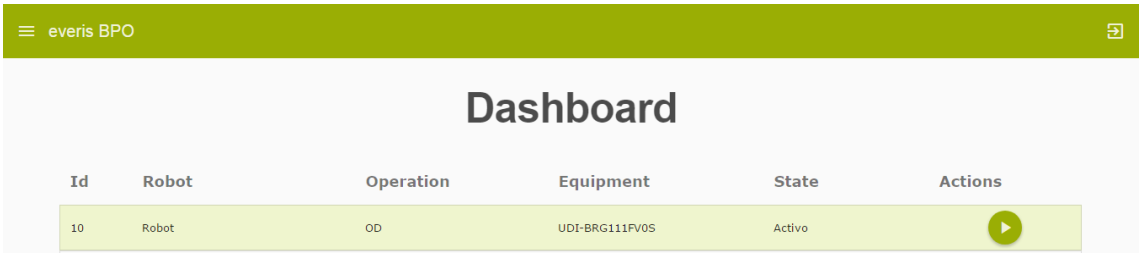
9.2.7 Vistes del Front-End

Un cop desenvolupat el back-end del panell de control i escollida l'arquitectura MVC + Services que usarà el Front – End falta implementar la part del client del panell de control.

A continuació es mostraran les vistes creades més importants i en el següent apartat una part del codi que s'ha usat per a crear-les.

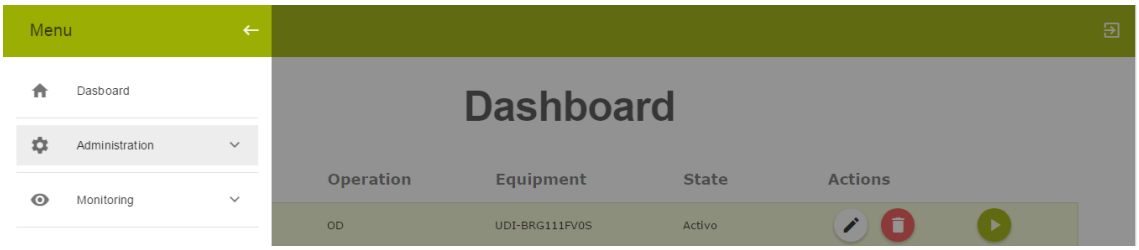
Vista principal – Dashboard

El dashboard es la vista principal del panell de control i mostra les execucions que existeixen i l'estat en que es troben actualment. A més també incorpora un boto per a iniciar les automatitzacions, o parar-les en cas de estar ja iniciades.



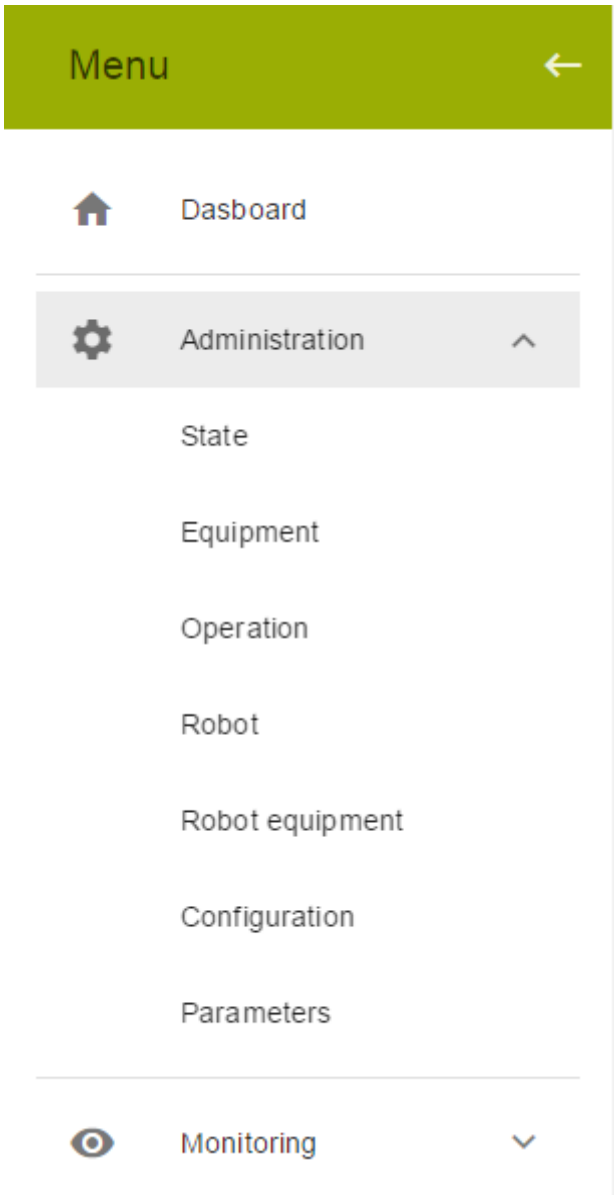
Menú lateral desplegable

A més el panell incorpora un menú desplegable a l'esquerra que permet navegar per a totes les opcions de les que disposa. A continuació es mostren una imatge del menú desplegable sobre el panell.

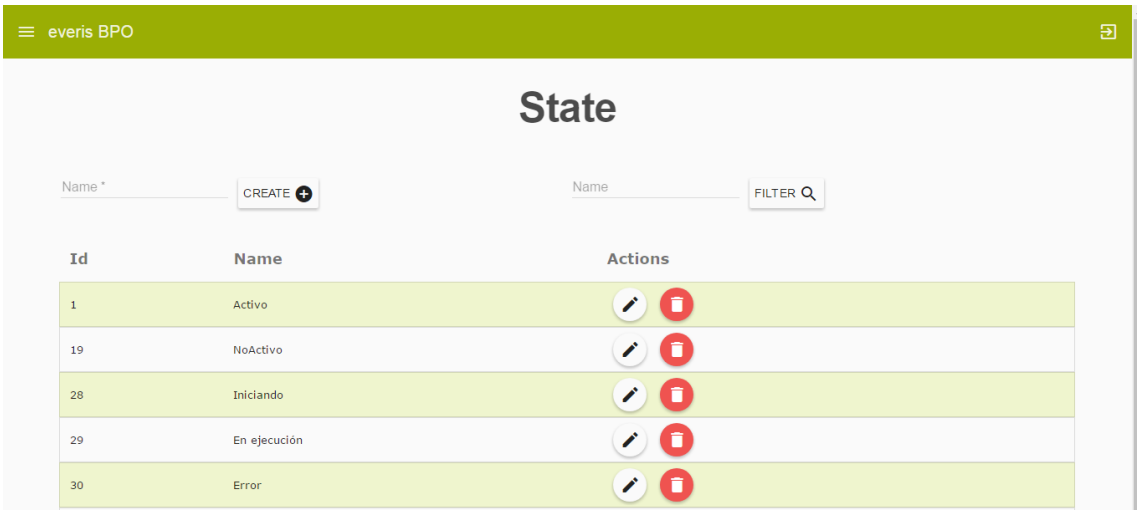


Aquí es pot veure el menú desplegable amb totes les opcions que incorpora.

Cada secció de l'apartat Administration dirigeix a una vista que permet afegir/modificar/eliminar dades de la taula que representen. A continuació es mostrarà la vista de State com a exemple de com són aquestes vistes.



Vista State

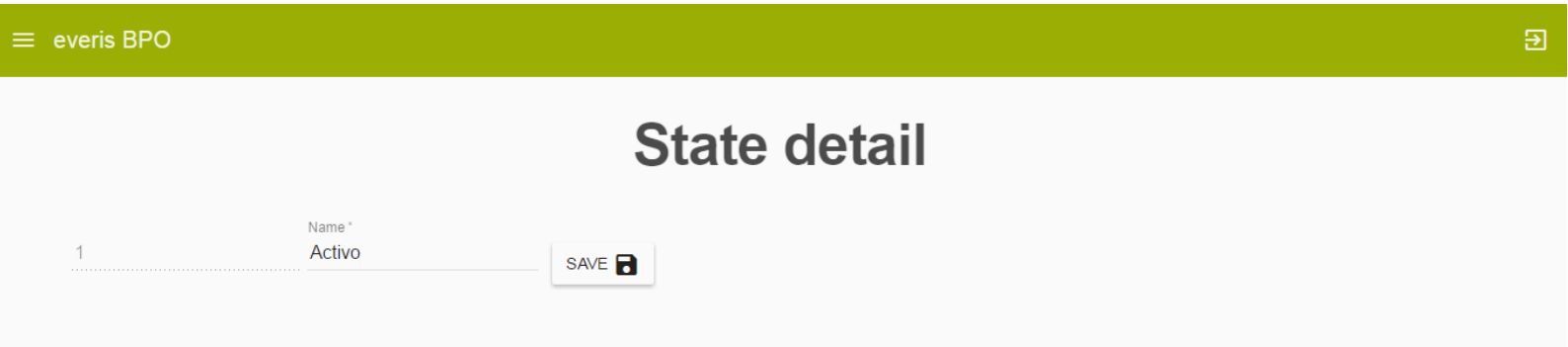


Aquesta vista segueix el mateix patró que totes les vistes que apareixen al menú vist anteriorment. A l'esquerra podem visualitzar tots els camps necessaris per a crear un estat nou, que en aquest cas simplement és el nom de l'estat.

A la dreta de la imatge podem veure que existeix un filtre per a seleccionar només les dades que volem mostrar.

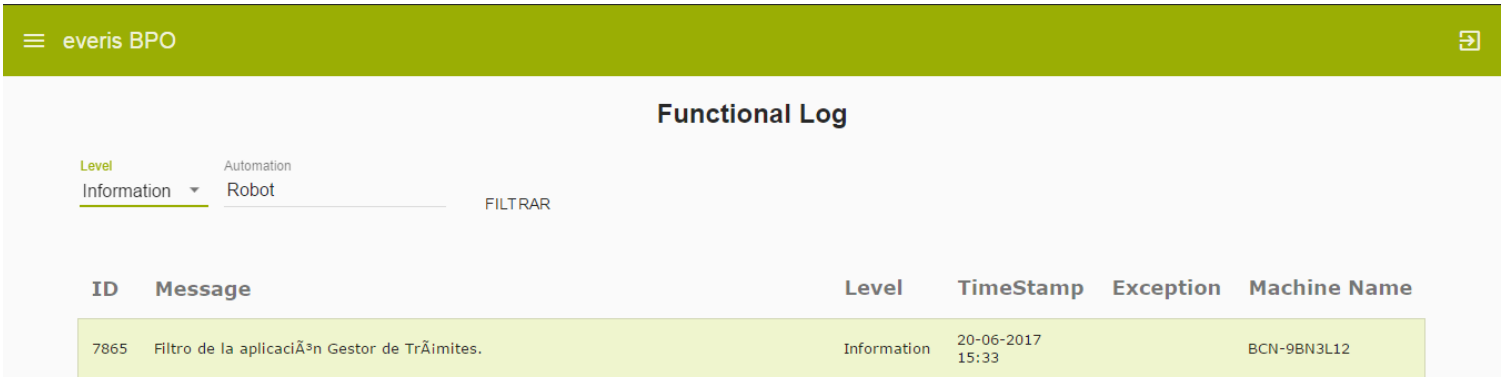
Per últim podem veure una taula que mostra els estats que existeixen i les accions que es poden realitzar amb cada estat, que en aquest cas són editar i esborrar.

En el cas que es vulgui editar, clicant al botó s'obre una vista de detalls que permet l'edició de manera senzilla.



Vista Logs

La vista dels logs ajunta tots els logs creats per les automatitzacions i permet filtrar per l'automatització que l'usuari desitgi.



Com podem veure a la imatge es mostra els logs de nivell informació de l'automatització que es mostrava al dashboard anomenada Robot. A més aporta informació respecte a l'hora en que s'ha creat, si hi ha hagut alguna excepció i en quina màquina s'ha executat.

9.2.8 Implementació del front-end

A continuació es mostrarà la implementació de la vista State mostrarà anteriorment. Totes els mòduls es componen d'una vista, un controlador que controla la lògica de la vista i model i per últim un service que accedeix al model de l'aplicació.

Vista:

```
<h1>State</h1>

<div layout="row">
    <md-input-container>
        <label>Name</label>
        <input type="text" ng-model="state.name" required="true">
    </md-input-container>
    <md-input-container>
        <md-button class="md-raised" ng-click="vm.create(state)">create <md-
icon>add_circle</md-icon></md-button>
    </md-input-container>

    <span flex></span>

    <md-input-container>
        <label>Name</label>
        <input type="text" ng-model="filter.name">
    </md-input-container>
    <md-input-container>
        <md-button class="md-raised" ng-click="vm.filter(filter)">filter <md-
icon>search</md-icon></md-button>
    </md-input-container>

    <span flex></span>
</div>

<md-data-table-container ng-if="vm.list.length > 0">
    <!-- style="width: 700px; align-self: center;" -->
    <table md-table md-data-table md-progress="deferred">
        <thead md-head>
            <tr md-row>
                <th md-column>Id</th>
                <th md-column>Name</th>
                <th md-column>Actions</th>
            </tr>
        </thead>
        <tbody>
            <tr md-row md-select="i" ng-repeat="i in vm.list">
                <td md-cell>{{i.id}}</td>
                <td md-cell>{{i.name}}</td>
                <td md-cell layout="row">
                    <md-button class="md-raised md-fab md-mini" ng-click="vm.edit(i)">
                        <md-icon>edit</md-icon>
                        <md-tooltip>Editar</md-tooltip>
                    </md-button>
                    <md-button class="md-warn md-fab md-mini" ng-click="vm.delete(i)">
                        <md-icon>delete</md-icon>
                        <md-tooltip>Borrar</md-tooltip>
                    </md-button>
                </td>
            </tr>
        </tbody>
    </table>
</md-data-table-container>

<p ng-if="vm.list.length <= 0">No avaliable states.</p>
```

Controlador

Desenvolupat en Angular JS, proporciona la lògica entre la vista i el model. Les funcions més importants serien getData, que executa la consulta a través del model per a retornar les dades a la vista, i create, que comunica amb el model amb la nova informació del nou objecte.

```
1.  (function() {
2.      'use strict';
3.
4.      angular
5.          .module('app.state')
6.          .controller('stateController', stateController);
7.
8.      stateController.$inject = ['$injector', '$location'];
9.
10.     /* @ngInject */
11.     function stateController($injector, $location) {
12.         var vm = this;
13.         vm.service = $injector.get('stateService');
14.
15.         vm.getData = function() {
16.             vm.service.all().$promise.then(function(result) {
17.                 vm.list = result;
18.             });
19.         };
20.
21.         vm.filter = function() {
22.             vm.service.filtered(vm.filter).$promise.then(function(result) {
23.                 vm.list = result;
24.             });
25.         };
26.
27.         vm.create = function(state){
28.             vm.service.createState(state).$promise.then(function() {
29.                 console.log("creado OK");
30.                 vm.getData();
31.             }, function() {
32.                 console.log("creado KO");
33.                 vm.getData();
34.             });
35.         }
36.
37.         vm.edit = function(state) {
38.             $location.path('/state/detail/' + state.id);
39.         }
40.
41.         vm.delete = function(state) {
42.             vm.service.deleteState(state).$promise.then(function() {
43.                 console.log("delete OK");
44.                 vm.getData();
45.             }, function() {
46.                 console.log("delete KO");
47.                 vm.getData();
48.             });
49.         }
50.
51.         vm.getData(); //cargar datos iniciales
52.     }
53.
54. })();
```

Model

Desenvolupat en angular JS, conté un seguit de funciones que comuniquen amb el rest creat per al panell de control, i retornen les dades que es demanin.

```
1.  (function() {
2.      'use strict';
3.
4.      angular
5.          .module('app.state')
6.          .service('stateService', stateService);
7.
8.      stateService.$inject = ['REST'];
9.
10.     /* @ngInject */
11.     function stateService(REST) {
12.         this.all = all;
13.
14.         function all() {
15.             return REST.State.query();
16.         };
17.
18.         this.byId = byId;
19.
20.         function byId(idFind) {
21.             return REST.State.get({id: idFind});
22.         }
23.
24.         this.filtered = filtered;
25.
26.         function filtered(params) {
27.             return REST.State.query(params);
28.         }
29.
30.         this.createState = createState;
31.
32.         function createState(state) {
33.             return REST.State.save(state);
34.         }
35.
36.         this.editState = editState;
37.
38.         function editState(state) {
39.             return REST.State.update({id: state.id}, state);
40.         }
41.
42.         this.deleteState = deleteState;
43.
44.         function deleteState(state) {
45.             return REST.State.delete({id: state.id});
46.         }
47.     }
48.
49. })();
```

9.3 Creació d'una automatització

La creació d'una automatització no l'he pogut desenvolupar jo sol, sinó que he necessitat l'ajuda d'un empleat d'un altre departament que m'ha explicat els diferents processos que existeixen, hem valorat quin era un bon candidat a ser automatitzat i ha fet d'intermediari amb el banc per que ens aproves el disseny tècnic i funcional.

Per això, adjuntaré a l'annex el document que hem creat conjuntament i que ha sigut validat per el banc. Aquest esta fet en castellà perquè la comunicació es realitza amb aquest idioma.

Seguidament, mostraré la implementació d'una de les classes desenvolupades per a l'automatització. Concretament mostraré una que s'encarrega de interactuar amb el gestor de correus Outlook.

```
1.  /**
2.   *
3.   * @author asantapi
4.   */
5.  public class ControllerOutlook {
6.
7.      private final String plantilla;
8.      private final FunctionalLog functionalLog;
9.      private String mensajeLog;
10.
11.     public ControllerOutlook() {
12.         this.plantilla = "Buenos días,\n\n"
13.                 + "Del trámite de referencia, salvo error, no hemos recibido la
    conformidad de firma. Necesitamos que nos informéis urgentemente de la situación del
    mismo. Si se firmó, necesitaremos urgentemente que nos lo confirméis con el fin de
    poder incorporarlo. \n\n"
14.                 + "En caso de no recibir respuesta en dos días laborables, daremos el
    trámite por finalizado.\n\n"
15.                 + "Gracias y saludos.";
16.         functionalLog = new FunctionalLog();
17.
18.     }
19.
20.     public void iniciarOutlook() {
21.         if (DesktopController.existImage("imgs\\outlook\\SibisOutlook.png", 5)) {
22.             DesktopController.clickImage("imgs\\outlook\\SibisOutlook.png", 2, 2);
23.             if (DesktopController.existImage("imgs\\outlook\\btAccesoCorreo.png", 5)) {
24.                 DesktopController.clickImage("imgs\\outlook\\btAccesoCorreo.png", 2, 2)
    ;
25.             }
26.         }
27.         Utils.sleep(5);
28.     }
29.
30.     public boolean enviarCorreoOficina(Tramite tramite) throws SystemCallError {
31.
32.         String referenciaTramite = tramite.getReferencia();
33.         boolean correoEnviadoCorrectamente = false;
34.         mensajeLog = "Correo del trámite con referencia " + referenciaTramite + "
    enviado correctamente.";
35.
36.         if (clickBotonNuevoCorreo(referenciaTramite)) {
37.             // Copiar Correo Oficina
38.             Utils.sleep(5);
39.             Clipboard.empty();
40.             Clipboard.setData(correoOficina, Clipboard.TEXT);
41.             KeyboardController.sendKey(KeyList.V, KeyList.CTRL, 3);
42.
43.
```

```
44.             // Copiar Numero Alta Oficina
45.             Utils.sleep(5);
46.             Clipboard.empty();
47.             Clipboard.setData(numeroAltaOficina, Clipboard.TEXT);
48.
49.             KeyboardController.sendKey(KeyList.TAB, 2, 2);
50.             KeyboardController.sendKey(KeyList.V, KeyList.CTRL, 3);
51.
52.             KeyboardController.sendKey(KeyList.TAB, 2, 2);
53.             KeyboardController.pasteText("PRUEBA Finalizar trámite
    " + referenciaTramite + " " + nombreCompleto, 3);
54.             KeyboardController.sendKey(KeyList.TAB, 5);
55.             KeyboardController.pasteText(plantilla, 2);
56.
57.             if (clickBotonOpciones(referenciaTramite)) {
58.                 if (configurarReplyTo(referenciaTramite, agente.getCorreoBanc())) {
59.                     if (DesktopController.existImage("imgs\\outlook\\btEnviar.png", 5))
     {
60.                         DesktopController.clickImage("imgs\\outlook\\btEnviar.png", 30)
    ;
61.                         correoEnviadoCorrectamente = true;
62.                     }
63.                 }
64.             }
65.         }
66.
67.         if (correoEnviadoCorrectamente) {
68.             functionalLog.information(mensajeLog);
69.         } else {
70.             functionalLog.error(mensajeLog);
71.         }
72.
73.         return correoEnviadoCorrectamente;
74.
75.     }
76.
77.     private boolean clickBotonNuevoCorreo(String referenciaTramite) {
78.         boolean clickCorrecto = true;
79.         if (DesktopController.existImage("imgs\\outlook\\btNuevoMensaje.png", 60, 10))
    {
80.             DesktopController.clickImage("imgs\\outlook\\btNuevoMensaje.png", 10);
81.             if(DesktopController.existImage("imgs\\outlook\\btNuevoMensaje.png", 60, 10
    )){
82.                 DesktopController.clickImage("imgs\\outlook\\btNuevoMensaje.png", 10);
83.             }
84.         } else {
85.             clickCorrecto = false;
86.             //mensajeLog = "No se ha podido hacer clic en el boton Enviar Correo para
    enviar el correo del trámite con referencia " + referenciaTramite;
87.             mensajeLog = "No se ha podido abrir la ventana Nuevo Correo para enviar el
    correo del trámite con referencia " + referenciaTramite;
88.         }
89.
90.         return clickCorrecto;
91.     }
92.
```

Per a comunicar-se amb la base de dades també s'ha desenvolupat uns serveis rest com els del panell de control que representen les taules mostrades en el disseny funcional i tècnic de l'annex.

10. Conclusions i treball futur

En primer lloc, ressalto una sèrie de conclusions a nivell personal que he tret al realitzar aquest treball de final de grau, totes positives.

Fer aquest treball m'ha ajudat a entendre més be com funciona a nivell professional empreses informàtiques i bancaries, ja que tot i estar en consultora informàtica tan gran com everis el projecte on he desenvolupat el treball esta totalment orientada al sector bancari. M'ha fet veure com d'important es la comunicació en un entorn professional i com cada vegada és més important la intel·ligència emocional enfront dels coneixements, ja que treballar i gestionar un equip exigeix molta intel·ligència emocional per ser capaç de treure el màxim rendiment dels treballadors i que aquests estiguin el màxim contents possibles i al final, puguin resoldre els dubtes que sorgeixin a becaris com jo.

A més, m'ha ajudat profundament en el meu desenvolupament professional tant a nivell de desenvolupar/programar, com a nivell d'analista, d'administrador de bases de dades i també fins a cert punt de gestor de projectes, doncs al haver organitzat tot el sistema d'informació nomes amb l'ajuda de la meva tutora m'ha donat molta experiència en aquest sentit.

Estic molt content de l'ús que s'està fent actualment amb el sistema que s'ha desenvolupat. Tot i que encara hi ha moltes coses que es poden millorar, és una base que ja estan utilitzant altres companys per a desenvolupar i gestionar automatitzacions. Automatitzacions que l'entitat financera ha validat i acceptat, i que estan molt a prop de passar a producció i executar-se diàriament. Que l'empresa hagi sigut capaç de dipositar confiança en el sistema desenvolupat i m'hagi permès crear-lo amb total llibertat es una experiència que agrairé sempre i de la qual estic molt orgullós.

Des del punt de vist professional, a nivell d'objectius, s'han assolit tots correctament. Que l'empresa estigui usant actualment el sistema desenvolupat és una prova suficient de que els objectius s´han assolit correctament. També tota la documentació i implementació (parcial, ja que el codi es confidencial de l'empresa) demostren l'assoliment dels objectius.

A nivell de les competències tècniques també s'han assolit totes correctament. S'ha desenvolupat un sistema d'informació, així com s'ha participat directament en el disseny, implementació i posada en marxa. A més, s'han avaluat diferents tecnologies per a escollir-ne la correcte i s'ha creat i administrat bases de dades en profunditat.

Com a treball futur, existeixen millores que es podrien implementar per optimitzar el sistema ja existent, com per exemple, afegir noves funcionalitats al framework utilitzades habitualment en l'entorn de l'entitat bancaria, actualitzar el panell de control afegint funcionalitats per a la generació de reports complexos, entre altes.

El que sembla clar es que el sistema ha arribat per quedar-se, i actualment el departament informàtic d'everis BPO esta iniciant un nou projecte centrat en les automatitzacions i basat en aquest sistema d'informació i per tant, en el futur arribaran noves millors a nivell de funcionalitats i de usabilitats, i qui sap si en un futur serà utilitzat també en altres departaments de BPO.

11. Bibliografia

[R1] Es.wikipedia.org. (2017). Framework. [online] Available at: https://es.wikipedia.org/wiki/Framework [Accessed 20 Mar. 2017].

[R2] Vanguardia, L. (2017). La automatización de procesos bancarios aumentará la productividad un 50 %. [online] La Vanguardia. Available at: http://www.lavanguardia.com/vida/20160309/40317518306/la-automatizacion-de-procesos-bancarios-aumentara-la-productividad-un-50.html [Accessed 3 Feb. 2017].

[R3] Schwab, K. and Botín, A. (2016). La Cuarta revolución industrial. Barcelona: Debate.

[R4] Blue Prism. (2017). Blue Prism. [online] Available at: https://www.blueprism.com/ [Accessed 6 Mar. 2017].

[R5] Opencv.org. (2017). OpenCV library. [online] Available at: http://opencv.org/ [Accessed 7 Mar. 2017].

[R6] Emgu.com. (2017). Emgu CV: OpenCV in .NET (C#, VB, C++ and more). [online] Available at: http://www.emgu.com/wiki/index.php/Main_Page [Accessed 7 Mar. 2017].

[R7] Sikulix.com. (2017). RaiMan's SikuliX. [online] Available at: http://www.sikulix.com/ [Accessed 4 Mar. 2017].

[R8] Oracle.com. (2017). Software Java | Oracle España. [online] Available at: https://www.oracle.com/es/java/index.html [Accessed 11 Mar. 2017].

[R9] Ca.wikipedia.org. (2017). Cascading Style Sheets. [online] Available at: https://ca.wikipedia.org/wiki/Cascading_Style_Sheets [Accessed 20 Jun. 2017].

[R10] Es.wikipedia.org. (2017). AngularJS. [online] Available at: https://es.wikipedia.org/wiki/AngularJS [Accessed 12 Mar. 2017].

[R11] Ca.wikipedia.org. (2017). REST. [online] Available at: https://ca.wikipedia.org/wiki/REST [Accessed 6 Mar. 2017].

[R12] Tarifaluzhora.es. (2017). Tarifaluzhora. [online] Available at: http://tarifaluzhora.es/ [Accessed 11 Mar. 2017].

[R13] Unirest.io. (2017). Unirest for Java - Simplified, lightweight HTTP Request Library. [online] Available at: http://unirest.io/java.html [Accessed 20 Jun. 2017].

[R14] seleniumhq.org (2017). Selenium for automations. [online] Available at: http://www.seleniumhq.org [Accessed 11 March. 2017].

[R15] Ca.wikipedia.org. (2017). NetBeans. [online] Available at: https://ca.wikipedia.org/wiki/NetBeans [Accessed 20 Jun. 2017].

[R16] Alvarez, M. (2017). Qué es MVC. [online] DesarrolloWeb.com. Available at: https://desarrolloweb.com/articulos/que-es-mvc.html [Accessed 20 Jun. 2017].

[R16] Es.wikipedia.org. (2017). GlassFish. [online] Available at: https://es.wikipedia.org/wiki/GlassFish [Accessed 20 Jun. 2017].

[R17] Volere Plantilla de Especificación de Requisitos -James & Suzanne Robertson, Edición 11 Febrero 2006

12. Annex – Disseny funcional/tècnic de l'automatització

A continuació es mostra el disseny funcional i tècnic realitzat i validat per el banc. S'han omès totes les dades que es poden considerar confidencials.

1. Control de cambios

Versión	Fecha	Autor	Responsable	Descripción
1.0	27/03/2017		OT everis	Primera versión Diseño funcional

2. Aprobación

	OT everis	OT BSOS	Comité Transformación	Observaciones
Nombre				
Fecha	27/03/2017			

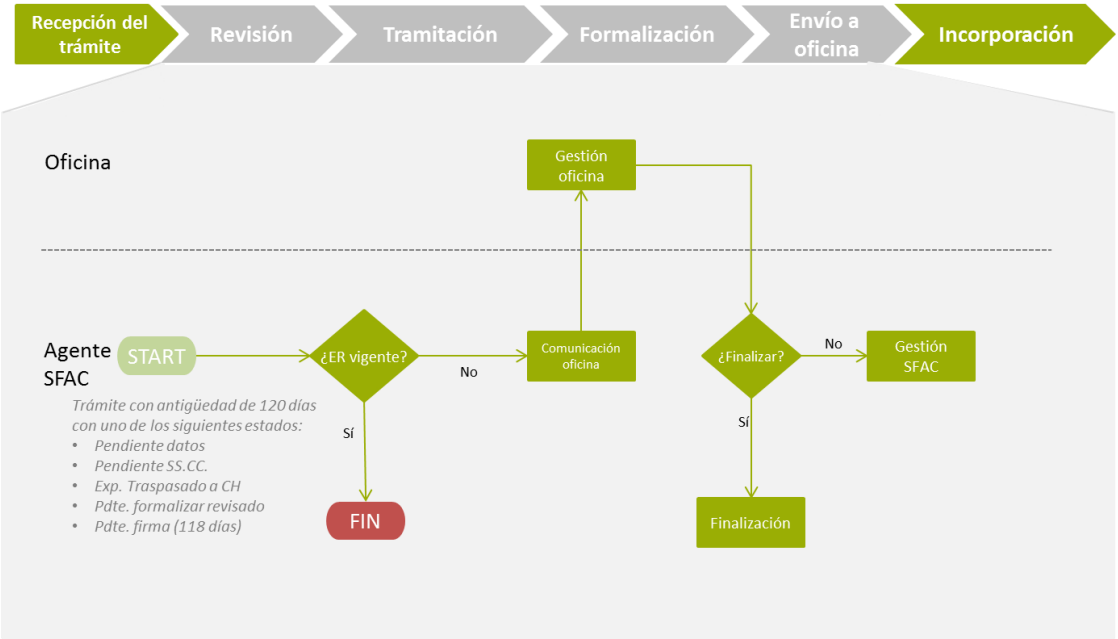
3. Propósito, alcance y objetivos

3.1. Situación actual

Actualmente, en el servicio de formalización de activo (SFAC), existe la necesidad de finalizar los trámites cuyo expediente de riesgo (en adelante, ER) haya expirado por caducidad. Este hito se produce a los 120 días de su aprobación.

Una vez se ha caducado el ER, deberá procederse a la cancelación del expediente de formalización y posteriormente, a la finalización del trámite.

Esta práctica se está llevando a cabo con controles diarios desde la Oficina de Calidad del proyecto, enviando avisos a diario de aquellos trámites con necesidad de finalización a los agentes asignados a dichos trámites y enviando avisos a los agentes para que procediesen a la finalización de aquellos trámites caducados previa verificación con oficina, realizando la tarea totalmente manual, además de digitalizar el correo de oficina subiéndolo al gestor documental.



Flujograma As-Is

3.2. Solución propuesta

A raíz de esta necesidad, se desarrolla la automatización de este proceso de finalización, analizando los trámites que por fecha de envío están caducado y tratándolos de forma distinta dependiendo del estado en que se encuentren (se finalizarán los que se hallen *pendiente de datos, pendiente SS.CC., pendiente formalizar revisado, pendiente de firma o expediente traspasado a centro hipotecario*).

Además, habrá que distinguir la persona asignada al trámite para no gestionar aquellas operaciones tratadas por agentes del CAR.

Tratamiento según estado

A continuación se detalla el tratamiento para cada tipo de estado:

- **Pendiente de firma**: para los trámites que se encuentren en este estado, se emitirá una notificación a oficina y, se cerrará el expediente de formalización y el trámite, una vez se haya obtenido la respuesta o al tercer día de la comunicación sin haber recibido contestación. Esta comunicación se ejecutará el día 118 (pudiendo ser modificable) des del envío a formalizar el trámite por parte de oficina.

  En el momento de entablar comunicación con oficina deberá indicarse la siguiente frase en Observaciones CAR (campo que se halla en el detalle del trámite): *"Enviado correo a oficina para la confirmación de si la operación se ha firmado. En caso de no recibir respuesta al correo, el trámite se finalizará en un plazo de dos días"*.

  En el caso de haber obtenido respuesta de oficina, el agente deberá cambiar el estado del trámite a *operación firmada*, estado en el cual el robot ya no lo leerá ni lo listará para su finalización en el próximo diagnóstico que realice. En el caso de que oficina confirme que

se deba cerrar, será finalizado por el robot al día D+XXX de la comunicación realizada, siendo D la fecha en la que se emitió la notificación a oficina.

- **Para el resto de estados (pendiente de datos, pendiente SS.CC., pendiente formalizar revisado y expediente traspasado a CH):** se llevará a cabo el mismo proceso pero sin realizar la notificación a oficina, procediendo directamente a la finalización cuando se detecte que ese trámite sobrepasa los 120 días desde que fue enviado para su formalización.

Estas gestiones comentadas, solamente se realizarán durante determinadas partes del proceso, que son durante la revisión, la tramitación, la formalización o bien, una vez se ha enviado a oficina la documentación.

La solución que se plantea consiste en tres partes:

1. El robot se encarga de realizar dos consultas con periodicidad diaria (configurable por el administrador/coordinador) y así obtener un listado de todos los trámites que estén sujetos a la necesidad de enviar notificación y a un segundo listado que será de los trámites identificados para finalizarlos.
2. Si se trata de un trámite con estado pendiente de firma, se entabla comunicación con oficina para notificar su finalización, si es que oficina no confirma que esa operación ha sido firmada.
3. En función de las respuestas recibidas de oficina se procederá de la siguiente manera:
    - En el caso de no obtener respuesta por parte de oficina, se finalizará el trámite al tercer día de la emisión de la comunicación.
    - En caso de que oficina si responda habrá dos opciones, confirmación de finalización o necesidad de mantenerlo abierto el trámite. En este último caso, el agente deberá modificar el estado del trámite a *Operación firmada* con el fin de que la automatización no la liste para la finalización el próximo día que extraiga un nuevo listado para finalizar.

    En el caso de que se tenga que finalizar, no se realizará ninguna gestión y al tercer día será cancelado por la automatización (ejecución similar a la de finalización por no respuesta).

3.3. Consideraciones

Como objetivo de esta solución se pretende conseguir liberar de dicha tarea a los agentes y así poder mantener un stock limpio de trámites actualizado al día sin posponer esta actividad como se realiza a día de hoy.

Esta automatización abarca todos los productos del servicio de formalización, liberando así al agente de tener que realizar esta gestión completamente manual.

La automatización deberá considerar que se tendrán que finalizar trámites del centro de trabajo de Barcelona y Alicante y, por ello, deberá ser parametrizable para que pueda atender estas necesidades y las diferencias de procedimiento que puedan existir en ambos centros.

De la misma forma, otros aspectos como el texto a comunicar, los días de espera antes de finalizar el trámite o bien, el número de comunicaciones a efectuar con oficina, también tendrán que ser parametrizables para cubrir posibles cambios operativos a posteriori.

Comunicación con oficina

A continuación se muestra la plantilla para realizar la comunicación que existe con oficina en los casos que el trámite se encuentre pendiente de firma:

Buenos días,

Del trámite de referencia, salvo error, no hemos recibido la conformidad de firma. Necesitamos que nos informéis urgentemente de la situación del mismo. Si se firmó, necesitaremos urgentemente que nos lo confirméis con el fin de poder incorporarlo.

En caso de no recibir respuesta en dos días laborables, daremos el trámite por finalizado.

Gracias y saludos.

La comunicación con oficina deberá llevarse a cabo desde el buzón genérico del servicio de SFAC, es decir, desde el (correo confidencial) de tratarse de trámites del centro de Barcelona o bien, (correo confidencial) si se trata de finalizar un trámite del centro de Alicante. Es necesario tener en consideración que deberá realizarse la notificación con la funcionalidad *reply to* para que la contestación de oficina (en caso de que hubiera), fuera enviada directamente al buzón personal del agente.

Para el resto de estados (pendiente de datos, pendiente de SS.CC., pendiente formalizar revisado y Exp. traspasado a CH) no se establecerá comunicación previa con oficina para finalizar el trámite siguiendo la operativa actual.

Para la finalización del trámite, se informará en el campo *Observaciones CAR* el siguiente mensaje:

Trámite finalizado por antigüedad superior a 4 meses. Para volver a tramitar la operación será necesario un nuevo ER si éste ha caducado, y la iniciación de un nuevo trámite una vez haya sido aprobado.
En el caso de operaciones sin ER, deberán enviar nuevo trámite. Tengan presente que si se envió documentación, esta no puede ser firmada.

3.4. Explotación de datos

La mecanización del proceso de finalización de trámites SFAC no comportará la necesidad de requisitos adicionales para los archivos de reporting que recibe everis diariamente, siendo suficiente la información que se recibe para su explotación y obtención de indicadores e información.

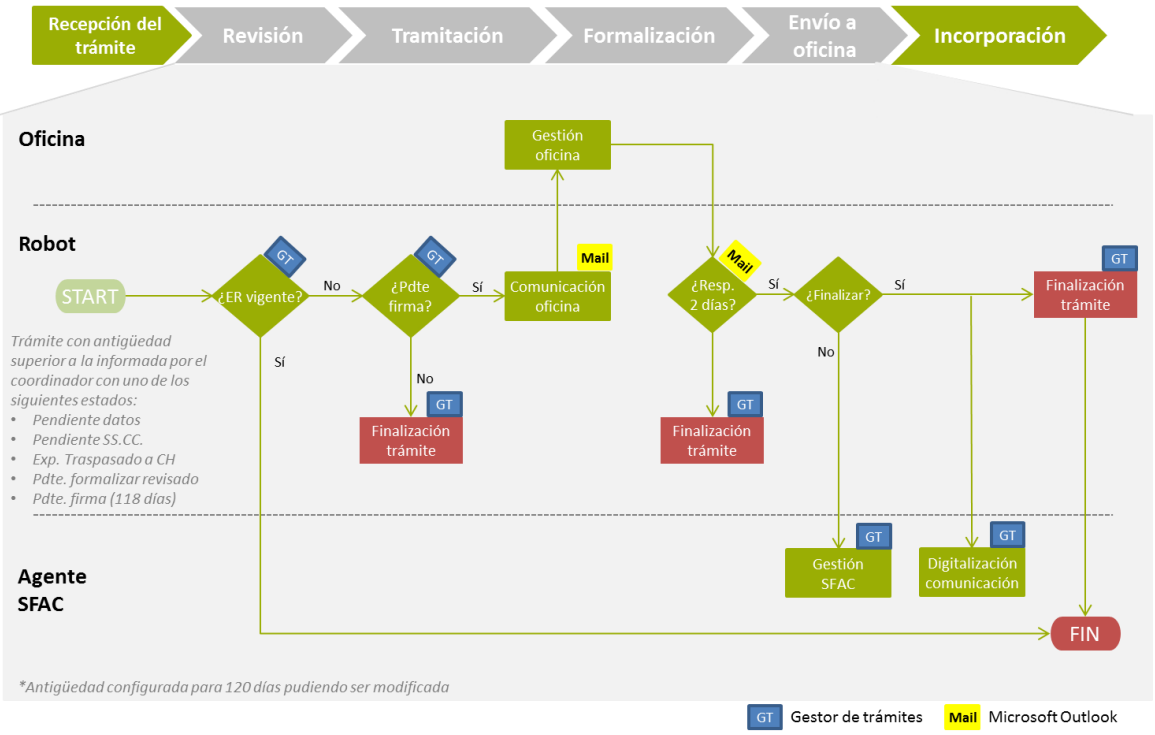
El robot deberá almacenar toda la información necesaria para poder realizar un reporting diario de su actividad y los resultados obtenidos. Asimismo, es necesario almacenar datos de las referencias de los trámites gestionados y el agente asignado a dicho trámite, tanto para aquellos finalizador por el robot como para las notificaciones enviadas a oficina.

El reporting que se pueda extraer, debe ser capaz de presentarse en formato Excel.

4. Solución técnica

4.1. Flujograma de la solución propuesta

A continuación se adjunta un flujograma donde se puede valorar como quedará el proceso una vez se implemente la solución propuesta.



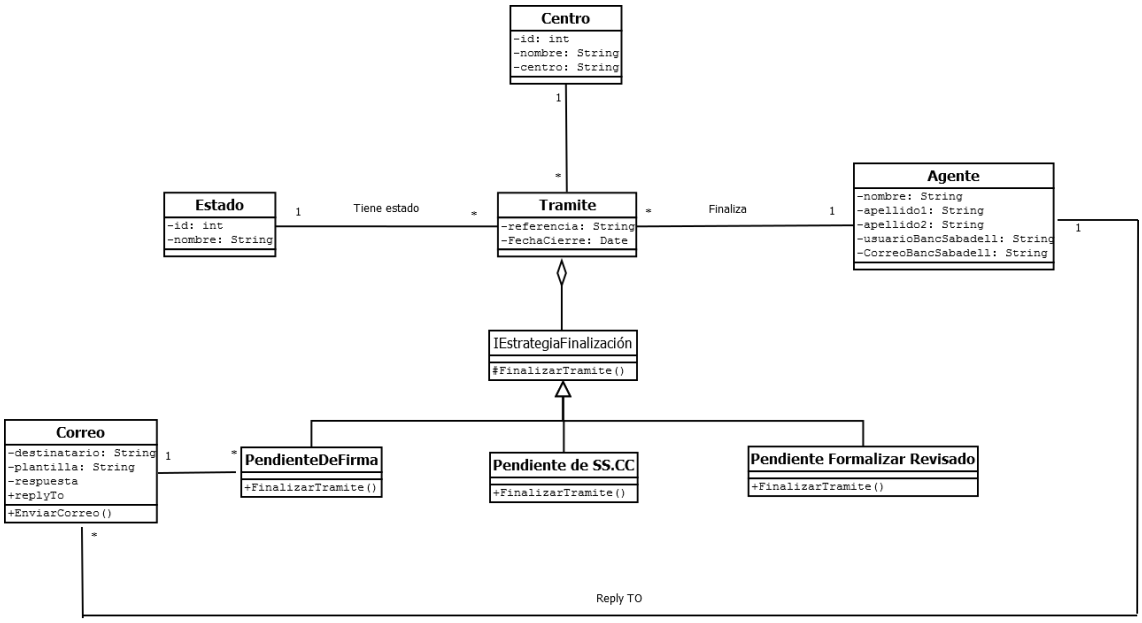
Flujograma To-Be

4.2. Aspectos tecnológicos

La tecnología que se usará consistirá en un controlador de escritorio para interactuar con SIBIS y estará desarrollado en el lenguaje de programación de alto nivel Java 8 y el framework de automatizaciones Saturnus. También, se usará SQL Server como gestor de bases de datos para guardar la información que sea necesaria.

Los parámetros de configuración del robot se introducirán por consola cada vez que se quiera ejecutar.

La arquitectura de clases del robot será la siguiente:



A continuación describiremos las clases usadas:

- **Estado:** Representa el estado del trámite.
- **Centro:** Representa los centros desde donde se finalizan los trámites.
- **Agente:** Representa el agente al que está asignado(Finaliza) el trámite.
- **Tramite:** Representa el trámite a finalizar y puede ser de distintos tipos.
- **IEstrategiaFinalizacion**: Interfaz que define los métodos necesarios para cada estrategia disponible.
- **Pendiente de Firma:** Estrategia para finalizar trámites de tipo Pendiente de Firma.
- **Pendiente de SS.CC:** Estrategia para finalizar trámites de tipo SS.CC.
- **Pendiente Formalizar Revisado:** Estrategia para finalizar trámites de tipo Formalizar Revisado.
- **Correo:** Clase que representa los correos a enviar para los trámites pendiente de firma.

Requisitos funcionales

Número requisito:	1	**Tipo de requisito:**	Funcional
Descripción:	El sistema debe gestionar el correo siendo capaz de redactar, seleccionar el destinatario, configurar la opción reply-to y enviar.		
Justificación:	Para poder finalizar un trámite del estado *Pendiente de Firma* se tendrá que enviar un correo a oficina para que se confirme la finalización del trámite.		

Número requisito:	2	**Tipo de requisito:**	Funcional
Descripción:	El sistema debe navegar correctamente a través de la herramienta proporcionada por el Banco llamada SoftBan y concretamente en todas las herramientas de la sección Sibis.		
Justificación:	La finalización de los trámites se realizan siempre a través de SoftBan, por lo que el sistema deberá interactuar correctamente con esta aplicación.		

Número requisito:	3	**Tipo de requisito:**	Funcional
Descripción:	El sistema debe finalizar los trámites con más de 120 días de antigüedad.		
Justificación:	Para realizar el objetivo del robot deberá finalizar los trámites con más de 120 días de antigüedad.		

Número requisito:	4	**Tipo de requisito:**	Funcional
Descripción:	El sistema debe escoger la plantilla adecuada dependiendo del tipo de trámite en él envió de correos.		
Justificación:	Para poder comunicar-se con oficina existen dos plantillas y el robot deberá escoger la adecuada.		

Número requisito:	5	**Tipo de requisito:**	Funcional
Descripción:	El sistema debe guardar la información de todos los trámites tratados, los correos enviados y las respuestas obtenidas.		
Justificación:	Para poder realizar una explotación de datos y generar reportes debemos guardar todos los datos mencionados previamente.		

Requisitos no-funcionales

Número requisito:	6	**Tipo de requisito:**	Seguridad
Descripción:	El sistema debe cumplir las disposiciones recogidas en la Ley Orgánica de Datos Personales y en el Reglamento de medidas de seguridad.		
Justificación:	Al tratar con datos confidenciales y personales de los clientes es necesario cumplir las disposiciones recogidas en la LOPD.		

Número requisito:	7	**Tipo de requisito:**	Velocidad y latencia
Descripción:	El sistema debe finalizar un trámite en un tiempo aproximado de 5 minutos.		
Justificación:	Para poder cumplir el número de trámites resueltos pactado, el sistema deberá resolverlos en el tiempo indicado.		

Número requisito:	8	**Tipo de requisito:**	Confiabilidad
Descripción:	El sistema estará disponible para usarlo 24 horas al día, 365 días por año.		
Justificación:	Es critico que el sistema esté disponible siempre para poder cumplir con el SLA establecido.		

Número requisito:	9	**Tipo de requisito:**	Sistemas adyacentes
Descripción:	El sistema trabajara con la última versión de la herramienta proporcionada por el Banco denominada SoftBan.		
Justificación:	Para que funcione correctamente la automatización se deberá trabajar con imágenes de la versión instalada en los ordenadores de los agentes, que es la última versión de SoftBan.		

Número requisito:	10	**Tipo de requisito:**	Producción
Descripción:	El producto podrá ser instalado por un usuario sin entrenamiento sin recurrir a instrucciones impresas por separado.		
Justificación:	El producto se ha de poder instalar de manera fácil en los ordenadores u ordenador en que se vaya a usar.		

Número requisito:	11	**Tipo de requisito:**	Producción
Descripción:	El producto será distribuido en un ejecutable.		
Justificación:	Se distribuirá en un ejecutable por su fácil instalación.		

Número requisito:	12	Tipo de requisito:	Adaptabilidad
Descripción:	Se espera que el producto se ejecute bajo Windows 7.		
Justificación:	El sistema operativo oficial de everis es Windows 7.		

4.3. Gestión de la documentación asociada a la operativa

Para poder finalizar los trámites necesarios se necesitara enviar un correo a la oficina para que confirmen si quieren o no que se resuelvan los trámites en cuestión.

- **Gestor de Microsoft Outlook**

Para poder satisfacer la primera parte de la solución propuesta, la aplicación tendrá que acceder al gestor de correos Outlook y gestionar los envíos de correos a oficina con la información que sea necesaria.
Como herramienta de gestión, se utilizarán plantillas de envío predeterminadas a oficina en relación al estado en el que se encuentre el trámite.

4.4. Necesidades de los sistemas del banco

El robot deberá tener un usuario y una contraseña para poder acceder al terminal financiero del banco.

4.5. Servicio de soporte técnico

Cualquier incidencia podrá ser reportada al departamento de Tecnología de la Oficina de Transformación del servicio BPO de everis.

5. Modelo de datos

A continuación, se adjuntan las tablas necesarias para poder abordar la solución técnica:

Trámites	
Id	Int: autoincremental (PK)
Referencia	Int: Clave foránea a la tabla Códigos
Estado Trámite	Int: Clave foránea tabla Estado Trámite
Agente	Text: Usuario BS del agente.
Centro	Int: Clave foránea de la tabla Centros
agenteOficinaBanco	Text: Usuario agente oficina banco
fechaExtraccion	Date: Fecha de extracción del trámite del gestor de trámites
Fecha de Cierre	Date: Fecha de cierre del trámite
fechaEnvioCorreo	Date: Fecha en que se envio el correo al agente Oficina del banco con Reply To al agente usuario del banco.
fechaCambioEstado	Date: Fecha en que se reviso el trámite y este no estaba en su estado original y por lo tanto había sido modificado.

Tabla 1: Trámites

Éstos son los conceptos que utilizará el aplicativo para definir la búsqueda de trámites que deberán ser comunicados y consultados a oficina para ser finalizados.

Estados Trámites	
Id	Int: autoincremental (PK)
Estado	String: Estado del trámite(Pendiente firma, Pendiente datos , Pendiente SS.CC. y Pendiente formalizar revisado).
diasComunicacionConOficina	Int: El número de días anteriores al dia actual en que los trámites de cada estado se deberán comunicar con oficina
diasFinalización:	Int: El número de días anteriores al actual en que el trámite se deberá finalizar.
diasEsperaRespuesta:	Int: Número de días que cada trámite del estado deberá esperar para poder ser finalizado si ha habido comunicación con oficina.

Tabla 2: Estados trámites

La tabla de *Estados Trámites* contiene la información de los diferentes estados de trámite que tratará el robot para realizar las gestiones oportunas.

Centros	
Id	Int: autoincremental (PK)
Centro	Int: número del centro de everis
Nombre	String: Nombre de la ciudad del centro

Tabla 3: Centros

La tabla referente a *Centros* indica a que centro resolutor pertenece el trámite a finalizar, pudiendo ser Barcelona o Alicante.

Logs	
Id	Int: autoincremental (PK)
ConfigurationID	Int: Id de la configuración del robot
Message	Text: Mensaje que informara del robot.
Level	String: Nivel del log (Error, Information, etc)
TimeStamp	Date: Fecha en que se ha introducido el log.
MachineName	Text: Máquina donde se ejecuta el robot.

Tabla 4 : Logs

En la tabla *Logs* se registrará toda la actividad que realizará el robot, desde abrir SIBIS, hasta la apertura del trámite y la finalización. Toda la información de las tareas que realiza el robot se guardarán en ésta tabla.

6. Procedimientos

6.1. Cambio en los procedimientos

A diferencia del procedimiento actual, el robot provocará cambios en la manera de ejecutar esta tarea y en la intervención del agente en dicha parte.

En primer lugar, el agente de SFAC ya no deberá llevar un control sobre la vigencia de los trámites que tiene asignado, liberándose así de una pequeña carga de trabajo.

En cuanto al proceso de finalización, también quedará liberado de efectuar la comunicación con oficina (en el caso necesario) y la ejecución de finalización del trámite, cancelando expediente y caso.

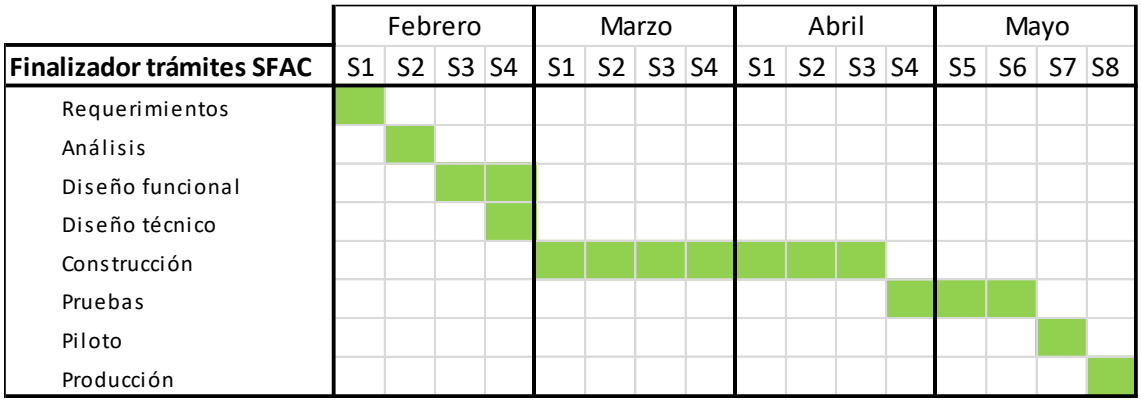
La única tarea que tendrá que llevar a cabo será la interpretación del correo en caso de que se haya enviado notificación a oficina para comprobar que si esa operación ha llegado a firmarse ante notario. En caso de que así sea, el agente deberá modificar el estado del trámite (*Operación firmada)* evitando así, el cierre a D+XXX de la notificación.

Con el nuevo procedimiento, el agente no digitalizará la comunicación en ninguno de los casos.

De esta forma, se realiza un cambio sustancial al pasar de realizar todo el proceso por el agente de SFAC a sólo tratar el correo recibido de contestación de oficina y modificar el estado del trámite si es necesario.

7. Calendario

En las siguientes imágenes, se tiene en cuenta las fases por las que pasará esta iniciativa para poder completarla.



La construcción podrá sufrir cambios en función del diseño técnico que se lleve a cabo de la solución.

8. Eficiencias y costes

8.1. Ventajas

La adopción de esta solución para la finalización de los trámites, aporta las siguientes ventajas:

- Liberación de carga de trabajo a los agentes de formalización.
- Actualización diaria del stock y limpieza de aquellos trámites.
- Imagen fiel de los trámites que requieren tramitación.
- Aumento de la productividad.
- Disminución de posibles errores humanos.

9. Implementación del robot

A continuación se describen las cinco partes en que se ha dividido el robot y como se ejecuta cada una de ellas:

1. **Extracción de trámites**

   El robot abre el gestor de trámites y por cada estado de trámite existente en la base de datos (actualmente: Pendiente firma, Pendiente Datos, Pendiente SS, Pendiente Formalizar Revisado) extrae los trámites existentes y los inserta en la base de datos. En concreto los datos que se guardan son: la referencia del trámite, el estado, el agente everis asignado, el centro y la fecha de extracción del trámite.

   Antes de realizar cada inserción se comprueba que tal trámite no exista ya en la base de datos, para asegurarnos que no extraemos dos veces el mismo trámite.

2. **Extracción del agente oficina banco**

   En el segundo paso, el robot ejecuta una consulta en la base de datos que le devuelve todos los trámites que se han extraído en el día actual.
   Busca cada trámite a través del campo Referencia del Gestor de trámites, obtiene el agente de oficina del banco asignado al trámite y actualiza la base de datos con esta nueva información.

   En este momento, una fila de la Tabla Trámites estaría rellenada de la siguiente manera:
   - Referencia: Referencia del trámite, extraída en el primer paso.
   - Estado: El estado correspondiente al trámite.
   - Agente: Agente asignado al trámite, extraído en el primer paso.
   - Centro: Centro del trámite.
   - AgenteOficinaBanco: Null (Vacío)
   - fechaExtracción: Fecha del día actual, día en que se extrajo el trámite.
   - FechaCierre: Null (Vacío).
   - FechaEnvioCorreo: Null (Vacío).
   - FechaCambioEstado: Null (Vacío).

3. **Envió de correos a oficina del banco**

   En este paso el robot realiza una nueva consulta a la base de datos que le devuelve los trámites que tengan un estado que se tenga comunicar con oficina, no esté cerrado y no se haya enviado previamente un correo.
   Para realizar este proceso, la consulta comprueba que el estado de cada trámite tenga los días de respuesta de la tabla Estados mayores que cero, (al ser mayor que cero indica que hay comunicación con oficina, si es cero indica que no hay comunicación con oficina) , que la fecha de extracción del trámite sea la del día actual , que no exista fecha de cierre del trámite (fechaCierre sea Null) y que no exista fecha de envio de correo (fechaEnvioCorreo sea Null).

   Con estas cuatro condiciones nos aseguramos de que los trámites para los que se enviara correo sean correctos.
   Una vez ejecutada la consulta, el robot abre el Outlook de SoftBan e inicia él envió de correos.
   Para cada correo especifica:
   - Para: Usuario Oficina Banco..
   - Asunto: Finalizar trámite + Referencia Tramite + Nombre completo del agente everis.
   - Cuerpo:

Cuerpo: *Buenos días,*

Del trámite de referencia, salvo error, no hemos recibido la conformidad de firma. Necesitamos que nos informéis urgentemente de la situación del mismo. Si se firmó, necesitaremos urgentemente que nos lo confirméis con el fin de poder incorporarlo.

En caso de no recibir respuesta en dos días laborables, daremos el trámite por finalizado.

Gracias y saludos.

   - ReplyTo: Correo banco agente everis.

   Por cada correo enviado se actualiza la tabla Trámites insertando la fecha en que se envió el correo (FechaEnvioCorreo).

4. Actualización de las observaciones de los trámites que se ha enviado correo

Una vez enviados los correos el robot va accediendo a los trámites que se han comunicado con oficina a través del campo Referencia del gestor de trámites y actualiza las observaciones con el siguiente texto:

Enviado correo a oficina para la confirmación de si la operación se ha firmado. En caso de no recibir respuesta al correo, el trámite se finalizará en un plazo de dos días.

5. **Finalización de los trámites**

En el último paso, el robot realiza dos consultas en la base de datos para obtener los tramites que se comunican con oficina y se tienen que finalizar y los trámites que no se comunican con oficina y se tienen que finalizar.

Para los que comunican con oficina se tienen que cumplir las siguientes condiciones:

- FechaCierre vacía (null).
- FechaEnvioCorreo es igual a la fecha actual menos los días de espera de la comunicación con oficina del estado del trámite.
- FechaCambioEstado esta vacía (null).

Para los tramites que no se comunican con oficina se tienen que cumplir las siguientes condiciones:

- FechaCierre vacía(null).
- FechaEnvioCorreo vacía (null).
- FechaCambioEstado vacía (null).
- FechaExtracción es igual al día actual.

Una vez obtenida la lista de trámites para finalizar se accede a cada uno de ellos a través del campo referencias del gestor de trámites y se obtiene su situación actual. Si la situación actual del trámite (estado) existe en la lista de estados existentes en la tabla Estados de la base de datos significa que el trámite no se ha modificado y por lo tanto se puede finalizar. A continuación se cancela el expediente en caso de que sea necesario y se actualiza la base de datos especificando la fecha de cierre.

En caso que la situación actual del trámite no exista en la tabla Estados de la base de datos, significa que oficina ha modificado ese trámite y por lo tanto no se debe finalizar. En ese caso no se cierra el trámite y se actualiza la fecha de cambio de estado de la base de datos con la fecha actual.

Por último, todos los pasos del robot se realizan por cada centro especificado en la tabla Centros de la base de datos.