Universidad Politécnica
de Madrid

Escuela Técnica Superior de Ingenieros Informáticos

Grado en Ingeniería Informática

Trabajo Fin de Grado

APLICACIONES de RPA en el ÁMBITO EMPRESARIAL

Autor: Laura María Gómez González

Tutora: María Mercedes Pérez Castellanos

Madrid, enero 2020

Este Trabajo Fin de Grado se ha depositado en la ETSI Informáticos de la Universidad Politécnica de Madrid para su defensa.

Trabajo Fin de Grado
Grado en Ingeniería Informática
Título: APLICACIONES de RPA en el ÁMBITO EMPRESARIAL
Enero 2020

Autor:
Laura María Gómez González

Tutor:
María Mercedes Pérez Castellanos
Departamento de Arquitectura y Tecnología de Sistemas Informáticos
ETSI Informáticos
Universidad Politécnica de Madrid

Resumen

Robotic Process Automation (RPA) es una tecnología emergente que sirve para desarrollar soluciones de software. Mediante la automatización con RPA, se crean y ejecutan robots de software que interactúan con aplicaciones, interfaces de usuario, ficheros y APIs de sistemas para llevar a cabo actividades repetitivas, de gran volumen y basadas en reglas lógicas, imitando la forma en que las realiza un ser humano sin modificar los sistemas sobre los que actúan y eliminado tiempo de operación y recursos del sistema. RPA aplica técnicas algorítmicas para tratar datos estructurados, utiliza la lógica para analizar y tomar decisiones sencillas y la Inteligencia Artificial para el tratamiento de datos no estructurados y el procesamiento del lenguaje natural.

El objetivo de este trabajo es el análisis de la aplicación de Robotic Process Automation (RPA) en el ámbito empresarial. Se explicará qué conlleva la automatización de tareas mediante la creación de robots de software, cuáles son las causas que han llevado a su uso, los beneficios y también las posibles consecuencias. Para entenderse mejor, se escogerá un ejemplo de organización en el que se configurarán robots de software con el programa UiPath que automatizarán flujos de procesos ejemplificados, simulando casos de uso y así demostrar el uso de RPA para la transformación digital en una empresa.

Abstract

Robotic Process Automation (RPA) is an emerging technology that is mainly used to develop software solutions. Through automation with RPA, software robots are created and executed to interact with applications, user interfaces, files and system APIs to carry out repetitive, high-volume activities based on logical rules, imitating the way in which human perform that activities without modifying the systems on which they act and eliminating operating time and system resources. RPA applies algorithmic techniques to treat structured data, uses logic to analyze and make simple decisions and Artificial Intelligence for the treatment of unstructured data and natural language processing.
The objective of this work is the analysis of the application of Robotic Process Automation (RPA) in the business field. It will be explained what the automation of tasks entails by creating software robots, what are the causes that have led to their use, the benefits and also the possible consequences. For a better comprehension, an example of an organization will be chosen in which software robots will be configured with the UiPath program that will automate exemplified process flows, simulating use cases and with thus demonstrate the use of RPA for digital transformation in a company.

Tabla de contenidos

Imágenes

Capítulo 1

Introducción y objetivos

Este trabajo se centra en el análisis de la aplicación de Robotic Process Automation (RPA) en el ámbito empresarial.
En la actualidad, las empresas se encuentran atravesando lo que se conoce como transformación digital o la revolución de la Industria 4.0. Lo que se pretende llevar a cabo es la automatización y eficiencia en los procesos de negocio, utilizando tecnologías más eficientes con la finalidad de reducir tiempos, errores humanos y eliminar mano de obra, permitiendo así que se pueda dedicar este tiempo a actividades más productivas, estratégicas y que aporten valor en una organización.
RPA, es una metodología que sirve para automatizar procesos, permitiendo crear robots de software que ejecuten tareas basadas en reglas lógicas, repetitivas y de grandes volúmenes, sustituyendo la actividad de un operador de negocio.
Es muy fácil confundirse y pensar que RPA se utiliza para crear robots físicos, pero esto dista mucho de la realidad, ya que con RPA lo que se desarrolla son robots de software que se adaptan a los procesos existentes en una organización, se programan mediante un lenguaje de programación adaptado a la aplicación que se use de RPA o se graban las acciones del usuario para luego ser ejecutadas en la interfaz gráfica de usuario.
A pesar de ser una tecnología emergente, muchas empresas ya están beneficiándose de las ventajas de RPA y dando el salto hacia la transformación digital, al igual que otras están interesadas y evaluando la posibilidad de aplicar RPA en sus procesos de negocio. No obstante, para incorporar RPA en una empresa, se deben analizar previamente qué procesos de negocio quieren ser automatizados y verificar que realmente puedan serlo, además de adaptar la estructura tecnológica y humana tras la incorporación de la automatización.
En este trabajo se explicará qué conlleva la incorporación de la tecnología RPA en el ámbito empresarial, las causas que han motivado la introducción de RPA en el modelo de negocio y también los beneficios generados y las consecuencias. Para entenderse mejor, se escogerá un ejemplo de organización en el que se configurarán robots de software con el programa UiPath que automatizarán ejemplos de procesos de negocio en una organización, simulando casos de uso y así demostrar el uso de RPA para la transformación digital en una empresa.

Los objetivos que se quieren alcanzar con la elaboración de este trabajo son:

- El estudio de la aplicación de RPA en una organización y las soluciones que aporta.
- El desarrollo de robots de software con UiPath y el estudio del comportamiento y capacidades de trabajo de estos robots de software.

Este trabajo está dividido en dos partes, el marco teórico en el que se alcanza el primer objetivo y la parte práctica de desarrollo en la que se logra el segundo objetivo.
En el marco teórico se estudia en profundidad lo que es la tecnología Robotic Process Automation. Se explica qué es, cuáles son sus funcionalidades y usos, los tipos de automatización existentes, los beneficios y consecuencias que aporta la integración de RPA en una organización y las diferencias con la automatización clásica. A parte de eso, también se centra en los softwares RPA y se realiza una comparativa entre aquellos adecuados para ser luego utilizados en el desarrollo de los robots de software. También cabe mencionar que se menciona la relación de RPA con la Inteligencia Artificial y las líneas futuras de RPA.
La parte práctica del trabajo consiste en la elaboración de dos robots de software con UiPath, el programa elegido. En este apartado, no sólo se centra en desarrollo software, sino que previamente se establece el diseño y arquitectura de los robots y los planes de pruebas. Se elige un tipo de organización y se buscan dos procesos de negocio para construir dos robots que los automaticen. En la parte de diseño del sistema, se explican las funcionalidades de UiPath y cómo es el modelo de trabajo con este programa, se establecen las pautas a seguir para diseñar y estructurar el desarrollo los robots, se explica detalladamente cómo se han programado los dos robots y los planes de pruebas elaborados para detectar fallos del sistema y probar que los robots funcionen correctamente.

Capítulo 2

Análisis del estado del arte

2.1 ¿Qué es RPA?

Robotic Process Automation (RPA) o Automatización Robótica de Procesos es una tecnología que permite desarrollar robots de software para emular e integrar las acciones de un humano que interactúa dentro de la interfaz de usuario de un sistema informático para ejecutar procesos de negocio. [1]
Los robots de software RPA utilizan la interfaz de usuario del sistema informático para capturar datos y manipular aplicaciones, de la misma forma que lo hacen las personas. Los robots de software interpretan lo que visualizan en las pantallas del sistema informático y transmiten las órdenes requeridas simulando el uso del teclado y ratón. Interpretan y elaboran respuestas y se comunican con otros sistemas para llevar a cabo una gran variedad de tareas repetitivas. Su función es reducir o directamente eliminar la carga de trabajo de las personas que trabajan en una organización realizando tareas repetitivas y voluminosas principalmente, además de dar el salto hacia la mejora de los niveles de calidad y la estrategia de transformación digital de una organización.
La implantación de RPA en una organización es rápida al igual que los resultados. Sin embargo, no detecta fallos. Las excepciones que puedan producirse durante la ejecución de un robot deben ser previamente controladas por un experto.
RPA engloba tres términos. En primer lugar, el propio software de RPA, la tecnología cognitiva y la Inteligencia Artificial.

2.2 Relación de RPA con la Inteligencia Artificial

RPA no es Inteligencia Artificial, pero si se apoya en técnicas de Inteligencia Artificial y en la tecnología cognitiva para trabajar. RPA se sirve de OCR (Reconocimiento Óptico de Caracteres), reconocimiento de imágenes, clasificación, identificaciones de escritura a mano, categorización, visión por computador y análisis de textos para poder realizar las mismas acciones que un ser humano.

Se podría realizar la automatización sin recurrir a la Inteligencia Artificial pero lo que la Inteligencia Artificial aporta es una mayor naturalidad en las interacciones del robot con la interfaz gráfica de usuario. [2]
La Inteligencia Artificial favorece a la precisión y potencia de los robots. Por ejemplo, utilizando el reconocimiento de patrones, formas e imágenes, resulta muy útil para ejecutar aquellas acciones visuales que un usuario manejaría fácilmente con el ratón. Usando la analítica de textos, se pueden mejorar ciertos contenidos textuales transformando los datos que no estén estructurados en datos estructurados.
Además, se espera que en un futuro el Machine Learning aporte mejoras a RPA, como el futuro caso de automatizar decisiones, ya que ahora mismo, el robot se sigue por las reglas y sigue el flujo de las acciones que ya tiene programadas, sigue un algoritmo determinista. Sin embargo, con la introducción de la Inteligencia Artificial en el ámbito RPA, el robot podría ir aprendiendo según los casos que vaya automatizando y en base a ello, poder automatizar las decisiones que pudiera tomar y cambiar su comportamiento, al igual que procesar todo tipo de datos sin que tengan que estar únicamente estructurados.
Ahora mismo, en lo que más destaca la Inteligencia Artificial con RPA es en el procesado inteligente de documentos ya que utilizando las tecnologías de Inteligencia Artificial se puede avanzar en el reconocimiento, procesado y estructurado de textos de una manera rápida, eficaz y segura. La automatización se basa principalmente en el manejo y transformación de datos de una forma sencilla. Esto significa que la extracción automatizada de los datos es un aspecto clave en toda automatización.

2.3 Procesado inteligente de documentos

El empleo de tecnologías de Inteligencia Artificial como Computer Vision, Machine Learning, el procesamiento del lenguaje natural y técnicas de reconocimiento de caracteres hace mucho más seguros y eficientes los procesos que se programan en los robots. [5]
Las aplicaciones de estas tecnologías en el ámbito de RPA, son destacadas en el reconocimiento de documentos. Existen múltiples formatos, diferentes fuentes y escrituras que se pueden aplicar en un documento de la misma índole como puede ser una credencial o certificado. La rapidez, reducción de tiempo y eficiencia, son áreas en las que las personas son más inteligentes que una máquina a la hora de reconocer el contenido de los documentos y realizar acciones sobre ellos.
OCR, Reconocimiento Óptico de Caracteres permite convertir a texto aquellos textos localizados en fotografías o documentos escaneados. Sin embargo, para que esta tecnología funcione, los documentos deben cumplir unas características. Es esencial que la calidad del documento sea buena, con caracteres alineados y fácilmente reconocibles. También, que el texto sea estructurado o semiestructurado para detectar patrones.
Cuando una persona lee un documento, cumple una serie de pasos. En un primer vistazo, se buscan los focos donde localizar la información. Después se extrae información relevante de esos focos detectados y posteriormente, se

analiza la información extraída. Las herramientas Inteligencia Artificial cubren este mecanismo con IDP, el Procesado Inteligente de Documentos.
El Procesado Inteligente de Documentos actúa en una primera fase con Computer Vision para procesar la fuente de información y analizar que pueda ser útil, y luego se dedica a clasificar y extraer la información relevante de los textos, simula exactamente el procedimiento de una persona ante la lectura de un documento.
En primer lugar, Computer Vision analiza y procesa una fuente de imagen con herramientas y algoritmos de procesamiento de imágenes. Lo que realiza finalmente es la eliminación de imperfecciones y ruido de la imagen, ajustando parámetros como el zoom, contraste, y calidad con el fin de obtener una imagen más fácilmente legible después de su procesamiento. Esta tecnología igualmente es aplicable para documentos.
En segundo lugar, se procede a la clasificación y extracción de los textos para realizar la búsqueda o reconocimiento de patrones, que es el objetivo. Para ello, algunos indicadores en cualquier documento son clave como por ejemplo los campos libres, encasillados, selectores... pues muestran diferentes formas de presentar la información y dependiendo de lo que se esté buscando puede ser un patrón indicativo.
Por último, el sistema debe realizar un número de pruebas considerable con documentos de la misma índole para mejorar su precisión y aportar soluciones de calidad ya que en los documentos, se pueden encontrar imperfecciones, rayados, blancos y no debería ser una dificultad para conseguir el final procesado del documento.

2.4 Diferencia de RPA con la automatización clásica

Para realizar automatizaciones de procesos, las herramientas clásicas serían la ejecución de scripts y macros. Las macros y los scripts se basan en la ejecución de instrucciones para la realización de tareas de forma automática. Están programados en cualquier lenguaje de programación, básicamente son fragmentos de código que realizan las acciones de un ser humano con un ordenador.
RPA se diferencia de ellos porque no es lineal ni fijo, sino que es un ámbito dinámico y más intuitivo, por este motivo, el desarrollo con RPA sea mucho más rápido que con cualquier software tradicional. Respecto a la forma de configuración, una arquitectura basada en RPA consta de dos partes principalmente, el orquestador que monitoriza los robots de software, y los propios robots que ejecutan las instrucciones con las que se les programe. Sin embargo, en la automatización tradicional se basa en fragmentos de código.
Los robots de RPA no están programados con instrucciones de código como cualquier tipo de software, sino que usan pasos demostrativos. Dentro del software proporcionado por las aplicaciones de automatización, existen una serie de instrucciones que permiten programar los robots para ejecutar sus acciones sin necesidad de tener conocimientos elevados de programación. De esta manera, un perfil no técnico podría configurar un robot teniendo conocimientos muy básicos de programación ya que lo que se pretende realizar

es una automatización. Para ello, se entrena al robot dándole a seguir los pasos que realiza cualquier operador para la realización de las tareas que se quieren programar por medio de instrucciones ya existentes en el programa que se use.

2.5 Funciones y usos de RPA

Los robots de software RPA con capaces de imitar la mayoría de las acciones de un ser humando frente a una interfaz gráfica de un sistema informático. Las acciones básicas que pueden realizar son el movimiento de archivos y carpetas, copiado y pegado de datos, cumplimentación de formularios, extracción y procesado datos en documentos, leer, enviar y descargar archivos adjuntos en emails, etc.
Además, los robots se pueden conectar a aplicaciones, abrir navegadores y manipular páginas web, acceder y modificar bases de datos, descargar archivos de páginas o aplicaciones web, hacer cálculos, cumplimentar formularios, archivar transacciones completadas, dar de alta, actualizar o eliminar contenido en aplicaciones, pasar datos entre sistemas, repetir tareas concretas en aplicaciones indefinidamente. . .
RPA no sólo sirve para automatizar tareas sencillas, sino que se pueden enlazar varias tareas para dar lugar a un flujo de proceso que resuelva un caso de negocio. Esto se puede realizar como cualquier problema resuelto gracias a la programación usando variables, bucles, condiciones y tratamiento de errores. Las soluciones basadas en RPA se utilizan generalmente para procesos de gran volumen, repetitivos, basados en reglas y con datos existentes en el mundo digital.
Hay que tener en cuenta que los robots no aprenden según se les vaya entrenando con la realización de procesos, sino que solamente los realizan porque han sido programados para ello. Por lo tanto, se deben controlar las posibles excepciones que puedan surgir durante la realización de una tarea ya que el robot no sabría como tratarlas sin que un técnico antes se lo haya indicado. También se debe mencionar que los robots de software no modifican las pantallas de los sistemas informáticos o los sistemas y ficheros con los que interactúan, simplemente interactúan con ellos.
Aplicar soluciones de RPA a cualquier proceso reduce los errores humanos, el tiempo y esfuerzo de los trabajadores a cero. Libera recursos de los sistemas digitales y las horas de trabajo de las personas, pudiéndose dedicar a realizar otras tareas. Mejora la producción y el servicio de los procesos creados al ser más eficientes.
Para poder proporcionar soluciones óptimas de RPA se deben seleccionar procesos adecuados que puedan automatizarse y gestionar y controlar las excepciones que se puedan generar. RPA es una iniciativa de automatización que permite una rentabilidad y efectividad inmediatas al aplicarlo en cualquier organización. Permite obtener beneficios rápidos con una inversión inicial mínima, aportando soluciones que automatizan procesos a un coste y tiempo menores del que previamente se estipula con otros sistemas de TI. Aprovecha la infraestructura existente sin causar interrupciones en otros sistemas subyacentes, lo que sería costoso y difícil de reemplazar. Es altamente escalable

y flexible, se adapta a los cambios de negocio.
RPA se aplica principalmente a las industrias relacionadas con los servicios financieros y bancos, seguros, manufactura, alta tecnología y comunicaciones, energía y servicios públicos. Los procesos que suelen automatizarse son la activación de tarjetas, reclamos de fraudes, preparación de nuevos negocios, creación de reportes automáticos, reconciliación de sistemas, generación de facturas, gestión de pedidos, informes de calidad, creación y configuración de cuentas entre muchos otros.

2.6 Tipos de automatización

Automatizando con RPA, podemos distinguir dos tipos de automatización, la atendida y la desatendida. A simple vista, se puede decir que la automatización desatendida es la que no requiere de la presencia de una persona en la ejecución del robot, al contrario de la atendida.
Al analizar las acciones que una persona realiza ante el escritorio de un ordenador, ya se puede proporcionar una ligera idea sobre qué tipo de automatización se va a necesitar para resolver un proceso, y lo que es más importante, identificar qué procesos son candidatos para automatizar con RPA. No todos los procesos imaginados se pueden dejar en manos de un robot, pero sí que un equipo de profesionales puede trabajar junto a los robots y complementarse.
Existen un montón de soluciones útiles para satisfacer cada problema de automatización, lo que se debe tener conciso es cuál es el nivel de automatización que se necesita para cada proceso. Conociendo el nivel de automatización de un proceso, se puede aplicar el tipo de automatización que mejor se adapte a él.
La automatización robótica de procesos desatendida diseña robots que automatizan procesos completos, en los que ninguna intervención ni decisión humana es necesaria durante su flujo. El funcionamiento del robot es ininterrumpido y ninguna persona tiene por qué preocuparse de los resultados finales obtenidos. Este tipo de automatización se utiliza para procesos que siempre van a seguir los mismos pasos y para los que no es necesaria la toma de decisiones.
Por el contrario, también existe la automatización atendida. Esta automatización es de tipo parcial puesto que es aquella en la que la presencia de una persona es necesaria durante la ejecución del robot. Es decir, una persona tiene que estar guiando y siguiendo los pasos que el robot realiza ante la pantalla del ordenador porque se va a necesitar la toma de decisiones en las que el robot no puede participar.

2.7 Beneficios comerciales de RPA

El desarrollo de robots de software con RPA ofrece rentabilidad directa, por lo tanto, mejora la efectividad y producción de las organizaciones e industrias. RPA pone soluciones a cualquier proceso, asimismo, transforma y racionaliza el flujo

de trabajo de la organización.
Construir robots de software es permitir la escalabilidad y flexibilidad de las empresas, ofreciendo respuestas rápidas y personalizadas a las necesidades encontradas.
Los robots de software son fáciles de desarrollar y se integran perfectamente en cualquier sistema. Informan constantemente de su progreso, son preventivos operacionalmente y ayudan a la estrategia de crecimiento y transformación digital de la organización.
Los beneficios que proporcionan los robots de software RPA en el ámbito empresarial son los siguientes:

- Exactitud y precisión: los robots están programados para seguir las normas y cumplir su objetivo, no cometen errores en sus cálculos y son consistentes.
- Continuidad sin errores: una vez están configurados, los robots se ejecutan sin fallo, reduciendo la exposición al riesgo en una organización. Todo lo que realizan está monitorizado y en cualquier momento se puede modificar para que operen de acuerdo con nuevas regulaciones y estándares.
- Gran escalabilidad: los robots pueden realizar cantidad de operaciones en paralelo, desde entornos de escritorio de los sistemas digitales, hasta en sistemas en la nube. Las soluciones de RPA pueden ser del tamaño que se necesite y se pueden desarrollar nuevos robots con costes mínimos de acuerdo con el flujo de trabajo y la estacionalidad.
- Incremento de la velocidad y productividad: los primeros que aprecian los beneficios de RPA son los empleados ya que RPA les sustituye las actividades de gran volumen, basadas en reglas y repetitivas, ejecutándolas en menor tiempo, utilizando menos recursos y eliminado la posibilidad de cometer errores.
- Disponibilidad: funcionan ininterrumpidamente las 24 horas del día, todos los días del año.
- Potenciador de talento: ya que los robots son innovadores y fortalecen la estrategia digital de las empresas.
- Ahorro: las herramientas de RPA reducen hasta un 80% los costes de procesamiento. En menos de un año, la mayoría de las empresas ya han ganado todo lo que invirtieron en la transformación digital utilizando RPA.

RPA cubre la automatización de muchos procesos y ofrece ventajas competitivas para las organizaciones ya que se crean resultados potentes a escala limitada. De esta forma, se consigue la transformación digital de la organización y un avance en el camino hacia la futura aplicación de la Inteligencia Artificial en RPA.
Los robots de software RPA no son difíciles de desarrollar y mantener. La seguridad y escalabilidad es lo que los caracteriza.

2.8 La seguridad en proyectos de RPA

RPA está siendo adoptado por las empresas para mejorar su productividad, eficiencia y resultar ser más competitivas. Esta tecnología ahorra costes, tiempos de ejecución, asegura la fiabilidad y produce resultados con cero fallos. No obstante, cuantas más funcionalidades tenga un proyecto de RPA, tendrá más riesgos de seguridad, sobre todo en la exposición de los datos.
Los riesgos de seguridad afectan a todo el ciclo de vida de un robot. El ciclo de vida de un robot comienza cuando se identifica una necesidad en una organización y acaba con la integración del robot en sus sistemas. Pasa por las fases de especificación, diseño de la arquitectura, implementación, las fases de pruebas unitarias y de integración y, por último, la aceptación del robot. En las fases implementación e integración es donde hay que tener más cuidado, sobre todo con el manejo de los datos. Se tiene que definir de una forma segura el acceso a aquellos datos confidenciales.
Por otro lado, es aconsejable seguir buenas prácticas para conseguir que los robots tengan una buena automatización, integración con los sistemas y seguridad. Las buenas prácticas en el ámbito de RPA se pueden clasificar en tres factores: el gobierno, la arquitectura y la seguridad de la información. [4]

- El gobierno
  Es esencial establecer un marco de gobierno con funcionalidades y responsabilidades que rijan el comportamiento de los robots.
  El comportamiento de un robot consiste en la imitación de las acciones de un humano, por lo tanto, se deben seguir políticas de gobierno ya definidas antes de empezar la fase de implementación. Una organización no se puede permitir la pérdida o robo de información ni el acceso no autorizado a ciertos contenidos de información o datos.
  Primero, se debe crear una lista de requisitos de seguridad, controles de acceso y patrones de autenticación. Es decir, establecer claramente a qué datos se puede acceder, si se necesitan credenciales para su acceso y quiénes pueden acceder a ellos. Después, se debe construir la estrategia y requisitos de seguridad que se quieren seguir. Finalmente, adoptar la estrategia de seguridad en los robots.

- La arquitectura
  Se tiene que revisar el diseño de la arquitectura antes de pasar a la fase de implementación para asegurar que los controles de seguridad establecidos están integrados.
  En el propio diseño de la arquitectura, se deben analizar los riesgos de seguridad que pueda suponer la arquitectura seleccionada de los robots con las conexiones con otras aplicaciones y entornos, la monitorización y la ejecución de los robots.
  Algunas medidas que deben tenerse en cuenta en el diseño de la arquitectura son la limpieza de las variables y de los archivos temporales cuando haya terminado la ejecución del robot, que el registro de auditoría no contenga información confidencial, la asignación de identificadores únicos a cada robot cuando se acceda a terceras aplicaciones o archivos

y la existencia de un repositorio de robots durante todo el ciclo de vida del proyecto RPA.
Luego, en la fase de pruebas, es conveniente que antes de realizarlas en el entorno de producción, se haya exigido una especificación de las funcionalidades del robot, revisiones de código y listas de casos de pruebas unitarias y de integración.

- La seguridad de la información
  Este punto trata la gestión de credenciales y los accesos para los robots, la segregación de responsabilidades y la auditoría en la organización a la hora de utilizar RPA. Algunos puntos que deberían seguirse serían:
  - Por un lado, las contraseñas ya que pueden ser posibles fugas de seguridad en una organización. Por ello, sería conveniente una asignación coherente de contraseñas, así como la gestión y el acceso de las contraseñas mediante un administrador de credenciales que las cifre
  - Por otro, la implementación de controles de seguridad estrictos para no desproteger las credenciales cuando estén en tiempo de ejecución de los robots, y así, proteger la información sensible o confidencial.
  - También, la supervisión de los datos confidenciales utilizados por los robots, para verificar que se hayan cumplido las políticas de uso de los datos. Además, ejecutar un análisis para detectar posibles vulnerabilidades de la plataforma y debilidades técnicas.
  - Por último, recopilar datos de registro de robots. Los tiempos de ejecución para detectar ejecuciones fallidas o tiempos anormales, el número de accesos a sistemas y el uso de cuentas privilegiadas.

2.9 Softwares RPA. ¿Cómo elegir el adecuado?

La elección del software para poner en marcha la automatización es un aspecto importante que se debe gestionar una vez se tengan decididos los procesos a los que se quiere poner solución mediante la automatización, ya que es algo que va a afectar no sólo al resultado sino al desarrollo, tiempo de creación y mantenimiento del software producido. [3]
Algunos aspectos para tener en cuenta son los siguientes:

- Los conocimientos previos de RPA en la organización, ya que dependiendo de la profundización que lleve la organización con RPA, conviene elegir una plataforma sencilla o más compleja en cuanto a arquitectura.
- Las integraciones del software RPA con otras aplicaciones que se necesiten en los procesos que quieran automatizarse. En este caso, todos los softwares que facilitan soluciones de RPA permiten integraciones con otras plataformas, lo único que variaría entre unos y otros sería la rapidez, que para muchas aplicaciones puede ser un factor clave.

- Las tecnologías que soporta el software elegido. Se debe tener en cuenta qué tecnologías ya se usan o se quieren empezar a usar en una organización para poder seguir trabajando con ellas sin tener que cambiar necesariamente todo el entorno al incorporar soluciones de RPA en sus productos.

- El mantenimiento posterior una vez desarrollado los robots de software, porque, se debe tener presente que una vez se hayan programado los robots, éstos están en constante estado de mantenimiento por posibles modificaciones que puedan surgir en los procesos y cambios de mejora que quieran aplicarse. Por ello, es conveniente trabajar con un software que no sea dependiente y que permita realizar cambios con agilidad.

- Otro aspecto importante de mencionar, la documentación y comunidad que proporcione el software. Muchos softwares de RPA al ser privados no proporcionan la suficiente documentación o foros en donde se pueda investigar y preguntar sobre soluciones y problemas. Si no se dispone de personas especializadas en trabajar con esos tipos de software, sería conveniente trabajar con aquellos que estén bien documentados y que tengan una comunidad a la que poder acceder en caso de duda y encontrar soluciones a los problemas, pues se eliminaría tiempo en el desarrollo de los robots al no tener que detenerse en aprender algo totalmente nuevo si se encuentran rápidas soluciones a los problemas que vayan surgiendo durante la programación.

Entre las plataformas de software RPA podemos mencionar WorkFusion, BluePrism, Automation Anywhere y UiPath. Todas ellas ofrecen los mismos resultados, pero la elección de uno u otro ya dependerá de la complejidad de los procesos que se automaticen, el grado de conocimientos sobre RPA, integraciones y tecnologías que soportan y comunidad y documentación que ofrezcan.

2.10 Comparativa entre softwares RPA

En este apartado se expondrá brevemente información sobre los cuatro tipos de software que nos permiten realizar soluciones de RPA. Cuáles son los lenguajes de programación en los que se fundamentan, la arquitectura de su plataforma para el diseño, programación y monitorización de los robots. Si están orientados para programadores, por qué destacan sobre otros y algún inconveniente de uso.

- BluePrism: es una compañía fundada en 2001. Es el software con más años y madurez en el ámbito de RPA. En cuanto al lenguaje de programación, la plataforma está construida sobre Microsoft .NET Framework. Su arquitectura está formada por Process Studio para el diseñar los procesos, Object Studio para la programación y la interactuación con las aplicaciones y Control Room ejecutar los robots y llevar su monitorización. En lo que destaca es en la automatización de procesos complejos de back office, ya que automatiza cualquier aplicación

y soporta cualquier plataforma. Sin embargo, uno de sus principales problemas es ese, que sólo admite la creación de robots de back office y no permite de front office.
En la siguiente imagen se aprecia el logo de la compañía de software BluePrism.

Imagen 2.1: Logo BluePrism

- Automation Anywhere: se fundó en 2003 y tiene amplia experiencia en RPA. La metodología de programación es basada en scripts, por lo que está más bien orientado a desarrolladores al no ser nada visual, además de que no existe una comunidad o documentación abierta para facilitar su aprendizaje. Con su plataforma, se pueden programar procesos tanto de back office como de front office. Se diferencia de las demás porque permite analizar datos estructurados y semiestructurados al ser capaz de procesar el lenguaje natural. Su principal inconveniente es que tiene problemas a la hora de detectar las ejecuciones fallidas.
  En la siguiente imagen se observa el logo de Automation Anywhere.

Imagen 2.2: logo Automation Anywhere

- UiPath: se fundó en 2005 y está en crecimiento. No se basa en un lenguaje de programación en concreto, tiene una buena interfaz de usuario muy intuitiva y visual, entonces está orientada a cualquier usuario que desee aprender RPA. Dispone de una comunidad activa y documentación donde encontrar información útil para nuevos programadores. La arquitectura de la plataforma está compuesta del Studio para diseñar los flujos de los robots, el Orquestador para llevar a cabo la monitorización y los

propios robots. Destaca en las automatizaciones sobre el paquete Microsoft Office y automatizaciones en escenarios remotos. Además de que tiene integración nativa con OCR y ML. Los inconvenientes al usar UiPath serían al debuggear, ya que no se asemeja en nada a lo que está acostumbrado un programador y su interfaz gráfica tan visual ya que para procesos grandes podría ser un problema al no visualizar la completitud del fujo de programación del robot.
A continuación, se incluye el logo de UiPath.

Imagen 2.3: Logo UiPath

- WorkFusion: es compañía que se fundó en 2010, con sede en Nueva York. Es el software RPA de menor antigüedad. Se fundamenta en un lenguaje de programación orientado a objetos sobre Java, Groovy. Esto implica que esté más orientado a programadores ya que se basa en un lenguaje específico de programación y no para usuarios con conocimientos básicos, aunque, también favorece su aprendizaje la documentación, formación y materiales gratuitos que se proporcionan en su página web. Su plataforma está formada por tres componentes, los propios robots, el Studio para facilitar su diseño y programación y el Control Tower para orquestar la ejecución de los robots. Destaca porque en su plataforma SPA (Smart Process Automation) cuenta con tecnología Machine Learning, que puede ser útil para la creación de robots que requieran de aprendizaje automático (aunque realmente esta característica aún está por mejorar).
  En la siguiente imagen se aprecia el logo de WorkFusion.

Imagen 2.4: Logo WorkFusion

Capítulo 3

Desarrollo

3.1 Tipo óptimo de organización

La búsqueda de una organización adecuada para enfocar el trabajo no es una tarea dificultosa, únicamente se tienen que cumplir algunas pautas para poder desarrollar robots en ella. Para empezar, es principal que esa organización necesite integrar RPA como parte de su funcionamiento, identificando necesidades en las que se pueda aplicar. Aunque se hayan identificado necesidades que se cubran gracias a RPA, se tienen que poder integrar las funciones de RPA en la organización. Esto es, se tiene que comprobar si son viables las soluciones de RPA antes de decantarse por ellas.
Este trabajo podría dedicarse para cualquier tipo de organización ya que se pueden buscar casos de uso para automatizar en muchas de ellas. Para centrarse en algo en concreto, y plantear ejemplos reales, se ha pensado en una organización que se dedique a la seguridad física. Esta organización requiere de RPA porque en ella existen procesos de alto volumen, repetitivos y que cumplen los requisitos para ser automatizados. Además, se empleará una automatización desatendida, porque no se van a necesitar las intervenciones de un humano.
En esta organización existen cámaras de seguridad por todo su perímetro y se controlan los accesos de las personas que quieran sobre pasar su perímetro. Cuando una cámara detecta a personas no identificadas u objetos, se produce una alarma que pasa a ser tratada por un operador, al final del tratamiento de la alarma, se genera un reporte para hacer constancia de lo sucedido.

3.2 Casos de uso

Como casos prácticos, se expondrán algunos ejemplos que se puedan aplicar a situaciones reales, en las cuales se podrán implementar robots de software que ejecuten un flujo de tareas, sustituyendo el trabajo de los operadores, siendo precisos, rápidos y sin riesgo a cometer errores.

- Caso práctico 1:
  Se requiere nuevo personal para trabajar en la empresa y se buscan candidatos para realizar entrevistas presenciales en las oficinas. Los entrevistados antes tienen que identificarse para que al entrar puedan acceder sin problemas. Al entrar en el perímetro de la organización, existen una serie de cámaras que sólo permiten el acceso a personas autorizadas, por ello los entrevistados deben de enviar una fotografía para que puedan ser detectados como personas no intrusas. A parte de eso, deben generarse un perfil en la página web de la organización para crearse su currículum online y estar pendientes de notificaciones por parte de la empresa.
  ¿Cómo se procede?
  Las personas que van a ser entrevistadas envían un correo electrónico con un asunto en concreto a una dirección de la organización, en este correo incluyen toda la información necesaria. El robot de software se encargará de leer los correos electrónicos y extraer los archivos adjuntos o la información útil que contengan.
  Por un lado, se añade una entrada por persona en la base de datos con la información y fotografía para identificar a los sujetos por las cámaras a la entrada. Por otra parte, se reutiliza esa información para rellenar un formulario en la web y crear un perfil por cada persona que vaya a ser entrevista. Finalizado el proceso, el robot envía un correo electrónico a cada persona con las credenciales para poder acceder a sus perfiles.

- Caso práctico 2:
  En la organización elegida existen cámaras de seguridad que detectan la presencia de personas, se comprueba si esas personas son intrusos o personal autorizado. Cuando salta una alarma en alguna de las cámaras, una persona responsable es encargada de revisar el vídeo y contestar una serie de preguntas acerca de la alarma activada. Estas preguntas están programadas en un diagrama de flujo, a su vez, se van registrando las respuestas que va dando el operador en un reporte. Una vez finalizado el tratamiento de la alarma, se puede descargar este reporte para usarlo posteriormente. Lo que se pretende realizar mediante la automatización, es el seguimiento de un diagrama de flujo que trate la alarma detectada. Este tipo de automatización es atendido debido a que un operador debe manejar el robot para llegar al resultado final y obtener el reporte con la información detallada sobre la alarma tratada.

3.3 Diseño del sistema

Antes de empezar a profundizar con el diseño de los robots, se va a explicar con qué software van a crearse los robots y los motivos que han llevado a la decisión. El software elegido para tratar los casos de uso planteados es UiPath. Se ha escogido principalmente por la buena documentación y la comunidad a la que se puede acceder. Este es un aspecto clave para iniciarse con RPA y poder desarrollar los robots de software, apoyándose en la documentación para

empezar con RPA y recurriendo a la comunidad en caso de duda. Además de este importante factor, también es un software que proporciona algunos tipos de licencia libres, a diferencia de muchos otros. Tiene integración nativa con OCR y destaca en automatizaciones sobre el paquete de Microsoft Office, por lo que es una ventaja respecto a otros.
Construir un robot de software es una forma de desarrollo de software, aunque de una forma distinta a lo tradicional. Para diseñar la arquitectura de un robot es necesario tener conocimientos de Ingeniería del Software puesto que se trata de un desarrollo software principalmente. Tampoco hay que dejar de lado que el fin de la automatización es para resolver procesos de negocio, y en este apartado también es útil esta materia para elaborar el ciclo de vida del robot de software, así como la planificación y los costes del proyecto de implantación de los robots en la organización.
Basándose en las técnicas de Ingeniería del Software, se pueden utilizar las mismas fases para el desarrollo de los robots que para cualquier tipo de implementación. Las fases para llevar a cabo la construcción de los robots de software serían las que siguen a continuación: [6]

- La elaboración del diseño del workflow: los robots de software automatizan procesos, antes de comenzar a realizar ningún tipo de implementación, se va a definir bien el proceso que se quiere automatizar. Para esquematizarlo se utilizará un flujo o workflow. Después, se optimizará el workflow para aprovechar al máximo el rendimiento y los recursos empleados.

- La estructuración y división los procesos en módulos: algunos workflows pueden resultar simples y sencillos mientras que si estamos ante un proceso de negocio amplio pueden resultar voluminosos e interminables. En este caso, se tiene que dividir el problema en diferentes casos menos complejos, con una lógica más simple. Una buena forma de desarrollar software es la descomposición de un problema difícil en otros más pequeños y sencillos, esto se traduce por la división de la programación en módulos y buscando su reutilización de los componentes.

- El manejo de las excepciones: siempre suceden errores y casos inesperados cuando se está desarrollando software. El tratamiento de las excepciones es una salida útil y beneficiosa para resolver este tipo de situaciones. El robot de software tiene que estar preparado para reaccionar de una manera adecuada ante situaciones inesperadas y errores que puedan surgir durante su ejecución.

- Limpieza del sistema: esta fase se refiere a la optimización de los recursos de cómputo de la máquina. Se puede conseguir liberando aquellos recursos de memoria que ya no sean necesarios después de haberlos utilizado. Por ejemplo, cerrando los ficheros, bases de datos, ventanas, aplicaciones cuyo tratamiento ya haya sido finalizado.

- Control de versiones y backup: en todo desarrollo de software se tiene que seguir un control de versiones y copias de seguridad periódicas para no perder todo lo que se llevaba desarrollado, así como para gestionar los cambios que se van realizando durante la implementación de los robots.

UiPath Studio sirve para desarrollar soluciones de automatización, simples y complejas. Estas soluciones pueden ser de integración con aplicaciones o de creación de aplicaciones, tareas administrativas y procesos de negocio. En UiPath Studio se implementan flujos de trabajo en archivos con extensión ".xaml", cada robot está compuesto por varios de ellos. Cada archivo realiza funciones diferentes que van constituyendo cada uno de los pasos del robot, siguiendo los pasos del workflow de acciones y simplificando a la vez que estructurando la parte de programación.
El programa tiene una interfaz amigable y muy intuitiva. En la pantalla principal del programa, se pueden visualizar cinco paneles, en la parte superior, inferior, derecha, izquierda y en el centro.

- En el panel superior, se puede observar dos ventanas. Una se corresponde con el diseño en la cual se puede observar una lista de iconos que permiten crear nuevos archivos, guardar, manejar los paquetes de referencias, grabar y publicar entre otros. La otra ventana se corresponde con la parte de debug en la que se listan iconos correspondientes a la parte de testeo del código.

- El panel inferior, está formado por las ventanas donde se muestran las salidas, las listas de errores y puntos de ruptura cuando se realizan ejecuciones de robots.

- El panel de la izquierda está formado por la carpeta con el proyecto abierto y otro con un panel de actividades.
  La carpeta donde se encuentra el proyecto de cada robot contiene los archivos ".xaml" que lo forman, así como demás estructuras de carpetas o archivos que se utilicen para el funcionamiento del robot. En cada archivo se pueden crear secuencias, diagramas de flujo o máquinas de estados. Dependiendo de las funciones, se elige uno u otro, aunque lo que más va a usarse serán las secuencias ya que permiten añadir y programar métodos con total libertad.
  El apartado de actividades tiene métodos que se pueden seleccionar y arrastrar hacia la secuencia que se esté desarrollando. Estos métodos vienen agrupados según la funcionalidad. Para dar funcionalidad a una secuencia se utilizan tanto estos métodos predefinidos como código en lenguaje *Visual Basic*.
  Dentro del catálogo de actividades se tienen predefinidos los siguientes tipos de métodos:

  - Integración con otras aplicaciones
    - CSV para manejar archivos de tipo CSV.
    - Excel para procesar tablas.
    - Correo electrónico (IMAP, POP3, SMTP, Outlook), para leer, borrar, guardar, enviar mensajes.
  - Computer Vision, para buscar elementos existentes, resaltar, escribir en archivos entre otras funciones.
  - Orquestador, para manejar colas, procesos, flujos de trabajo, alertas.

  - Programación clásica, para el manejo de colecciones de datos, tablas, valores genéricos, métodos de tratamiento de strings y debuggeador.
  - Métodos del sistema, para controlar las funciones que dependen de la clase system como pueden ser los que se sirven del entorno, clipboard, conexiones con aplicaciones y scripts de *Visual Basic.*
  - Automatizaciones de interfaz de usuario, en este caso son métodos que trabajan con la pantalla como con los buscadores, algunos elementos de ratón, teclado y controles, tratamiento de imágenes, relaciones con OCR, texto y ventanas.
  - Eventos de usuario: de elementos, imágenes y del sistema.
  - Workflow: constituido por métodos para realizar invocaciones, máquinas de estados, controles de flujo, control de errores y puntos de ruptura.

- En el panel central se visualiza la secuencia que se esté construyendo.
- Por último, el panel de la derecha sirve para poder visualizar las propiedades de los métodos que se estén tratando y realizar modificaciones en ellos.

En la imagen siguiente se pueden visualizar los paneles nombrados anteriormente que consituyen la interfaz del programa UiPath Studio.

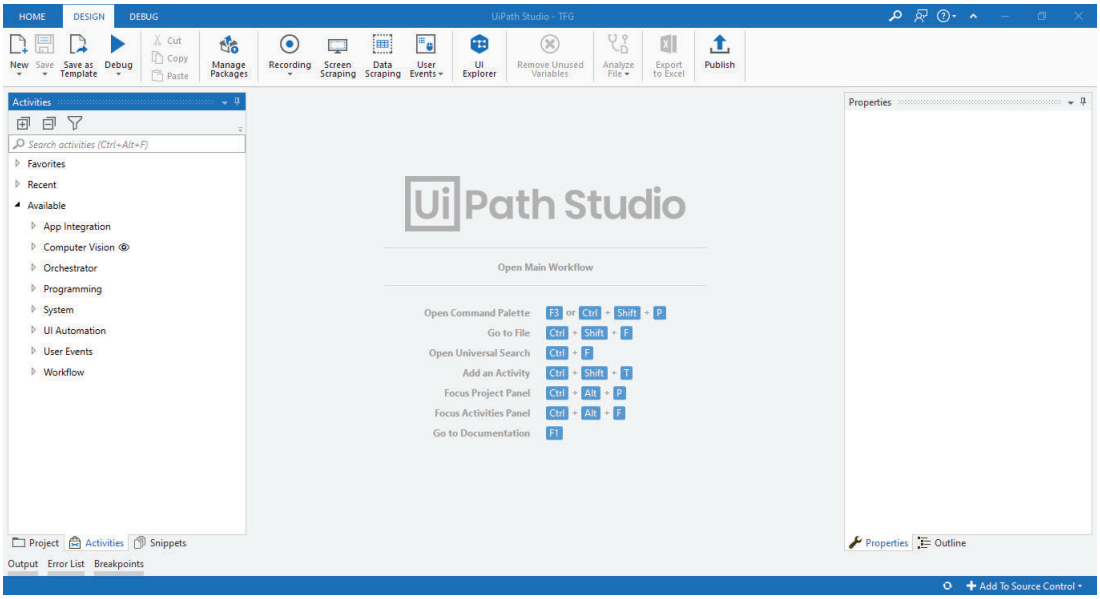

Imagen 3.1: Interfaz UiPath Studio

Para empezar a implementar los robots, se debe tener en cuenta que existen dos formas distintas para el desarrollo. Una sería creando un proyecto de automatización simple formado por procesos y la otra creando una librería que pudiera incorporarse a la funcionalidad de otras aplicaciones.

- Proyecto de automatización: pueden incorporar todo tipo de workflows, secuencias, máquinas de estados y diagramas de flujo. También se les puede añadir librerías como dependencias para usar sus funciones. Un proyecto típico se compone del archivo "Main.xaml", que será el que ejecute el workflow y el archivo con extensión ".json", que contendrá información sobre el proyecto. Según lo que se desee implementar, se irán añadiendo más archivos ".xaml" que implementen las funciones a desarrollar.
  Centrándonos más en el trabajo, se utilizarán los proyectos de automatización para desarrollar los robots. Cada archivo ".xaml" tendrá funcionalidades distintas para dividir el proceso en fases simples y el archivo "Main.xaml" o llamado con el nombre del robot en cuestión, será el que enlace todos archivos y los ejecute.

- Librería: es un paquete que contiene componentes reutilizables. Al igual que pueden usarse librerías y reutilizar sus métodos ya existentes, también se pueden crear bloques para integrarlos en otros proyectos o aplicaciones. Las librerías se guardan con extensión ".nupkg" y se pueden instalar como dependencias de workflows usando el manejador de paquetes.

Una unidad mínima de desarrollo/implementación de cualquier función es el workflow. Se trata de una secuencia de instrucciones que dan lugar a un resultado. Los workflows se pueden construir mediante secuencias, diagramas de flujo y máquinas de estados.

- Las secuencias son un tipo de desarrollo secuencial. Se basan en la construcción de procesos lineales que permiten ir de una actividad a otra a la vez que incorporar fragmentos de código. Pueden reusarse en otras secuencias, además.

- Los diagramas de flujo tienen una gran variedad de ajustes y son útiles para todo tipo de proyecto. La diferencia entre las secuencias y los diagramas de flujo es que estos presentan múltiples ramas y operadores lógicos que permiten desarrollar procesos complejos de negocio y conectar actividades de muchas formas.

- Las máquinas de estados son un tipo de automatización que usa un número finito de estados en su ejecución. La ejecución del robot se va moviendo por los diferentes estados según vaya cumpliendo las condiciones en las transiciones entre estados que permiten acceder a ellos. Contienen una entrada, estados y transiciones intermedias y el estado final.

3.4 Implementación del sistema

En los siguientes dos apartados se explicará como se han implementados los dos Robots de software utilizando UiPath. Para almacenar los proyectos e ir gestionando las versiones y modificaciones que se vayan creando durante el desarrollo, se utilizará un repositorio de código en GitHub.

3.4.1 Caso de uso 1 - Robot 1

Funciones que realiza el Robot 1:
Se han divido en dos partes. La primera parte consta del envío de información por parte de los usuarios, almacenado en base de datos de esta información y la creación de sus perfiles en el formulario web. La segunda parte consta de la generación de las credenciales, para que los usuarios puedan identificarse en la organización, y el envío de cada una de ellas a los mails correspondientes.

- Parte 1
  Primero, el robot busca en el servidor de correo los emails que contengan el asunto indicado. Después, descarga los archivos adjuntos que contengan el formato pedido y los almacena en una carpeta temporal. A continuación, se guarda la información que contienen los archivos adjuntos en sus correspondientes campos de la base de datos, uno a uno.
  Cada fila de la tabla de la base de datos se corresponde con una persona, y en cada columna se almacenan sus datos personales para identificarse en la organización.
  Por cada fila de la base de datos o persona se crea un perfil de usuario en el formulario web de la organización. El robot abre el formulario y va introduciendo en los campos los datos de cada usuario, cuando finaliza, se envían los datos y se crea el perfil. Realiza esta acción tantas veces como usuarios se tengan que dar de alta.

- Parte 2
  El robot general unas credenciales aleatorias para cada usuario y las almacena en la base de datos. Por cada usuario, se genera un archivo en el que se introducen sus credenciales. Finalmente, se envía a cada usuario este archivo.

Implementación del Robot 1:
A continuación, se va a explicar cómo se han implementado las secuencias en UiPath que constituyen el desarrollo en cada parte del Robot 1.
Se ha decidido la automatización del proceso mediante secuencias, ya que es la opción más adecuada para seguir los pasos que sigue el workflow ya que se trata de operaciones secuenciales que van dependiendo unas de las otas hasta acabar el proceso.
La primera parte, está constituida por las secuencias: ReadEmail.xaml, SaveToDataBase.xaml y WebProfile.xaml.

ReadEmail.xaml

Esta secuencia se encarga de la descarga de los archivos adjuntos a los emails que se envían a la dirección de correo electrónico de la organización.

- Estructura:

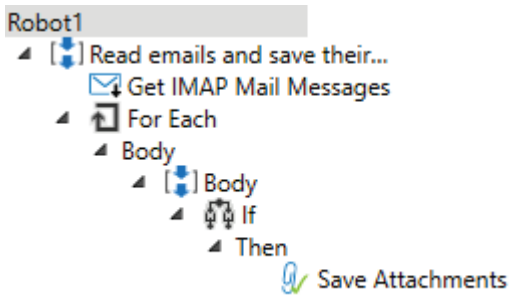

Imagen 3.2: Estructura de la secuencia ReadMail.xaml

- Variables:

Nombre	Tipo	Ámbito	Valor
Messages	List<MailMessage>	Todo el programa	Sin inicializar

Imagen 3.3: Variables ReadEmail.xaml

La variable *Messages*: utiliza las referencias *System.Collections.Generic.List* y *System.Net.Mail.MailMessage* para generar el tipo de mensaje y la lista que los contiene. En esta variable es donde se almacenan los mensajes leídos.

- Métodos utilizados:
  Get IMAP Mail Messages: se utiliza para leer los correos electrónicos.
  - Configuración del host
    Carpeta: Inbox/Entrada
    Puerto: 993
    Servidor: imap.gmail.com
  - Inicio de sesión
    Se añade el email que va a usarse y la contraseña. La contraseña es una clave que se ha generado para acceder a ese servidor de correo electrónico desde la aplicación UiPath, sin necesidad de introducir la contraseña de la cuenta real.
  - Opciones
    Se le indica que busque en los mails no leídos y una vez leídos los marque como leídos.
  - Salida
    Este método genera una lista con sus elementos del tipo MailMessaage. Esta lista debe estar declarada en la lista de variables.

En la imagen a continuación, se pueden observar ordenadamente las propiedades del método *Get IMAP Mail Messages*.

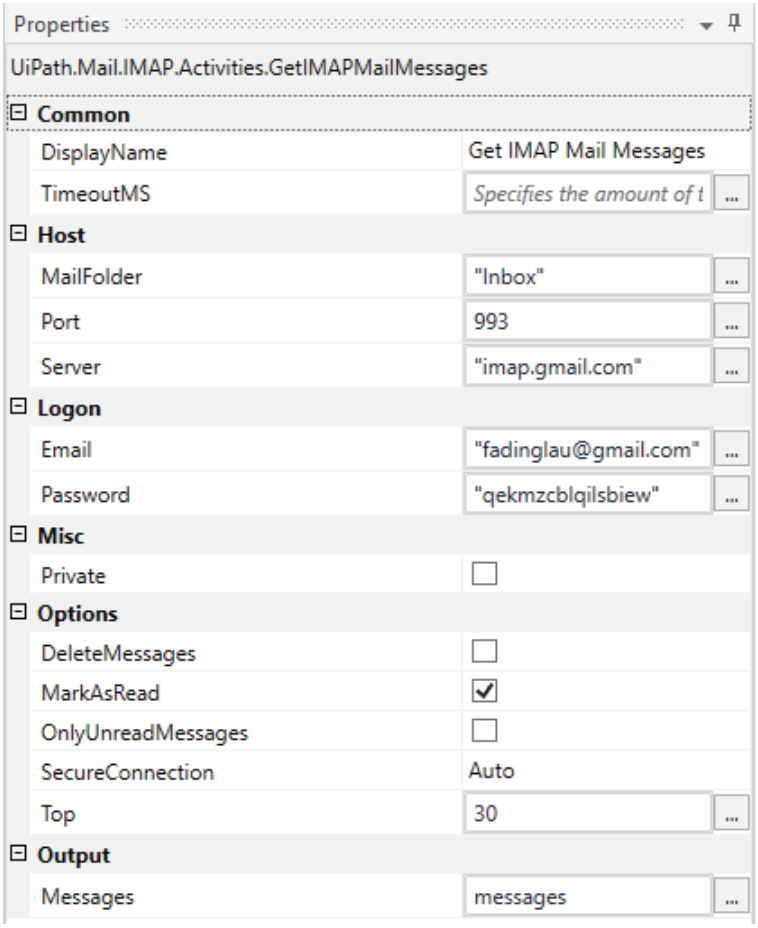

Imagen 3.4: Propiedades del método Get IMAP Mail Messages

For each: bucle donde se examinan los mails y se guardan los archivos adjuntos. Recorre cada mensaje de la lista de mensajes que se ha generado en *Get IMAP Mail Messages.* Por cada mensaje comprueba si su asunto contiene el asunto indicado. Si se cumple esa condición, entonces se guardan los archivos adjuntos que contengan los emails. La siguiente imagen muestra una captura del bucle tomada desde UiPath.

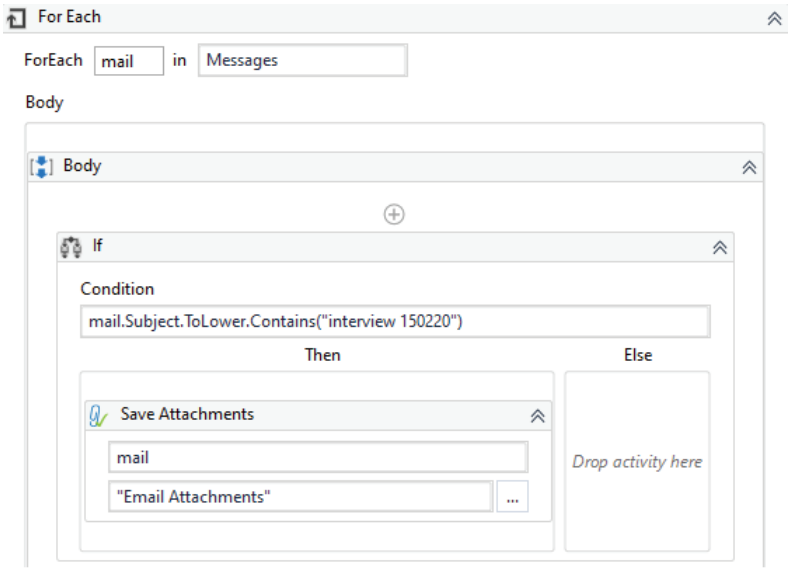

Imagen 3.5: Bucle For Each de ReadEmails.xaml

Save Attachments: este método guarda archivos en una ruta especificada.

- Entrada
  La ruta de la carpeta donde guardar los archivos adjuntos y la variable de tipo mensaje de donde se van a extraer.
- Opciones
  Se filtra por los documentos que tengan la extensión ".xlsx", ya que son estos los que se quieren almacenar.
- Salida
  Los archivos adjuntos almacenados en la ruta indicada.

- Manejo de excepciones:
  Se utiliza un bloque *Try-Catch*. En el caso de que se intenten leer mensajes con el asunto indicado y no existan, saltará una excepción junto a un mensaje. Si se han leído mensajes, se procede a la continuación de la secuencia. La siguiente imagen muestra el bloque *Try-Catch* agrupando la secuencia *ReadEmail.xaml*.

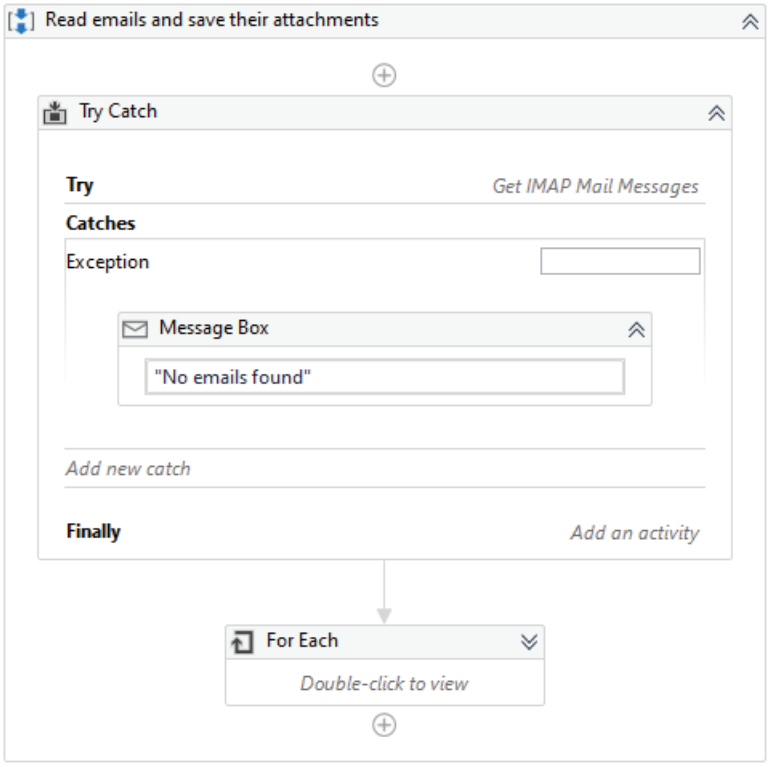

Imagen 3.6: Try-Catch en ReadEmail.xaml

- Limpieza del sistema:
  En este caso, al ser la primera secuencia que se ejecuta en el Robot 1, no es necesario realizar ninguna acción, ya que posteriormente se van a necesitar los archivos adjuntos que se descarguen.

SaveToDataBase.xaml
Esta secuencia guarda en la base de datos la información de cada usuario. Lee de cada archivo descargado en la anterior secuencia la información, y la guarda en los correspondientes campos de la base de datos.

- Estructura:

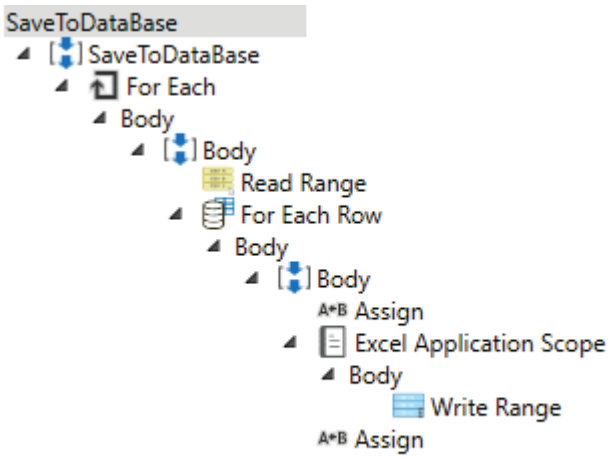

Imagen 3.7: Estructura de la secuencia SaveToDataBase.xaml

- Variables:

Nombre	Tipo	Ámbito	Valor
OutputExcel	DataTable	Todo el programa	Sin inicializar
Range	String	Todo el programa	Sin inicializar
FolderPath	String	Todo el programa	Path email attachments
CellNumber	Int32	Todo el programa	2

Imagen 3.8: Variables SaveToDataBase.xaml

La variable *OutputExcel* utiliza la referencia *System.Data.DataTable.* Es una variable en la que se almacena la lectura de la base de datos.
Las variables *Range* y *CellNumber* se utilizan para seleccionar la celda donde escribir en la base de datos.
La variable *FolderPath* se utiliza para indicar la ruta de la carpeta donde están los archivos adjuntos descargados de los emails.

- Métodos utilizados:
For each: bucle que recorre cada archivo en la carpeta indicada mediante la variable *FolderPath*.
  - Condición
    for each *item* in *Directory.GetFiles(FolderPath)*
  - Cuerpo del bucle
    Read Range: por cada elemento *item* que se va recorriendo en el bucle,

lee su rango y extrae la fila indicada. La entrada del método es el objeto *item* que se corresponde al archivo descargado del mail y la salida, de tipo *DataTable*, se guarda en la variable *OutputExcel*.
For each row: bucle que recorre el resultado de *Read Range*. Toma como entrada la variable *OutputExcel*. Recorre sus elementos, leyendo y almacenando la información correspondiente en las casillas de la base de datos.

* Condición
  for each *row* in *OutputExcel*
* Cuerpo del bucle
  Abre el Excel base de datos y escribe la información. Utiliza el método *Excel application scope*, que recibe como entrada la tabla de Excel base de datos. En el cuerpo del método, se utiliza *Write Range* que toma como entradas las variables *OutpulExcel* y *Range*, ya que le indican los datos y donde tiene que introducirlos.

Cada vez que el bucle principal realiza una iteración, la variable *CellNumber* va incrementando el número de fila de la base de datos.

- Manejo de excepciones:
  Se tiene que comprobar que se hayan descargado archivos en la carpeta indicada. La extensión de los archivos no es necesario comprobarla en este apartado ya que en la secuencia anterior sólo se realiza la descarga de los archivos con la extensión indicada. Ahora, simplemente se comprueba si hay algún archivo en la carpeta y si es así se continúa con la secuencia, guardando la información en la base de datos.
  Para realizar la comprobación, es suficiente con obtener el número de elementos de esa carpeta, si ese número es cero, se finaliza el proceso. A continuación, se muestra una imagen de la secuencia.

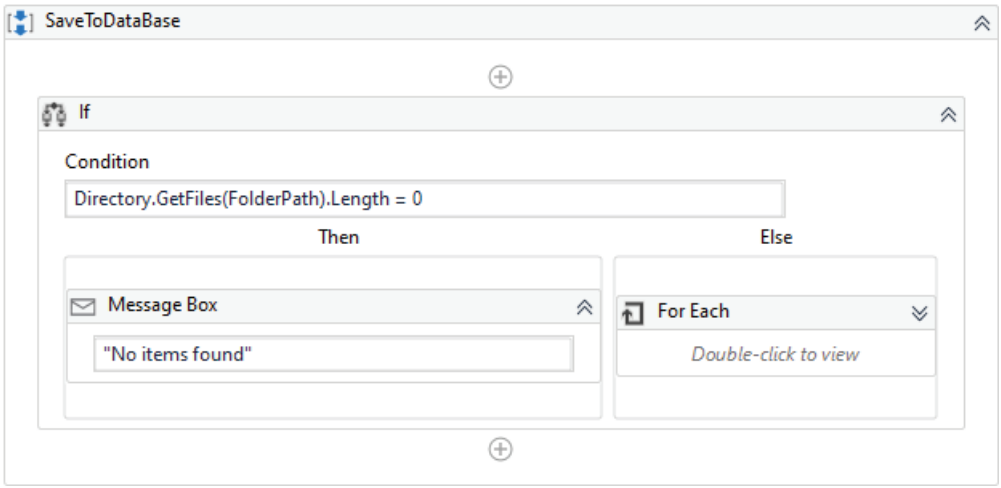

Imagen 3.9: Manejo de excepciones en SaveToDataBase.xaml

- Limpieza del sistema:
  En esta secuencia se eliminan los archivos descargados de la carpeta ya

que no van a ser útiles para la ejecución posterior del proceso del robot, además de que se deja la carpeta vacía para cuando se vuelva a ejecutar el robot.

WebProfile.xaml
En esta secuencia, se crea un perfil en la página web de la organización para cada usuario existente en la base de datos. Por cada usuario, abre un formulario web y va introduciendo la información requerida extrayéndola de la base de daos, después cierra el formulario y lo deja preparado para la siguiente iteración.

- Estructura:

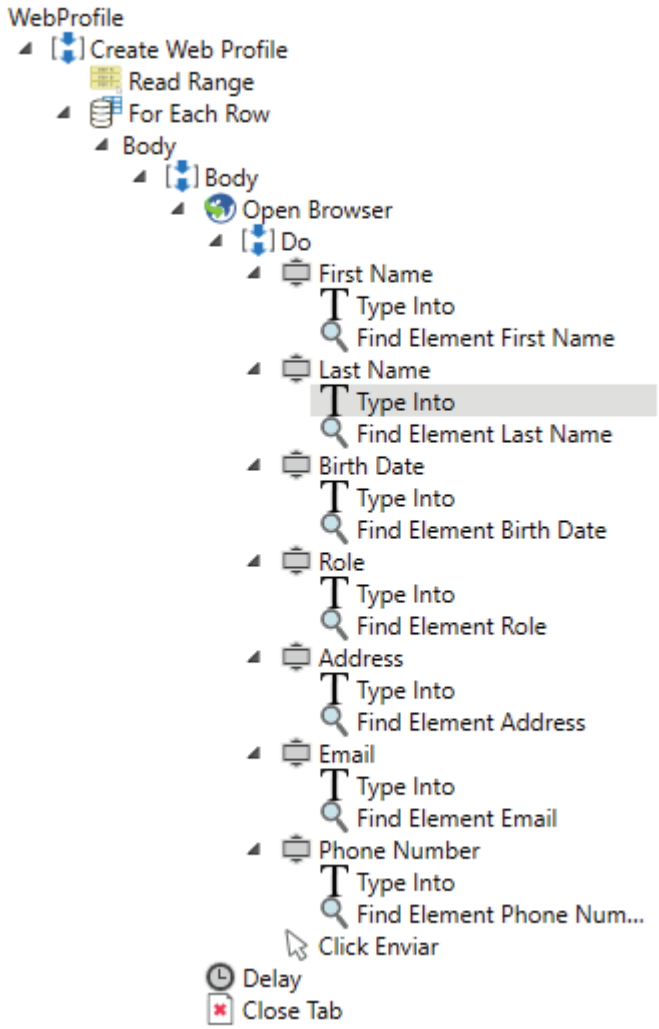

Imagen 3.10: Estructura de la secuencia WebProfile.xaml

- Variables:

Nombre	Tipo	Ámbito	Valor
URLWebProfile	String	Todo el programa	URL formulario web
OutputBrowser	Browser	Todo el programa	Sin inicializar
OutputDataBase	DataTable	Todo el programa	Sin inicializar

Imagen 3.11: Variables WebProfile.xaml

La variable *URLWebProfile* se utiliza para indicar la ruta del formulario web.
La variable *OutputBrowser* utiliza la librería *UiPath.Core.Browser*, es una variable en donde se almacena el objeto de tipo *Browser* que contiene la página abierta en el navegador.
La variable *OutPutDataBase* utiliza la referencia *System.Data.Table.DataTable*, se utiliza para almacenar el contenido de la base de datos.

- Métodos utilizados:
Read Range: extrae el contenido de la base de datos, dejándolo en una variable de tipo *DataTable*.

  - Entrada
    Excel base de datos
  - Salida
    La variable *OutputDataTable*, de tipo *DataTable*.

  For ech row: bucle en el que se recorren todas las entradas de la base de datos, por cada un de ellas abre un buscador introduciendo la dirección del formulario web, rellena los campos del formulario con el contenido correspondiente a cada usuario identificado en la base de datos y crea un perfil. Al final, cierra el buscador para dejarlo preparado para la siguiente iteración.

  - Concidición
    For each *row* in *OutputDataTable*
  - Cuerpo del bucle
    Open Browser: abre un buscador con la URL introducida.
    * Entrada
      Recibe una variable de tipo URL, contenida en la variable *URLWebProfile*.
    * Salida
      Devuelve una variable de tipo *Browser*, que se almacena en *OutputBrowser*.

    Anchor Base: se utiliza este método para realizar dos acciones a la vez. La primera acción es la búsqueda por imagen de contenido especificado en una página exterior a la aplicación. La segunda acción es la escritura de datos. De tal forma, en la página web, se busca el

campo del formulario correspondiente a cada columna de la base de datos y se escribe la información de cada usuario.
Anchor base utiliza dos métodos, *Find Element* y *Type Into* para llevar a cabo su desarrollo.

  * *Find Element:* método que busca por imagen. Recibe como entrada la imagen del campo que se quiere introducir en el formulario web.
  * *Type Into:* método que escribe en un selector de texto. Recibe como entrada el siguiente código: row("Nombre de la columna de la base de datos").ToString

Se utilizan siete métodos de *Anchor Base*, uno para cada campo del formulario (First name, last name, birth date, role, address, email y pone number).
Click: este método simula un click de ratón. La entrada que recibe es la imagen sobre la que pulsar para enviar los datos del formulario.
Delay: método para producir un retardo en la ejecución del robot. Es útil para esperar a que se envíen los datos antes de cerrar la página web.
Close Tab: método que cierra una página web. Recibe como argumento una variable de tipo *Browser*, se usa la variable *OutputBrowser* para cerrar la página web abierta en cada iteración.

- Manejo de excepciones:
  En esta secuencia, lo que se debe controlar es que se pueda abrir el navegador e introducir los datos de los perfiles que van a crearse. Se utiliza un bloque *Try-Catch*, en el que en la parte del *Try* se realiza todo el proceso desde la abertura del navegador hasta el cierre. Esto se repite en el bucle tantas veces como usuarios existan en la base de datos. Si por el motivo que fuera no se pudiera abrir el navegador o no se pudiera seguir procesando el rellenado de datos en el formulario, salta una excepción.

- Limpieza del sistema:
  No es necesario realizar ninguna acción de limpieza en este punto.

La segunda parte del robot, está constituida por las secuencias: PasswordsDB.xaml, CredentialDocument.xaml y SendEmail.xaml.

PasswordsDB.xaml

El resultado de la ejecución de esta secuencia es añadir en la base de datos una credencial para cada usuario. Primero, se leen de un documento las credenciales que han sido generadas de forma aleatoria constituidas por letras y números. Las credenciales se obtienen en una lista, se van leyendo de esa lista y se añaden en la celda correspondiente de la base de datos.

- Estructura:

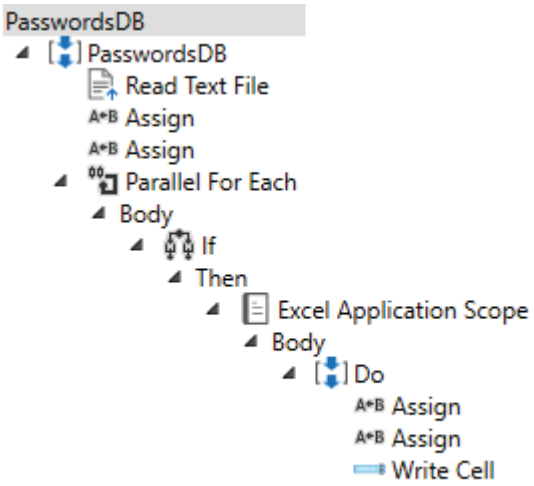

Imagen 3.12: Estructura de la secuencia PasswordsDB.xaml

- Variables:

Nombre	Tipo	Ámbito	Valor
OutputPasswords	String	Todo el programa	Sin inicializar
SplitPasswords	Array[String]	Todo el programa	Sin inicializar
Range	String	Todo el programa	Sin inicializar
CellNumber	Int32	Todo el programa	1

Imagen 3.13: Variables PasswordsDB.xaml

En la variable *OutputPasswords* es donde se almacena la información leída del documento que contiene las credenciales.
La variable *SplitPasswords* es un array de tipo string en donde se almacenan las credenciales que contiene la variable *OutputPasswords*, tras haber realizado la lectura del documento.
Las variables *Range* y *CellNumber* se utilizan para indicar en qué celdas de la base de datos deben almacenarse las credenciales.

- Métodos utilizados:
Read Text File: lee la información que contiene un documento y la almacena en una variable. Se utiliza para obtener las credenciales del documento.

- Entrada
  El documento que contiene las credenciales.
- Salida
  La información que contiene el documento en la variable *OutputPasswords*.

Assign: realiza una asignación en una variable. En este caso se eliminan los espacios del principio y final de la lectura del documento.

- Variable
  OutputPasswords
- Varlor
  OutputPasswords.Trim

Assign: realiza una asignación en una variable de tipo array. Se divide la variable de tipo string que contiene la información de las credenciales en las credenciales individualmente.

- Variable
  OutputPasswords
- Valor
  OutputPasswords.Split

Parallel For Each: este método es un bucle que recorre cada variable de tipo string de la variable *OutputPasswords*, leyendo cada credencial y asignándosela a cada persona en la base de datos.

- Condición
  For each *item* in *SplitPasswords*
- Cuerpo del bucle
  If: secuencia de escape para comprobar que estamos ante un *item* válido. Se comprueba que el *item* tenga longitud y por tanto, sea una credencial válida. La condición que se le aplica es: item.ToString.Length < 0
  Si se cumple se llama al método *Excel Application Scope*, que se utiliza para abrir el archivo base de datos. Dentro del cuerpo del método *Excel Application Scope* aparecen estos dos:
  Assign: se incrementa el valor de la variable *CellNumber* y se realiza una asignación posterior a la variable *Range*, para poder escribir en la celda adecuada según se vayan recorriendo los elementos en el bucle.
  Write Cell: método que escribe en las celdas de la base de datos. Las entradas serían las variables *Range*, para indicar dónde añadir información e *item*, que contiene la credencial actual del array que se está tratando en el bucle.

- Manejo de excepciones:
  Se engloba toda la secuencia en un bloque *Try-Catch*, de tal forma que si surge algún error en el procesado de las contraseñas o su paso a la base de datos se detenga la ejecución del robot.

- Limpieza del sistema:
  En este punto, no es necesario realizar ninguna acción.

CredentialDocument.xaml
Esta secuencia sirve para generar un nuevo documento por cada persona existente en la base de datos. El título del documento son el nombre y apellido, en su contenido se incluyen algunos datos personales y las credenciales de acceso al sistema de la organización, independientes para cada usuario.

- Estructura:

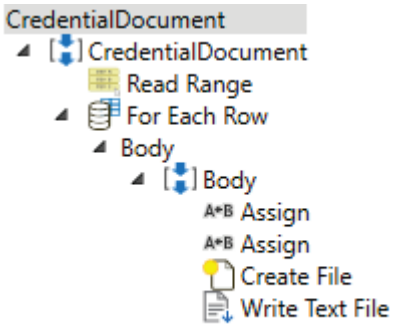

Imagen 3.14: Estructura de la secuencia CredentialDocument.xaml

- Variables:

Nombre	Tipo	Ámbito	Valor
OutputDataBase	DataTable	Todo el programa	Sin inicializar
PathCredentials	String	Cuerpo del bucle	Ruta de la carpeta
DocumentName	String	Cuerpo del bucle	Sin inicializar
Content	String	Cuerpo del bucle	Sin inicializar

Imagen 3.15: Variables CredentialDocument.xaml

La variable *OutPutDataBase* utiliza la referencia *System.Data.Table.DataTable*, se utiliza para almacenar el contenido de la base de datos.
En la variable *PathCredentials* se almacena la ruta de la carpeta donde se van a crear los documentos personalizados para cada persona.
En variable *DocumentName* se guarda el nombre del documento y en la variable *Content*, el contenido de este.

- Métodos utilizados:
  Read Range: método útil para leer la información de la base de datos y extraerla en una variable.

- Entrada
  Excel base de datos
- Salida
  La variable *OutputDataBase*, de tipo *DataTable*.

For each row: bucle para recorrer cada fila de la base de datos.

- Condición
  For each *row* in *OutputDataBase*.
- Cuerpo del bucle
  En cada iteración se asocian nuevos valores a las variables que se usan, y se crea un documento con el nombre y contenido especificados. Los métodos utilizados en el cuerpo del bucle se explican a continuación:
  Assign: asigna a la variable *DocumentName* el nombre deseado del documento. Código: row("First Name").ToString + row("Last Name").ToString + ".txt"
  Assign: asigna a la variable *Content* el contenido del documento. Código: "Dear " + row("First Name"). ToString + " " + row("Last Name").ToString + ":" +VbCrLf+ "Your credentials to access the organization are: " +VbCrLf+ "Email: " + row("Email").ToString + " and password: " + row("Credential"). ToString
  Create File: se utiliza este método para crear los documentos. La entrada del método son las variables *DocumentName* y *PathCredentials* para indicar el nombre del documento y la ruta de creación.
  Write Text File: este método escribe contenido en el documento seleccionado. Las entradas del método son, la variable *Content* para indicar el contenido y la ruta de ubicación del archivo, formada por las variables *PathCredentials* y *DocumentName*.

- Manejo de excepciones:
  Se engloba todo el proceso de la secuencia en un bloque *Try-Catch*, si se produce algún error durante la creación de los documentos con las credenciales, se producirá una excepción que detendrá la ejecución del robot.

- Limpieza del sistema:
  No es necesario realizar ninguna acción de limpieza en esta secuencia.

SendEmail.xaml
Esta secuencia envía un email a cada persona de la base de datos con un archivo adjunto, que es el documento generado en la secuencia *CredentialDocument.xaml*. Busca los emails en la base de datos y por cada uno de ellos, se envía el documento correspondiente a la persona asociada.

- Estructura:

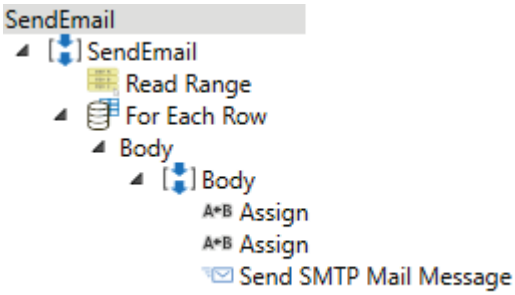

Imagen 3.16: Estructura de la secuencia SendEmail.xaml

- Variables:

Nombre	Tipo	Ámbito	Valor
OutputDataBase	DataTable	Todo el programa	Sin inicializar
EmailTo	String	Cuerpo del bucle	Sin inicializar
DocumentName	String	Cuerpo del bucle	Sin inicializar

Imagen 3.17: Variables SendEmail.xaml

La variable *OutPutDataBase* utiliza la referencia *System.Data.Table.DataTable*, se utiliza para almacenar el contenido de la base de datos.
Las variables *EmailTo* y *DocumentName* se utilizan en el cuerpo del bucle para almacenar la dirección de correo electrónico a la que enviar el email y el nombre del documento que se adjunta.

- Métodos utilizados:
Read Range: sirve para extraer la información que contiene la base de datos y dejarla almacenada en una variable de tipo *DataTable*.

  - Entrada
    Excel base de datos.
  - Salida
    La información resultante en la variable *OutputDataBase*.

  For Each Row: con este bucle se recorren las filas de la base de datos, extraídas en la variable *OutputDataBase*, para después operar.

  - Condición
    For each *row* in *OutputDataBase*

- Cuerpo del bucle
  Por cada fila recorrida, correspondiente a una persona, se extrae su correo electrónico y el nombre del documento que se tiene que adjuntar. Por último, se envía el mensaje. Los métodos empleados en el cuerpo del bucle son:
  Assign: se asigna a la variable *DocumentName* el nombre del archivo a adjuntar. Código: row("First Name").ToString + row("Last Name").ToString
  Assign: se asigna a la variable *EmailTo* la dirección de correo electrónico en la que realizar el envío. Código: row("Email").ToString
  Send SMTP Mail Message: este método realiza el envío de los correos electrónicos para cada usuario. Se tiene que configurar el servidor de correo e indicarle los siguientes parámetros: destinatario, asunto y cuerpo. En destinatario utilizamos la variable *EmailTo*, en asunto escribimos "Credentials" y el cuerpo se deja vacío. Finalmente, en la parte de archivos adjuntos, le idicamos la ruta donde localizar los archivos adjuntos, que esta formada por la ruta de la carpeta donde se almacenan los documentos de las credenciales más el nombre de cada archivo, guardado en la variable *DocumentName*.

En la siguiente imagen, se puede apreciar la configuración del método *Send SMTP Mail Message*:

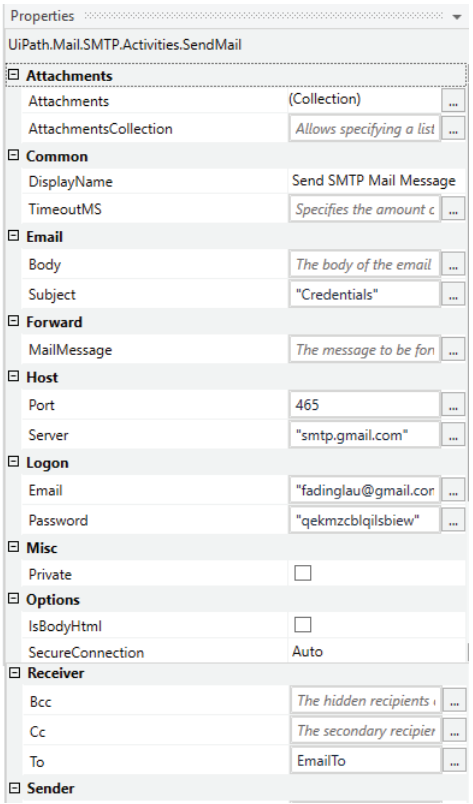

Imagen 3.18: Configuración del método Send SMTP Mail Message

Al igual que en la parte 1 de desarrollo del robot, en la secuencia *ReadEmail.xaml*, para iniciar sesión en el servidor de correo se introduce el email desde que se realizan las pruebas y una contraseña que no se corresponde con la contraseña real de la cuenta, sino una clave de acceso obtenida para trabajar con el servidor de correo desde la aplicación de UiPath.

- Manejo de excepciones:
  Al igual que en las secuencias anteriores de la segunda parte del robot. Se engloba toda la secuencia en un bloque *Try-Catch*, de tal forma que si en algún momento de la ejecución de esta parte, tanto en la lectura de los emails en la base de datos, como en el envío del email a cada usuario se produjera algún fallo, sería notificado mediante una excepción.

- Limpieza del sistema:
  Se procede a borrar todos los documentos con las credenciales de cada usuario, ya que han sido enviados y no son útiles después de que haya acabado el robot su ejecución. Después de adjuntarse el documento al enviar el correo electrónico, se borra el mismo.

Para finalizar el Robot 1, se ha creado una secuencia que pone en funcionamiento las anteriores de forma ordenada y secuencial. De tal manera que solamente con la ejecución de esta secuencia final, se ejecuta el Robot 1 en su totalidad.

Robot1.xaml
Secuencia final que constituye el Robot 1, invocando a las seis secuencias que lo constituyen.

- Estructura:

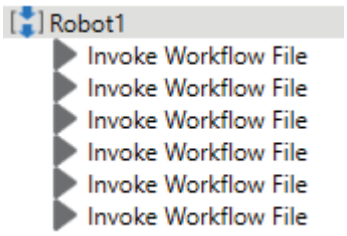

Imagen 3.19: Estructura de la secuencia Robot1.xaml

- Variables:
  No es necesario el uso de ninguna variable para desarrollar la secuencia *Robot1.xaml*.

- Métodos utilizados:
  Invoke Workflow File: método que invoca una secuencia ya existente. Se utiliza seis veces, cada una de ellas para invocar a cada una de las secuencias que componen al Robot 1, de forma ordenada. No recibe ningún

tipo de entrada o salida. Únicamente se ha hecho uso de una propiedad que permite establecer la continuación en la ejecución de la secuencia *Robot1.xaml* si se produjera fallo en la secuencia a la que invoca, esta propiedad se ha marcado como falso ya que no sería conveniente que el robot siguiera ejecutando las secuencias a continuación si se ha producido un error en la anterior, el cual se va a ir acarreando, además de que es un proceso secuencial en el que cada acción depende de la realizada anteriormente.
En la imagen a continuación se pueden ver las seis invocaciones a las secuencias del Robot 1, observándose la secuencia total *Robot1.xaml* desde UiPath.

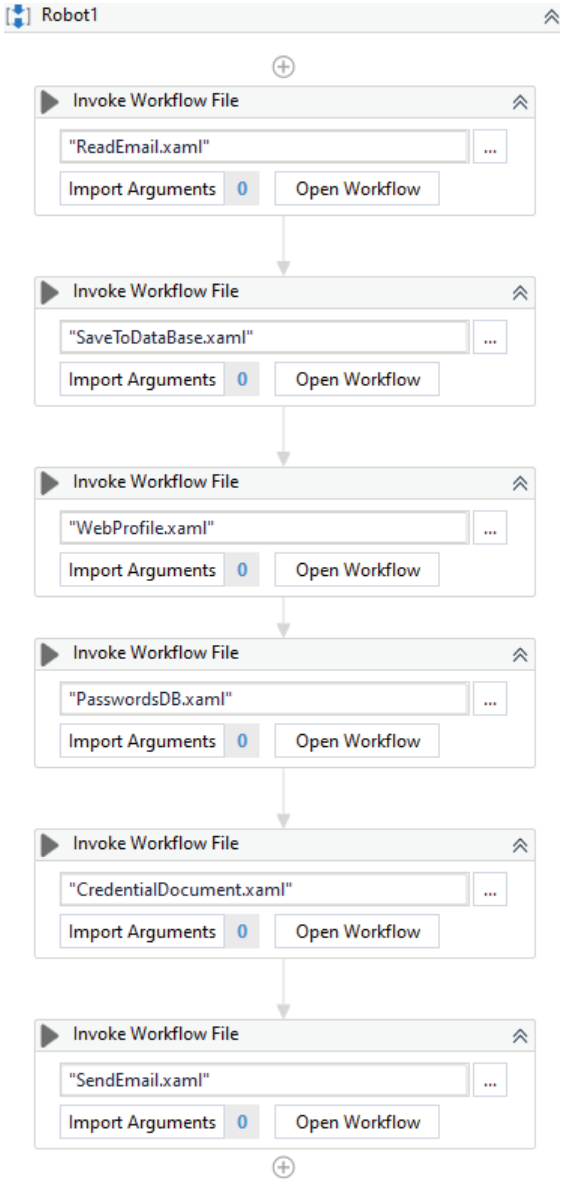

Imagen 3.20: Secuencia del Robot 1

- Manejo de excepciones:
  Únicamente puede fallar en el méodo *Invoke Workflow File*. Se engloba toda la secuencia del Robot 1 en un bloque Try-Catch, de tal forma que si en algún momento de la ejecución de esta secuencia, se produce un fallo al innovar a alguna de las demás secuencias, se notificaría mediante una excepción y un mensaje indicando que ha fallado la invocación. Se puede visualizar el bloque Try-Catch que engloba todos los métodos de la secuencia en la siguiente imagen.

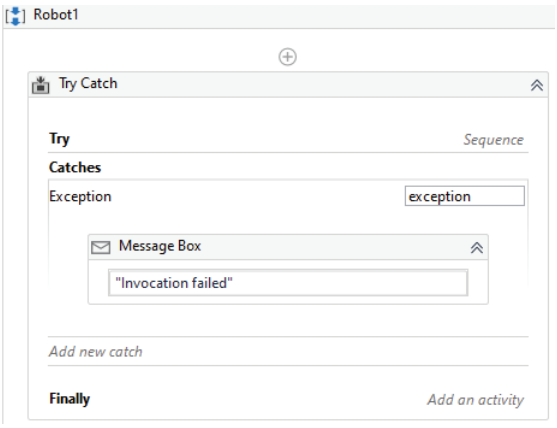

Imagen 3.21: Try-Catch en Robot1.xaml

- Limpieza del sistema:
  No es necesario realizar ninguna acción de limpieza puesto que ya se ha ido controlando la limpieza del sistema en las secuencias a las que invoca *Robot1.xaml*.

3.4.2 Caso de uso 2 - Robot 2

El Robot 2 se trata de un diagrama de flujo, en UiPath implementado mediante un *flowchart*. Los diagramas de flujo presentan operadores lógicos y ramificaciones que permiten seguir los procesos del robot por distintos caminos y conectar las actividades de múltiples maneras.
Este robot trata de representar el tratamiento de una alarma mediante una serie de preguntas, las respuestas de las preguntas llevarán a distintos resultados y ramas del diagrama, generando finalmente un reporte en el que se indica el proceso que se ha seguido que equivale a lo que ha representado esa alarma en la realidad.
En UiPath se ha creado un proyecto de tipo *flowchart* y en una única secuencia se ha representado todo el diagrama de flujo que constituye el proceso que sigue el Robot 2. El diagrama de flujo está constituido por tres entradas de diálogo y tres ramificaciones. Las entradas de diálogo representan una pregunta con dos posibles respuestas, según la opción que se seleccione la ramificación conducirá a una u otra rama del diagrama.
A continuación, se explicará más en detalle la estructura, las variables

necesarias, los cuadros de diálogo, las ramas de decisión y los métodos restantes que se han utilizado en el Robot 2.

- Estructura:

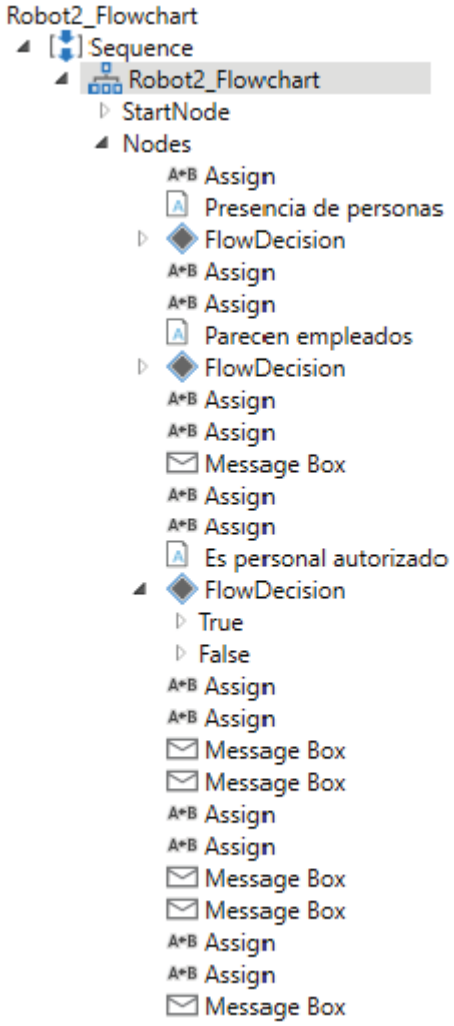

Imagen 3.22: Estructura del flowchart del Robot 2

- Variables:

Nombre	Tipo	Ámbito	Valor
Report	String	Todo el programa	Sin inicializar
Result	String	Todo el programa	Sin inicializar

Imagen 3.23: Variables flowchart Robot 2

La variable *Report* es donde se va almacenando la información que produce el ir avanzando en las ramas del diagrama de flujo para generar finalmente un texto con información legible sobre el tratamiento de la alarma.

La variable *Result* se utiliza para guardar el resultado seleccionado en los cuadros de diálogo y posteriormente evaluarlo en las ramas de decisión del *flowchart*.

- Métodos utilizados:
  Assign: este método modifica la variable *Report* durante toda la ejecución del *flowchart*. Recibe como entrada la variable *Report* y la salida que genera dependerá de la posición en el *flowchart* donde se encuentre, es decir, irá añadiendo información en esa variable según por los flujos de decisión que pase el robot y las decisiones tomadas.

  Input Dialog: este método muestra un cuadro de diálogo con opciones para seleccionar. En todo el *flowchart* se encuentran tres *Input Dialog*.

  - Entrada
    Recibe una etiqueta con la pregunta que se formula cuando aparece el cuadro de diálogo. Las tres etiquetas que aparecen en cada cuadro de diálogo son: "Se ha detectado presencia de personas" en el primer cuadro de diálogo, "¿Parecen empleados?" en el segundo cuadro de diálogo y en el último "¿Es personal autorizado?".
    También un array con la lista de opciones que se muestran en el cuadro de diálogo y que se pueden seleccionar. En los tres cuadros de diálogos se han elegido las mismas opciones, que son un array con los campos "Sí" y "No".
  - Salida
    El resultado seleccionado, que se guarda en la variable *Result* para usarse posteriormente.

  Flow Decision: método que evalúa la condición y conduce al robot por la rama que debe ir según el resultado de la evaluación. La condición es una expresión de tipo lógico, en este caso se comprueba si la salida del método anterior, *Input Dialog*, es afirmativa. La comprobación se realiza usando la variable *Result*. Si el resultado es correcto, entonces el robot seguirá su ejecución por la rama afirmativa y en caso contrario por la rama negativa.

  Message Box: este método se utiliza para mostrar una ventana con información cuando ha finalizado la ejecución del robot, esto es, al final de una rama del *flowchart*. Como entrada del método se utiliza la variable *Report* ya que esta variable habrá ido almacenando el seguimiento y tratamiento de la alarma en forma de texto, para ser visualizado al finalizar.

En este robot no se van a tratar excepciones ni limpieza del sistema. El tratamiento de excepciones no se considera necesario ya que es una automatización atendida, en la que un operador seguirá y monitorizará los pasos del robot. La limpieza del sistema no es necesaria tampoco en este robot porque no se utilizan otros medios fuera del robot que pudieran utilizar memoria o rendimiento del ordenador.

Una imagen del diagrama de flujo o *flowchart* del Robot 2 se presenta a continuación, donde se muestra la interfaz gráfica y los métodos utilizados desde UiPath.

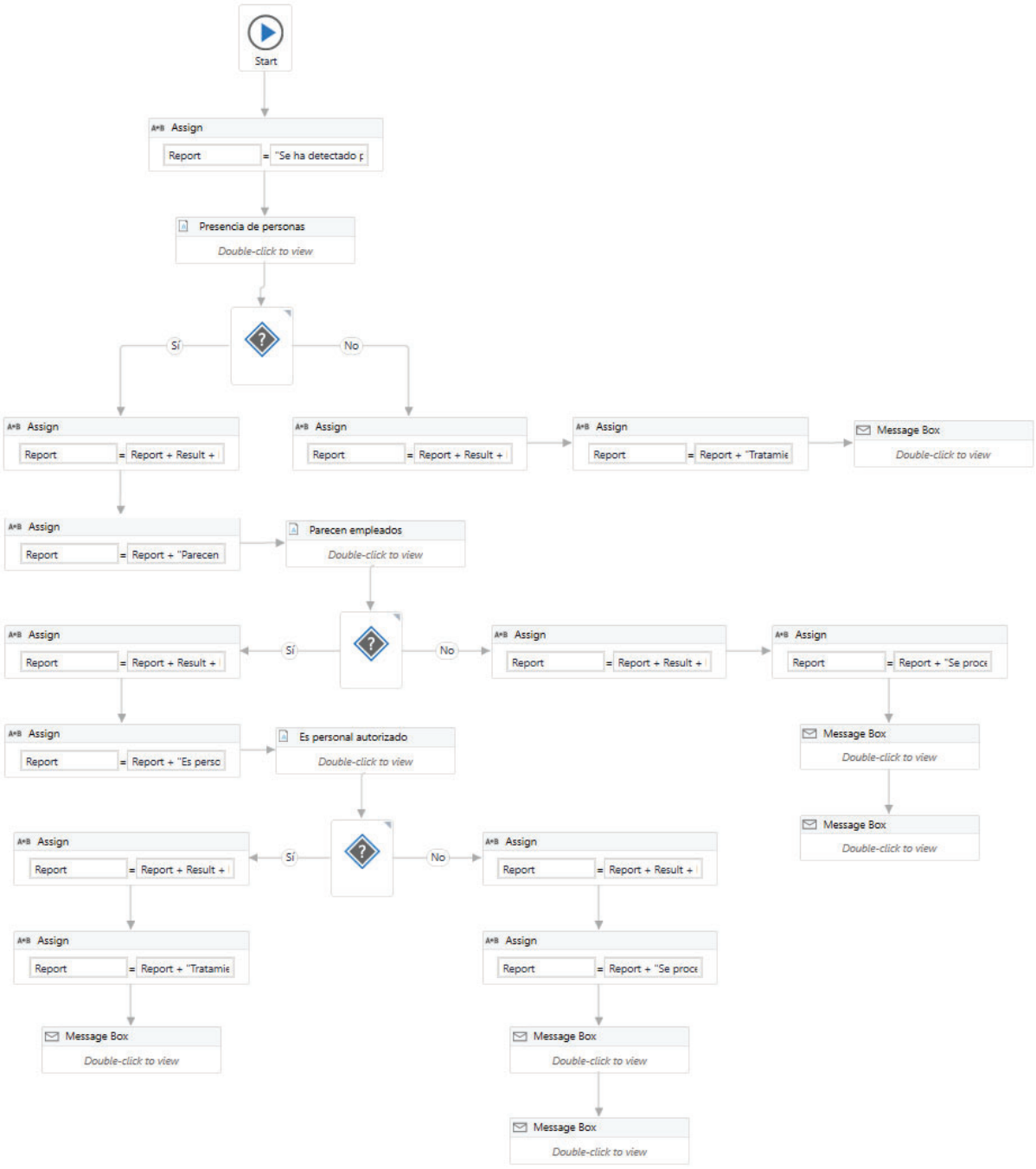

Imagen 3.24: Diagrama del Robot 2

3.5 Pruebas

En este apartado se van a explicar las pruebas que se han ido realizando mientras se estaban implementando los robots. Gracias a las pruebas, se pueden controlar aquellos comportamientos inusuales, fallidos o errores de programación que se hayan podido cometer y corregirlos antes de presentar una versión final de los robots. Se han elegido aquellos escenarios que podrían darse en la ejecución de los casos de uso, para así poder obtener robots más realistas adaptados a situaciones reales que pudieran surgir durante su funcionamiento.
Se diferencian dos tipos de pruebas, las unitarias y las del sistema.
Las pruebas unitarias se han ido realizando cada vez que se añadían métodos o extractos de código en el desarrollo de los robots, para poder continuar verificando que lo anterior ya estaba funcionando. Esto es, cada vez que se avanzaba en el desarrollo de una secuencia de los robots, se iba verificando que cumpliera los requisitos y finalmente funcionaba correctamente.
Las pruebas del sistema se refieren al funcionamiento del robot de forma global. Este tipo de pruebas se han realizado en los dos robots, en el Robot 1 cuando se unen las seis secuencias de las que está formado para dar lugar a el Robot 1 totalmente y en el Robot 2 comprobando que funciona correctamente todo su conjunto ya que únicamente está constituido por una secuencia.

3.5.1 Caso de uso 1 - Robot 1

ReadEmail.xaml
Se pueden encontrar o no emails en la bandeja de entrada con el asunto indicado. Si se encuentran emails, pueden contener o no archivos adjuntos. También se prueba que funcione correctamente el servidor de correo y la conexión.

- Prueba 1: no se encuentran emails con el asunto indicado en la bandeja de entrada. Como resultado, salta una excepción indicando que no se han encontrado emails.
- Prueba 2: se encuentran emails, se descargan los archivos adjuntos y se almacenan en la carpeta especificada.
- Prueba 3: se encuentran emails, pero no contienen archivos adjuntos. Entonces, no se almacena nada en la carpeta especificada quedando vacía.
- Prueba 4: no existe conexión con Internet o el servidor de correo electrónico está mal configurado. El robot falla en su ejecución al intentar leer los emails y se produce una excepción.

SaveToDataBase.xaml
Se escribe o no en la base de datos el contenido de los archivos adjuntos que haya en la carpeta donde se ha descargado la información que contenían los emails de la bandeja de entrada.

- Prueba 1: no hay ningún archivo en la carpeta donde se almacenan los archivos adjuntos de los emails, por lo tanto, no se escribe nada en la

base de datos. Salta una excepción indicando que no se han encontrado elementos para poder continuar con la secuencia.

- Prueba 2: se encuentran elementos en la carpeta, entonces se procede a leerlos y escribir los datos correspondientes en la base de datos.

WebProfile.xaml
Se crean perfiles web según la ocupación de la base de datos. Si está vacía, no se realiza ninguna acción. La parte más importante de esta secuencia es el navegador, que funciona mediante conexión a Internet, puede funcionar correctamente o fallar debido a la conexión.

- Prueba 1: la base de datos está vacía ya que no se han recibido emails o éstos no contenían archivos adjuntos. En este caso, el robot no realiza ninguna acción puesto que no hay información que procesar.
- Prueba 2: la base de datos contiene información y existe conexión con Internet. El robot lee cada fila de la base de datos y crea un perfil web para cada persona. Continúa la ejecución de la secuencia como debería.
- Prueba 3: la base de datos contiene información, pero falla la conexión con Internet. El robot lee la información de la base de datos y trata de abrir un navegador web, no puede realizarlo ya que no hay conexión o trata de enviar los datos una vez rellenado el formulario, pero se le impide por el mismo motivo. En este caso salta una excepción impidiendo que siga la ejecución del robot hasta que reaparezca la conexión.

PasswordsDB.xaml
Genera una credencial por cada usuario existente. La ejecución de la secuencia dependerá de la ocupación de la base de datos.

- Prueba 1: no hay usuarios en la base de datos. El robot termina su ejecución inmediatamente.
- Prueba 2: hay usuarios en la base de datos. El robot continúa su ejecución introduciendo una credencial por cada usuario.

CredentialDocument.xaml
Genera un documento por cada usuario. La ejecución de la secuencia dependerá de la ocupación de la base de datos principalmente.

- Prueba 1: no hay usuarios en la base de datos. El robot termina su ejecución inmediatamente.
- Prueba 2: hay usuarios en la base de datos. El robot continúa su ejecución creando un documento por cada persona. Puede producir fallo si faltan campos en la base de datos necesarios para la creación del documento, tales como el nombre y la credencial asignada. Como resultado a fallos en la ejecución, se producirá una excepción.

SendEmail.xaml
Se envía un correo con el documento asociado a cada persona a cada usuario existente en la base de datos. Puede haber usuarios o no, entonces la secuencia se ejecutará tantas veces como correos haya que mandar. También hay que revisar la conexión con Internet y la configuración del servidor de correo electrónico elegido.
No hace falta comprobar que las direcciones de correo electrónico a las que enviar el email resultante existan, puesto que son las mismas de las que se ha recibido el correo electrónico en la primera secuencia del robot.

- Prueba 1: no existen usuarios en la base de datos. El robot termina su ejecución inmediatamente.
- Prueba 2: existen usuarios en la base de datos y conexión con Internet o el servidor de correo está bien configurado. El robot procede a realizar el envío. El robot puede dar fallo si no se encuentran los archivos adjuntos asociados a cada perfil o si al realizar alguna lectura en la base de datos se produjera error no encontrando algún campo que esté en blanco. Como resultado, saltaría una excepción que interrumpiría la ejecución del robot.
- Prueba 3: existen usuarios en la base de datos, pero no funciona la conexión con Internet o la configuración del servidor de correo electrónico no es correcta. Se produce una excepción al abrir la parte de servidor de correo electrónico, interrumpiendo la ejecución del robot.

Robot1.xaml
En esta secuencia se realiza una prueba de sistema, en la que se mide que la ejecución de las seis secuencias anteriores en su conjunto funcione correctamente, ya que su ejecución secuencial es lo que determina el comportamiento del Robot 1. Esta prueba puede producir fallo si falla alguna de las secuencias de las que está compuesta, en otro caso el Robot 1 funciona correctamente de forma desatendida.

3.5.2 Caso de uso 2 - Robot 2

El Robot 2 se pone en ejecución cuando salta una alarma que detecta presencia de personas en la organización, y se debe comprobar si esas personas tienen acceso permitido o no para dejarles continuar. El Robot 2 tiene cuatro posibles casos de ejecución distintos. Cada uno de ellos está representado en una prueba. Se pueden encontrar personas o puede que sea una falsa alarma que ha saltado, pero no ha detectado personas realmente. En el caso de que se hayan detectado personas, pueden ser empleados o no, o puede ser personal autorizado o no autorizado. Dependiendo de qué opción se trate, el Robot 2 seguirá un flujo u otro. Siempre se trata de una automatización atendida puesto que una vez se ha iniciado el robot, el operador debe instruirle hasta que finalice su ejecución.

- Prueba 1: salta una alarma y no se ha detectado presencia de personas. Se genera el reporte y el tratamiento de la alarma finaliza inmediatamente. Lo que debería visualizarse al final del tratamiento de la alarma es la siguiente imagen, donde se muestra el texto generado por el reporte.

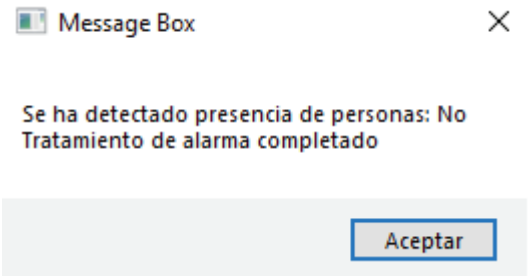

Imagen 3.25: Reporte de la prueba 1 del Robot 2

- Prueba 2: salta una alarma en la que se ha detectado presencia de personas y no parecen empleados. Una vez seguido el flujo del *flowchart* por el operador, se anuncia que se debe hacer la llamada de alerta ya que las personas detectadas no son identificadas. Por último, se genera reporte y finaliza el tratamiento de la alarma. En la imagen se visualiza el texto del reporte de esta prueba, que explica el tratamiento de la alarma que ha seguido el operador.

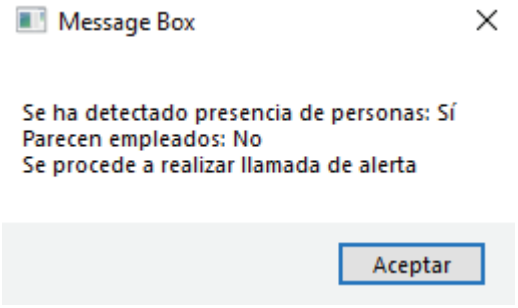

Imagen 3.26: Reporte de la prueba 2 del Robot 2

- Prueba 3: salta una alarma en la que se ha detectado presencia de personas y parecen empleados. Las personas identificadas sí que son personal autorizado. El operador sigue el flujo del *flowchart* respondiendo a las preguntas según se den los hechos hasta que el tratamiento de la alarma finaliza y se genera el reporte. En la imagen a continuación, se observa el texto del reporte indicando las acciones que ha seguido el operador hasta la finalización de la alarma.

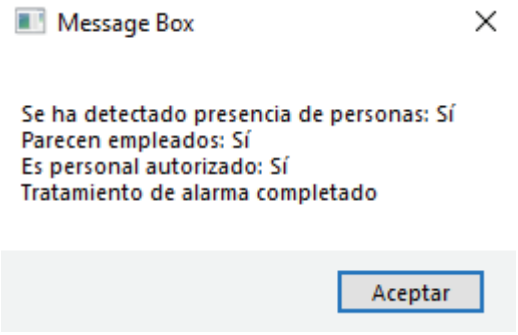

Imagen 3.27: Reporte de la prueba 3 del Robot 2

- Prueba 4: salta una alarma en la que se ha detectado presencia de personas y parecen empleados. Las personas que se han identificado no corresponden a personal autorizado. El operador realiza el tratamiento de la alarma hasta su finalización. Por último, en esta prueba se notifica que se debe realizar la llamada de alerta al encontrarse con personas intrusas y no identificadas en la organización. Se termina el procesamiento de la alarma y se genera el reporte correspondiente. En la imagen siguiente se visualiza el resultado del reporte con las acciones realizadas.

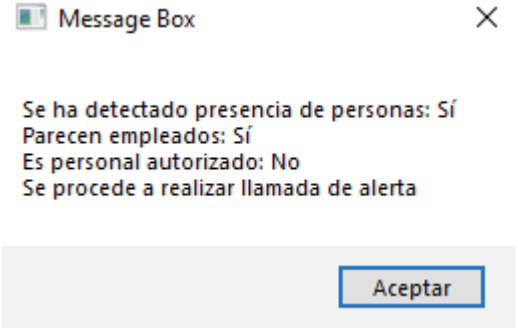

Imagen 3.28: Reporte de la prueba 4 del Robot 2

En los dos apartados de pruebas anteriores, del Robot 1 y del Robot 2 se han explicado las pruebas que se han realizado para finalizar con el desarrollo de los robots y comprobar que verifican los requisitos del sistema. En ambos casos, los objetivos han sido alcanzados puesto que lo mencionado anteriormente en los casos de prueba lo cumplen los dos robots.

3.6 Orquestación y usos

El orquestador de UiPath sirve para controlar a los dos robots de software que se han creado desde una única aplicación. Desde el apartado orquestador de UiPath se pueden controlar, gestionar y monitorizar la ejecución de los robots, se pueden programar colas de trabajo y todo tipo de procesos en los que participen los robots. Además, todas las ejecuciones y actividades de los robots se quedan

registradas en el orquestador, lo cual es muy útil para tener copias de seguridad y un seguimiento de las acciones de los robots en cualquier proceso.
El uso que podría darse a los dos robots creados dentro de la organización elegida de sistemas de seguridad sería que fueran partícipes dentro de otros procesos o aplicaciones de software ya existentes en dicha organización, viéndose de esta manera a los robots como una parte de una gran aplicación o desarrollo software.
Ambos robots están formados por archivos ".xaml" modificados desde UiPath, lo que se ha realizado es convertir las secuencias que forman los robots, en una librería de clases, simplemente haciendo una pequeña transformación y cambiando su extensión. De esta forma se pueden utilizar los dos robots como dos librerías de clases, y se pueden utilizar sus componentes dentro de otros proyectos de automatizaciones o aplicaciones que se creen y con las cuales sean compatibles.
Centrándonos mas en los dos casos de uso, suponemos que tenemos dos proyectos de software en los que funcionan más componentes a parte de los robots desarrollados, las librerías de los robots formarían parte de estos proyectos y los robots se ejecutarían cuando se les indicase desde el proyecto de cada caso de uso.
Esto no implica que se tengan dos robots automatizados, independientemente de formar parte de proyectos mayores, que se puedan poner en funcionamiento siempre y cuando un operador lo desee. Son monitorizados desde el orquestador y su rastro será registrado.

Capítulo 4

Resultados y conclusiones

Se han conseguido los dos objetivos que este trabajo tiene principalmente. Por una parte, el marco teórico sobre RPA en relación con su aplicación en las empresas y el segundo objetivo la creación de robots de software aplicados para mejorar la productividad en una organización.
En el estudio de la aplicación de RPA en una organización y las soluciones que aporta, al principio no pensé que encontraría tanta información al ser una tecnología emergente y que se está empezando a implantar en empresas.
Estaban establecidas unas secciones sobre las que buscar información para añadir a este trabajo, pero con la búsqueda de contenidos para el capítulo de *Análisis del estado del arte*, se fueron encontrando nuevas fuentes y avances sobre RPA que han servido para añadir nuevos apartados al capítulo y así aumentar el volumen de contenidos. No obstante, aunque no se ha podido recurrir a libros o artículos, toda la información obtenida ha sido gracias a páginas en Internet y las propias páginas web de softwares RPA.
El desarrollo de los robots de software con UiPath y el estudio de su comportamiento y capacidades de trabajo ha sido el bloque que ha llevado más tiempo de dedicación, pero, sin embargo, no considero que haya habido dificultades. Se estableció un plan de trabajo en el que para cada robot se tuviera que elaborar un diseño sobre cómo desarrollarlo, después estructurar y dividir en bloques la sección anterior, manejar las excepciones y la limpieza del sistema.
Durante la fase de implementación y pruebas las dificultades encontradas se han solventado rápidamente recurriendo a la documentación de UiPath. Es cierto que la idea que se tenía en un principio sobre el Robot 2 no ha sido la que finalmente se ha desarrollado, esto es debido a que esa idea sería conveniente resolverla con otras herramientas y no mediante la automatización de procesos. Por ese motivo, se cambió la estructura y se decidió realizar un *flowchart*, al menos para utilizar otra funcionalidad de UiPath y dar una solución al segundo caso de uso que tuviera utilidad dentro de una organización.

Como consecuencia de lo expuesto durante todo el trabajo, se puede afirmar que la automatización de procesos está cobrando cada vez más relevancia en la transformación digital de las compañías de todos los tamaños. En principio, RPA lo adoptan empresas de grandes tamaños porque gestionan un

mayor volumen de procesos y obtienen más beneficios con la automatización. No obstante, esta tecnología también está siendo adoptada por empresas de menores tamaño debido a todo lo que les aporta.
RPA agrupa tecnologías de software basadas en Inteligencia Artificial, junto con herramientas tradicionales basadas en reglas para aplicarlo a robots de software. RPA aporta soluciones para eliminar aquellos procesos repetitivos de una organización, con el propósito de que los empleados sustituyan ese tiempo por tareas que aporten más valor a la organización.
Es corriente pensar, que la automatización eliminará puestos de trabajo al sustituir directamente el trabajo de una persona. Sin embargo, esto es una creencia falsa, ya que RPA creará más puestos de trabajo que los que destruirá. Es cierto que ya no serían necesarias las personas en una organización que realizan las mismas tareas que los robots de software, pero aún así, estos robots tienen que ser monitoreados y controlados. Los robots realizan el trabajo tedioso y aparatoso pero las personas serán quienes tomen las decisiones correctas cuando los robots hayan finalizado su ejecución. Todavía es necesaria esa dependencia de las personas en la toma de decisiones, aunque los robots vayan aprendiendo de estas decisiones. Resumiendo, se puede afirmar que los robots de software transforman los puestos de trabajo, pero no los destruyen.
Lo que se espera de RPA es, por un lado, el crecimiento económico a la vez que una transformación digital. Por otro lado, una reducción de la tasa de errores prácticamente a cero.
El movimiento de la automatización de procesos está teniendo mucho éxito actualmente en industrias dedicadas a servicios financieros, telecomunicaciones, sanidad, etc. RPA proporciona muchos beneficios en las industrias y hace que aumenten sus ganancias. Muchas soluciones de RPA sirven para no cometer errores y así reducir costes, mientras que otras para abordar gran cantidad de tareas y ahorrar tiempo. En conclusión, para proporcionar un crecimiento en la organización donde se apliquen.
En muchas inversiones tecnológicas, los resultados son efectivos, pero se puede tardar en recuperar la inversión meses. Esto no ocurre con RPA, la automatización es increíblemente rentable y se puede empezar a recuperar lo invertido en tan sólo semanas. Usando RPA, el trabajo se realizará rápidamente. Esto implica que ese ahorro de tiempo permita realizar más trabajo, por lo tanto, mejorará la productividad y se reducirán los gastos a su vez. Visto desde una perspectiva financiera, es un avance para el crecimiento de cualquier industria.
En todo grupo de trabajo se cometen errores. Sin embargo, el coste de los errores decaerá utilizando las técnicas de RPA. Una persona puede cometer errores en su trabajo, y estos errores tienen su coste. No es que sea una gran pérdida para una organización, pero ya simplemente teniendo que reparar los errores, se invierte tiempo extra. Y una inversión de tiempo en algo que no era necesario, también supone invertir más dinero. Además, los errores también reducen la satisfacción de las personas que se benefician de lo que proporciona la organización, esto implica que se vaya perdiendo su reputación, por lo que no es nada beneficioso.
Sustituyendo la ejecución de los procesos ahora por robots, no se producirán errores. Realmente sí podrían producirse errores, porque son máquinas y

pueden fallar igualmente como las personas, pero no de la misma forma. Los robots están programados ante las excepciones y ejecuciones anómalas que puedan darse, minimizan los impactos negativos que puedan surgir.
En conclusión, el crecimiento en la automatización de procesos es una forma de abordar las necesidades de los clientes más exigentes. Lo que sucede es que como sus necesidades se cumplen, y además rápidamente y sin errores, es muy probable que los clientes se vuelvan aún más exigentes. Solicitarán tiempos de respuesta más rápidos y soluciones más sofisticadas a la vez que se volverán menos tolerantes con los fallos.
Con el auge de la Inteligencia Artificial y el uso de RPA, las empresas encontrarán nuevas formas de satisfacer las necesidades de los clientes. La mejora de la satisfacción de los clientes es sólo una parte en la que es bueno el uso de RPA, pero como es lo que mantiene a cualquier organización, es un objetivo potencial. Esto no quiere decir que RPA sea una herramienta tecnológica puramente comercial, ya que otro de los objetivos potenciales del uso de RPA es la transformación digital.
Según avanza la tecnología, se van creando nuevas expectativas que alguien tiene que cumplirlas. No en un futuro lejano, los robots de software se convertirán en una parte inferior de la infraestructura de cualquier tipo de compañía. Las compañías cambian y se transforman en función de las necesidades y las nuevas herramientas que proporcione la tecnología si son beneficiosas para ellas.
Automatizar decisiones es el uso futuro más atractivo para RPA, utilizando una Inteligencia Artificial avanzada. Machine Learning abre las puertas a RPA para poder automatizar las decisiones y que se produzcan en tiempo real gracias a lo que haya aprendido el robot durante la implementación. Unido a este tema, hay que recordar siempre que RPA no es Inteligencia Artificial, sino que las técnicas de RPA incluyen elementos de Inteligencia Artificial para resolver problemas de automatización. Fundamentalmente, se usa el reconocimiento de imágenes y el tratamiento del lenguaje natural. No se pretende usar una inteligencia avanzada para tomar decisiones complejas ni crear robots inteligentes, sino analizar y ejecutar correctamente acciones y manejar información desestructurada dentro de pantallas para solventar los procesos de una organización.

4.1 Líneas futuras

En cuanto a líneas futuras, no se pretende dar continuidad a este trabajo. Se podrían mejorar los robots creados aumentando sus funciones y usos. La parte de este trabajo que sí podría mejorarse sería el orquestador, creando colas de trabajo con los robots o programando las ejecuciones de los robots de alguna manera de las que se les pueda dar algún uso más importante del que ya tiene. De todas formas, está la opción de la conversión de los robots en librería de clases y poder utilizar estas librerías en cualquier programa de software que se quiera desarrollar en un futuro o formar parte como componente de aplicaciones.

Bibliografía

[1] UiPath. (2019). "Robotic Process Automation (RPA)". [Online]. Available: https://www.uipath.com/rpa/robotic-process-automation.

[2] Gavilán, I. G. R. (2018). "TECNOLOGÍA PARA LA DIGITALIZACIÓN DE PROCESOS (III). ROBOTIC PROCESS AUTOMATION (RPA)". [Online]. Available: http://www.reingenieriadigital.es/tecnologia-para-la-digitalizacion-de-procesos-iii-robotic-process-automation-rpa/.

[3] pfstech (2019). "Elegir la herramienta de RPA adecuada". Digitalbiz. [Online]. Available: https://www.digitalbizmagazine.com/elegir-la-herramienta-de-rpa-adecuada/.

[4] pfstech (2019). "Seguridad en proyectos de RPA". Digitalbiz. [Online]. Available: https://www.digitalbizmagazine.com/seguridad-en-proyectos-de-rpa/

[5] pfstech (2019). "Extracción automatizada del dato".Digitalbiz. [Online]. Available: https://www.digitalbizmagazine.com/extraccion-automatizada-del-dato/.

[6] Gavilán, I. G. R. (2018). "Cinco buenas prácticas para implantar RPA (Robotic Process Automation)". [Online]. Available: http://ignaciogavilan.com/cinco-buenas-practicas-para-implantar-rpa-robotic-process-automation/.

Este documento esta firmado por

Firmante	CN=tfgm.fi.upm.es, OU=CCFI, O=Facultad de Informatica - UPM, C=ES
Fecha/Hora	Thu Jan 09 20:06:09 CET 2020
Emisor del Certificado	EMAILADDRESS=camanager@fi.upm.es, CN=CA Facultad de Informatica, O=Facultad de Informatica - UPM, C=ES
Numero de Serie	630
Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)