



IMPLEMENTACIÓN DE UNA SOLUCIÓN MODULAR DE MIGRACIÓN DE DATOS DE USUARIOS

Rubén Cruz Macias
Grado de Ingeniería Informática
Gestión de Proyectos

Consultor: Xavier Martínez Munné
Profesor: Atanasi Daradoumis Haralabus

04 de Enero de 2019



Esta obra está sujeta a una licencia de Reconocimiento-NoComercial-SinObraDerivada [3.0 España de Creative Commons](https://creativecommons.org/licenses/by-nc-nd/3.0/es/)

FICHA DEL TRABAJO FINAL

Título del trabajo:	<i>IMPLEMENTACIÓN DE UNA SOLUCIÓN MODULAR DE MIGRACIÓN DE DATOS DE USUARIOS</i>
Nombre del autor:	<i>Rubén Cruz Macias</i>
Nombre del consultor/a:	<i>Xavier Martínez Munné</i>
Nombre del PRA:	Atanasi Daradoumis Haralabus
Fecha de entrega (mm/aaaa):	01/2019
Titulación:	<i>Grado de Ingeniería Informática</i>
Área del Trabajo Final:	<i>Gestión de proyectos</i>
Idioma del trabajo:	<i>Castellano</i>
Palabras clave	<i>Migración, automatismo, modular</i>

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados y conclusiones del trabajo.*

La transformación digital de las organizaciones es un hecho real que está a la orden del día. Para que ésta sea lo menos invasiva de cara al usuario final, se requieren de potentes herramientas que contemplen todas las necesidades del cliente y que no pierdan la confiabilidad en el proceso de implantación.

Es por ello que, con este trabajo final de grado, se pretende abordar este punto conflictivo que es la migración de los datos de usuario (llamados *settings*, en referencia a las propiedades del usuario), creando un sistema que encapsula y encripta de forma segura la información de cada usuario mediante scripting, siendo el mismo usuario quien escoja qué exportar y qué importar nuevamente a su máquina.

La metodología utilizada ha sido método *agile*, permitiendo que, con cada iteración del proceso, se pueda comprobar el avance de la solución y modificarla en caso necesario.

El resultado será una solución modular (cada módulo es independiente entre ellos), aplicable a cualquier escenario (según dominios), personalizable (permite al cliente crear nuevos módulos) y asequible (no requiere de licenciamiento).

En definitiva, este sistema permite que el proceso de migración del usuario se simplifique y no precise de asistencia manual durante su desarrollo (todo basado en sistema Windows).

Abstract (in English, 250 words or less):

The digital transformation of organizations is a real fact that is the order of the day. To be the least invasive for the final user, powerful tools are required to meet all customer needs and not lose reliability in the implementation process.

That is why, with this final degree work, it is intended to address this conflicting point that is the migration of user data (called settings, in reference to the user's properties), creating a system that securely encapsulates and encrypts each user's information using scripting, being the same user who chooses what to export and what to import back into their machine.

The methodology used has been *agile*, allowing, with each iteration of the process, you can check the progress of the solution and modify it if necessary.

The result will be a modular solution (each module is independent between them), applicable to any scenario (according to domains), customizable (allows the client to create new modules) and affordable (does not require Licensing)

In short, this system allows the user's migration process to be simplified and does not require manual support during its development (all based on Windows system environments).

AGRADECIMIENTOS

Tengo que agradecer – y mucho – a mi futura esposa, Eva M.^a García Castro, por todo su apoyo y paciencia durante estos cinco años. Han sido muchas las horas dedicadas y siempre he contado con su apoyo y ayuda, en ningún momento una mala cara por tener que dedicarle horas al estudio y han sido muchísimos fines de semana en los que no ha podido realizar planes por mi “culpa”. Así que, seguramente sin su apoyo y su paciencia no hubiera llegado a este momento. Solo puedo decir “Gracias y te quiero”.

Como no, también a la familia, los amigos y compañeros de trabajo, ya que en alguna ocasión también he fallado en alguna fecha señalada y en todo momento lo han comprendido.

Y por último, pero no menos importante, agradecer a todo el grupo docente de la UOC que entiende y valora la dificultad que supone compaginar trabajo y estudios hoy en día.

Índice

1. Introducción	1
1.1 Contexto y justificación del Trabajo	1
1.2 Objetivos del Trabajo	3
1.3 Enfoque y método seguido	5
1.4 Planificación del Trabajo	5
1.5 Breve resumen de productos obtenidos	9
1.6 Breve descripción de los otros capítulos de la memoria ...	9
2. Diseño e implementación de la solución	10
2.1 Entorno Laboratorio	10
2.2 Diseño y desarrollo de Módulos exportación de datos	11
2.2.1 Módulo Favoritos Navegadores.....	12
2.2.2 Módulo Configuración de correo Outlook	16
2.2.3 Módulo Configuraciones Office.....	18
2.2.4 Módulo Unidades de Red	20
2.2.5 Módulo Certificados.....	21
2.2.6 Módulo SAP GUI	22
2.2.7 Módulo ODBC.....	24
2.2.8 Módulo Conexiones SNA	25
2.3 Diseño y desarrollo del HTA de exportación de datos	27
2.3.1 Identificación Usuario.....	28
2.3.2 Gestión de los datos.....	29
2.3.3 Proceso del HTA	29
2.4 Diseño y desarrollo del HTA de importación de datos	34
2.4.1 Identificación usuario.....	35
2.4.2 Gestión de datos.....	36
2.4.3 Proceso del HTA	36
2.5 Diseño y desarrollo de Módulos importación de datos	39
2.5.1 Módulo Favoritos Navegadores.....	39
2.5.2 Módulo Configuración de correo Outlook	41
2.5.3 Módulo Configuraciones Office.....	46
2.5.4 Módulo Unidades de Red	48
2.5.5 Módulo Certificados.....	48
2.5.6 Módulo SAP GUI	49
2.5.7 Módulo ODBC.....	52
2.5.8 Módulo Conexiones SNA	52
2.6 Validación de la solución	53
3. Conclusiones	55
4. Glosario	57
5. Bibliografía	59
6. Anexos	61
6.1 Estructura del share	61
6.2 Script para encriptar .vbs	61

Lista de figuras

Figura 1. Análisis de costes.....	3
Figura 2. Diagrama Gantt I.....	6
Figura 3. Diagrama Gantt II.....	7
Figura 4. Diagrama Gantt III.....	7
Figura 5. Diagrama Gantt IV.....	8
Figura 6. Versiones Windows más usadas	10
Figura 7. Máquinas virtuales	11
Figura 8. Fichero profiles.ini	14
Figura 9. Equivalencias sistema operativo/versión.....	15
Figura 10. Ejemplo contacto “cacheado” en Outlook.....	16
Figura 11. Equivalencia Office/Versión	16
Figura 12. Opciones configuración MS Office	19
Figura 13. Unidades de red mapeadas	20
Figura 14. Administrador de ODBC.....	25
Figura 15. Aplicativo bajo conexión SNA	26
Figura 16. HTA Exportación	27
Figura 17. Error contraseña I.....	28
Figura 18. Error contraseña II.....	28
Figura 19. Error Selección.....	29
Figura 20. Inicio proceso	30
Figura 21. Error Microsoft Edge	30
Figura 22. Exportación realizada.....	30
Figura 23. Correo a usuario con su ID	31
Figura 24. Contenido envíoCorreo.vbe.....	31
Figura 25. HTA Importación	35
Figura 26. Error ID.....	35
Figura 27. Error contraseña III.....	35
Figura 28. Error contraseña IV	36
Figura 29. Inicio proceso II	36
Figura 30. Error Microsoft Edge II	37
Figura 31. Importación realizada	37
Figura 32. Apertura Outlook	42
Figura 33. Cuenta correo usuario.....	43
Figura 34. Contraseña correo usuario.....	43
Figura 35. Ejemplo Log Exportación	53
Figura 36. Ejemplo Log Importación.....	54

1. Introducción

La temática de este Trabajo Final de Grado es la creación de una solución automatizada de traspaso de datos a nivel de configuración de usuario, enmarcada en la migración de usuarios asociado a un renove tecnológico de todo tipo de organizaciones.

En concreto, se pretende conseguir un automatismo que ayude a configurar de una manera rápida y segura el puesto de trabajo de los usuarios que se plantee migrar a nuevas plataformas tecnológicas y desde cualquier entorno, incidiendo en la diversidad de escenarios de origen (organizaciones sin dominio, cambios de dominio, renovación *workstation* por obsolescencia, etc), dentro de la estrategia de negocio de cada organización, y manteniendo la homogeneidad del sistema.

Este trabajo no sólo se limitará a formular dichos automatismos necesarios en cada migración, sino que además habrá de tener en cuenta el cumplimiento de la Ley Orgánica de Protección de Datos en cuanto a tratamiento; así como atajando los diferentes problemas que vayan surgiendo durante el desarrollo de éste.

En la implementación de esta solución, no se contemplará el procedimiento de cambio de escenario a nivel hardware o plataforma, si no que se centrará en el desarrollo de los diferentes scriptings o aplicaciones que guiarán el proceso de configuración básica a cada puesto de trabajo (unidades de red, configuración de correo, certificados...).

Es decir, se propone una solución con garantía de calidad que permita tener una estrategia a medio y largo plazo coherente y de futuro para posibles clientes en sus cambios tecnológicos.

1.1 Contexto y justificación del Trabajo

Actualmente, son cada vez más las empresas que emprenden la odisea de renovarse tecnológicamente sin perderse por el camino, invirtiendo ingentes cantidades de recursos que finalmente no satisfacen las necesidades de todos los *stakeholders* implicados en el proyecto.

Sustituir una infraestructura tecnológica parece, a primera vista, un hito conseguible y asequible, pero a medida que van surgiendo incidencias o problemas, aparece el paradigma que constituye una inversión financiera cuantiosa, así como una desoptimización de los recursos dedicados y que no se habían tenido en cuenta a la hora de planificar el proyecto.

Uno de los puntos conflictivos en la migración de usuarios es el de asegurar que el usuario final dispondrá de todas las funcionalidades básicas que había estado utilizando hasta el momento (teniendo en cuenta las configuraciones requeridas por el cliente), consiguiendo una experiencia de usuario positiva que debe ser el objetivo primordial de la implantación de nuevas tecnologías.

En la actualidad, existen en el mercado diversas herramientas que facilitan la puesta en servicio de un sistema (Pc Mover[1], Transwiz[2], EaseUs Todo Backup[3], ...), basadas en el traspaso de perfil de usuario. Pero estas herramientas no siempre ofrecen ventajas, debido a:

- El alto coste económico incrementa substancialmente el presupuesto de gasto dedicado a dicha transformación.
- Las funcionalidades que ofrecen no cumplen con todos los requisitos requeridos para una correcta y total migración.
- Necesidad de realizar las instalaciones de los productos o estar conectados a la red de manera simultánea en dos equipos en caso de cambio de máquina.
- Aplicaciones enfocadas a necesidad de cambio de máquina. Sino existiera esa necesidad de cambio, no sería factible.
- En escenarios de origen sin dominio, es necesaria la instalación manual del producto en cada estación de trabajo.

En este tipo de renove tecnológico de la infraestructura de la empresa con múltiples configuraciones manuales, el cual pretende estandarizar y racionalizar los diferentes recursos y servicios del negocio, se suelen detectar diversos problemas que de manera usual se subsanan sobre la marcha; pero que, a largo plazo, requieren de una solución definitiva que ataque directamente el punto crítico.

Uno de los principales escollos es el de realizar las configuraciones básicas a usuario - tras la sustitución del equipamiento-, realizadas de manera manual, invirtiendo excesivo tiempo de los recursos y que obligan al usuario final a demorarse en el reinicio de sus tareas cotidianas.

Esta falta de automatización en los procesos obliga a la empresa al aumento del coste por instalación de cada máquina, haciendo que el proyecto sea poco rentable. Además, el empleo de herramientas de soporte a la configuración existentes en el mercado no abarca la totalidad de los requerimientos necesarios y la necesidad de ser una configuración atendida por el técnico, implica una experiencia de usuario negativa.

Estas herramientas obligan a su instalación en dos equipos distintos, no contemplando la posibilidad de migrar sobre la misma máquina. La solución propuesta superaría ese escollo al trabajar mediante scripts y un share. Asimismo, las licencias ofertadas son por máquina a migrar y no contemplan el coste de mano de obra de técnico, lo que en empresas de media/alta capacidad no sería rentable. Con el mismo coste y capacidad de máquinas similar, esta nueva propuesta ofrece un producto totalmente adaptable a cada entorno.

Con la elaboración de este trabajo final de grado se pretende crear una herramienta funcional, de bajo coste y automatizada para la configuración de puestos de trabajo de manera no manual, aportando simplicidad a los procesos de trabajo actuales y reduciendo costes innecesarios para la empresa. De manera que, al realizar cualquier cambio de infraestructura (cambio de hardware o actualización del entorno), todo los *settings* de usuario requeridos estén disponibles en el nuevo escenario.

Este proyecto deberá presentar una solución global, asequible y modular, independientemente de la cantidad de usuarios o datos a gestionar, acotando el riesgo y optimizando la inversión a realizar.

Propuesta de análisis/comparación de costes:

Descripción	Horas	Coste x hora	Total coste
Desarrollo de la solución 41 días x 5 horas= 205 horas	205	18,00 €	3.690,00 €
Documentación de la solución 35 días x 5 horas= 175 horas	175	12,00 €	2.100,00 €
Total costes desarrollo			5.790,00 €

Descripción	coste x máquina	Total máquinas equiparable a coste propuesta
PC Mover PRO	35,95 €	161
Transwiz PRO	84,95 €	68
EaseUs Workstation	36,00 €	161

Figura 1. Análisis de costes

1.2 Objetivos del Trabajo

El objetivo primordial de este trabajo es la implementación de una solución modular, multiusuario, adaptable a cualquier entorno y personalizable, de automatización ágil del proceso de configuración de los puestos de trabajo en un entorno transformado o migrado tecnológicamente, permitiendo optimizar recursos. Esto permitirá, mediante un HTA, que el usuario reciba su id aleatorio (en otros casos, cada organización puede tener unas indicaciones: dni, correo, número empleado, ...) y traspase los datos según necesidades (unas mandatorias de la organización, otras libres).

Esto permite la identificación del usuario en un entorno destino haciendo que la solución sea transversal a los diferentes escenarios. Por ejemplo, soluciona la problemática de un usuario que trabaje en *workgroup* y vaya a trabajar en dominio, asociando un usuario a otro; en cambios de dominio, asociar los usuarios de ambos, sin necesidad de migrar usuario de uno a otro o creación de tabla de equivalencias.

El trabajo se centrará en la creación de un automatismo basado en scripts que apunten a los diferentes registros del sistema buscando enlazar los sistemas obsoletos con los nuevos de una manera dinámica, evitando tener que realizar intervenciones que consuma de manera innecesaria el tiempo de los recursos asignados al proyecto.

Esta reducción del tiempo de gestión ha de permitir ser más ágiles en los procesos de migración, consiguiendo un retorno de la inversión al poder reducir los costes del proyecto, tanto desde una vertiente financiera (se reducen los costes de las jornadas de migración a dedicar a cada usuario),

como desde una vertiente de negocio (se optimiza el tiempo de los recursos pudiendo dedicarlos a otros procesos que aporten más beneficio al negocio).

Los objetivos técnicos que se buscan son:

- Crear los módulos de importación de datos basados en scripts para poder capturar la información del usuario en el entorno origen. Constará de unos módulos estándar y otros personalizables. Los estándares recogerán:
 - Configuración del correo Outlook, incluyendo:
 - Firmas configuradas.
 - Nk2 en el caso que sea necesario (según versión Office).
 - Configuraciones Office (Idioma, diccionarios, complementos, plantillas, temas, etc.)
 - Unidades de red.
 - Favoritos de navegadores (Internet Explorer, Mozilla Firefox, Google Chrome).
 - Certificados públicos

Los personalizables recogen los requerimientos específicos de cada cliente como puede ser la configuración de:

- SAP GUI
 - ODBCs
 - Conexiones SNA para aplicativos Host i/o AS400
- Diseñar HTAs (Aplicaciones en HTML) que permitan al usuario identificarse mediante un id y contraseña:
 - En origen: especificar los datos a importar al nuevo entorno. Esto hará que lance los scripts de los módulos según las preferencias escogidas y guarde esa información encriptada en un *share*.
 - En destino: en base a la identificación, se buscará la configuración en el share para importarla al nuevo entorno.
 - Implantar un share donde depositar las configuraciones encriptadas de usuario.
 - Al ser una solución transversal, habrá que adaptar las configuraciones al nuevo entorno, mediante scripts que modifiquen la nueva situación. Como ejemplo, las diferentes versiones de Paquete *Office* existentes que puedan tener origen y destino. Posteriormente a esta adaptación, importar los datos.

En otras palabras, los objetivos del TFG son la optimización de los procesos de migración mediante la automatización, y la reducción de costes de dicha migración.

1.3 Enfoque y método seguido

El principal enfoque de este proyecto es la de ofrecer un producto adaptable, asequible y funcional a las organizaciones que requieran un cambio tecnológico evitando las manualidades de los procesos y englobando todas las necesidades de la migración.

Para llevar a cabo este cambio existen diferentes herramientas comerciales que pueden facilitar el proceso referenciado; pero en este caso, se ha planteado una estrategia innovadora desarrollando un producto nuevo que sea totalmente transparente para el cliente a la hora de extraer y migrar esos datos, y que facilite todas las tareas subyacentes. La elección de ésta y no de otras herramientas, es por su naturaleza modular y personalizable, que ofrece un producto "al gusto" de cada cliente.

Para ello, será necesario el uso de una metodología ágil basada en un desarrollo iterativo e incremental y colaborativo con cliente, que aporte sus ventajas durante el proceso de desarrollo e implementación y la variabilidad de escenarios que pueden aparecer. Dicha metodología permite, mediante las iteraciones en cada fase de desarrollo de producto (codificación y pruebas posteriores) incorporar modificaciones durante la implantación de la solución debido al cambio de requerimientos en un momento concreto.

Como paso previo al desarrollo de los scripts, se diseñarán los diferentes escenarios en los que se va a implantar el producto, tomando requerimientos de cada funcionalidad. De este modo, se plasmarán todos los casos de uso y las posibles líneas a abordar. Seguidamente, se codificará el script asociado a cada módulo de datos diferenciado, procediendo a la comprobación de la funcionalidad con el uso de datos ficticios en un entorno de pruebas, permitiendo verificar la satisfacción de las expectativas programadas. Para finalizar, se pasará a entorno de producción, encajando cada módulo dentro de la solución final a presentar, adaptando cada desarrollo a las necesidades pre-establecidas al inicio.

Dado que las entregas se realizan en cortos periodos de tiempo y de forma continua, permite reaccionar de manera dinámica y ágil ante posibles rectificaciones o cambios de alcance. También se prioriza ver el funcionamiento de la solución para realizar una posterior y exhaustiva documentación, optimizando en todo momento los recursos de material y tiempo.

Todos estos pasos contribuyen en la efectividad de esta, facilitando la consecución de los objetivos fijados para este proyecto, ya que, de lo contrario, se deberá asumir un riesgo de implantación de baja calidad y un cliente insatisfecho.

1.4 Planificación del Trabajo

La planificación del trabajo está basada en los entregables programados de cada PAC, así como de la entrega final y la defensa virtual. Cada hito y tarea

identificada está relacionado con estas fechas definidas y que han de garantizar el éxito del proyecto.

Para calcular los tiempos, se han tenido en cuenta jornadas semanales de 25 horas, repartidas en los siete días naturales, siendo los fines de semana y días festivos los que más carga de trabajo se podrán asumir. Los días más señalados (como es el día Navidad y fin de año) no se prevé dedicación, todo y que se compensa con días festivos propios para esas fechas.

Dicha información se ha plasmado en el siguiente diagrama de Gantt, donde se muestran, aparte de las tareas e hitos, los tiempos y recursos dedicados.

Proyecto de automatización de migración de datos de usuario	83 días	jue 27/09/18	lun 14/01/19	
Elaboración PAC1	12 días	jue 27/09/18	vie 12/10/18	Project Manager
Definición del proyecto	7 días	jue 27/09/18	vie 05/10/18	
Contextualización y justificación del TFG	2 días	jue 27/09/18	vie 28/09/18	
Justificación del TFG	1 día	lun 01/10/18	lun 01/10/18	
Definir los objetivos del TFG	2 días	mar 02/10/18	mié 03/10/18	
Establecer enfoque y método de trabajo	1 día	jue 04/10/18	jue 04/10/18	
Descripción productos obtenidos	1 día	vie 05/10/18	vie 05/10/18	
Descripción capítulos de la memoria	1 día	vie 05/10/18	vie 05/10/18	
Planificación del proyecto	6 días	vie 05/10/18	vie 12/10/18	
Identificar y secuenciar las tareas	5 días	vie 05/10/18	jue 11/10/18	
Diagrama de Gantt	1 día	vie 12/10/18	vie 12/10/18	
Documentar en memoria PAC1	12 días	jue 27/09/18	vie 12/10/18	
Documentar PAC1	11 días	jue 27/09/18	jue 11/10/18	
Elaboración de resumen, abstract y elección de licencia	1 día	vie 12/10/18	vie 12/10/18	
Revisión y entrega de PAC1	1 día	vie 12/10/18	vie 12/10/18	
Elaboración PAC2	22 días	sáb 13/10/18	vie 09/11/18	
Preparación entorno Laboratorio	2 días	sáb 13/10/18	dom 14/10/18	Administrador
Creación de las maquinas virtuales	1 día	sáb 13/10/18	sáb 13/10/18	
Instalación y configuración de las maquinas	1 día	dom 14/10/18	dom 14/10/18	
Diseño y desarrollo de los modulos	19 días	lun 15/10/18	jue 08/11/18	
Definición final de los modulos	1 día	lun 15/10/18	lun 15/10/18	Analyst
Modulo Favoritos Navegadores Web	3 días	mar 16/10/18	jue 18/10/18	
Análisis de datos a exportar	1 día	mar 16/10/18	mar 16/10/18	Developer/Analyst
Desarrollo script	2 días	mar 16/10/18	mié 17/10/18	Developer/Analyst
Validación script en laboratorio	1 día	jue 18/10/18	jue 18/10/18	Tester
Modulo Configuración de correo Outlook	4 días	jue 18/10/18	mar 23/10/18	
Análisis de datos a exportar	1 día	jue 18/10/18	jue 18/10/18	Developer/Analyst
Desarrollo script	3 días	jue 18/10/18	lun 22/10/18	Developer/Analyst
Validación script en laboratorio	1 día	mar 23/10/18	mar 23/10/18	Tester
Modulo Configuraciones Office	3 días	mar 23/10/18	jue 25/10/18	
Análisis de datos a exportar	1 día	mar 23/10/18	mar 23/10/18	Developer/Analyst
Desarrollo script	2 días	mar 23/10/18	mié 24/10/18	Developer/Analyst
Validación script en laboratorio	1 día	jue 25/10/18	jue 25/10/18	Tester
Modulo Unidades de red	3 días	jue 25/10/18	lun 29/10/18	
Análisis de datos a exportar	1 día	jue 25/10/18	jue 25/10/18	Developer/Analyst
Desarrollo script	2 días	jue 25/10/18	vie 26/10/18	Developer/Analyst
Validación script en laboratorio	1 día	lun 29/10/18	lun 29/10/18	Tester
Modulo Certificados	3 días	lun 29/10/18	mié 31/10/18	
Análisis de datos a exportar	1 día	lun 29/10/18	lun 29/10/18	Developer/Analyst
Desarrollo script	2 días	lun 29/10/18	mar 30/10/18	Developer/Analyst
Validación script en laboratorio	1 día	mié 31/10/18	mié 31/10/18	Tester
Modulo SAP	3 días	mié 31/10/18	vie 02/11/18	
Análisis de datos a exportar	1 día	mié 31/10/18	mié 31/10/18	Developer/Analyst
Desarrollo script	2 días	mié 31/10/18	jue 01/11/18	Developer/Analyst
Validación script en laboratorio	1 día	vie 02/11/18	vie 02/11/18	Tester
Modulo ODBC	3 días	vie 02/11/18	mar 06/11/18	
Análisis de datos a exportar	1 día	vie 02/11/18	vie 02/11/18	Developer/Analyst
Desarrollo script	2 días	vie 02/11/18	lun 05/11/18	Developer/Analyst
Validación script en laboratorio	1 día	mar 06/11/18	mar 06/11/18	Tester
Modulo Conexiones SNA	3 días	mar 06/11/18	jue 08/11/18	
Análisis de datos a exportar	1 día	mar 06/11/18	mar 06/11/18	Developer/Analyst
Desarrollo script	2 días	mar 06/11/18	mié 07/11/18	Developer/Analyst
Validación script en laboratorio	1 día	jue 08/11/18	jue 08/11/18	Tester
Documentar en memoria PAC2	22 días	sáb 13/10/18	vie 09/11/18	
Revisión de PAC1	1 día	lun 22/10/18	lun 22/10/18	
Documentar PAC2	21 días	sáb 13/10/18	jue 08/11/18	Developer/Analysts
Revisión y entrega de PAC2	1 día	vie 09/11/18	vie 09/11/18	Project Manager

Figura 2. Diagrama Gantt I

Elaboración PAC3	21 días	sáb 10/11/18	vie 07/12/18	
 Diseño y desarrollo HTA origen	7 días	sáb 10/11/18	lun 19/11/18	
Análisis para el desarrollo	1 día	sáb 10/11/18	sáb 10/11/18	Developer/Analyst
Análisis de encriptación de datos	1 día	lun 12/11/18	lun 12/11/18	Developer/Analyst
Desarrollo script de llamada a los módulos	4 días	mar 13/11/18	vie 16/11/18	Developer/Analyst
Redireccionamiento datos a la ubicación determinada	1 día	lun 19/11/18	lun 19/11/18	Developer/Analyst
Validación desarrollo HTA	1 día	lun 19/11/18	lun 19/11/18	Tester
 Diseño y desarrollo HTA destino	7 días	lun 19/11/18	mar 27/11/18	
Análisis para el desarrollo	1 día	lun 19/11/18	lun 19/11/18	Developer/Analyst
Análisis desenscriptación de datos	1 día	mar 20/11/18	mar 20/11/18	Developer/Analyst
Desarrollo de búsqueda y recuperación de datos	4 días	mié 21/11/18	lun 26/11/18	Developer/Analyst
Validación desarrollo HTA	1 día	mar 27/11/18	mar 27/11/18	Tester
 Scripts de importación	6 días	mar 27/11/18	mar 04/12/18	
Comprobación de versiones entorno destino	2 días	mar 27/11/18	mié 28/11/18	Developer/Analyst
Desarrollo de scripts por cada uno de los módulos	3 días	jue 29/11/18	lun 03/12/18	Developer/Analyst
Validación script en laboratorio	1 día	mar 04/12/18	mar 04/12/18	Tester
Validación de todo el proceso de migación de datos	2 días	mié 05/12/18	jue 06/12/18	Tester
 Documentar en la memoria la PAC3	21 días	sáb 10/11/18	vie 07/12/18	
Revisión de PAC2	1 día	lun 19/11/18	lun 19/11/18	
Documentar PAC3	20 días	sáb 10/11/18	jue 06/12/18	Developer/Analysts
Revisión y entrega de PAC3	1 día	vie 07/12/18	vie 07/12/18	Project Manager
Entrega Final	21 días	sáb 08/12/18	vie 04/01/19	Project Manager
 Escritura documento final de la memoria	9 días	sáb 08/12/18	mié 19/12/18	
Revisión de PAC3	1 día	lun 17/12/18	lun 17/12/18	
Revisión final de memoria	9 días	sáb 08/12/18	mié 19/12/18	
 Elaboración presentación	10 días	jue 20/12/18	mié 02/01/19	
Elaboración documento presentación	7 días	jue 20/12/18	vie 28/12/18	
Grabación vídeo presentación	3 días	lun 31/12/18	mié 02/01/19	
 Entrega TFG	2 días	jue 03/01/19	vie 04/01/19	
Defensa Virtual	2 días	sáb 12/01/19	lun 14/01/19	Project Manager

Figura 3. Diagrama Gantt II

Diagrama de Gantt:

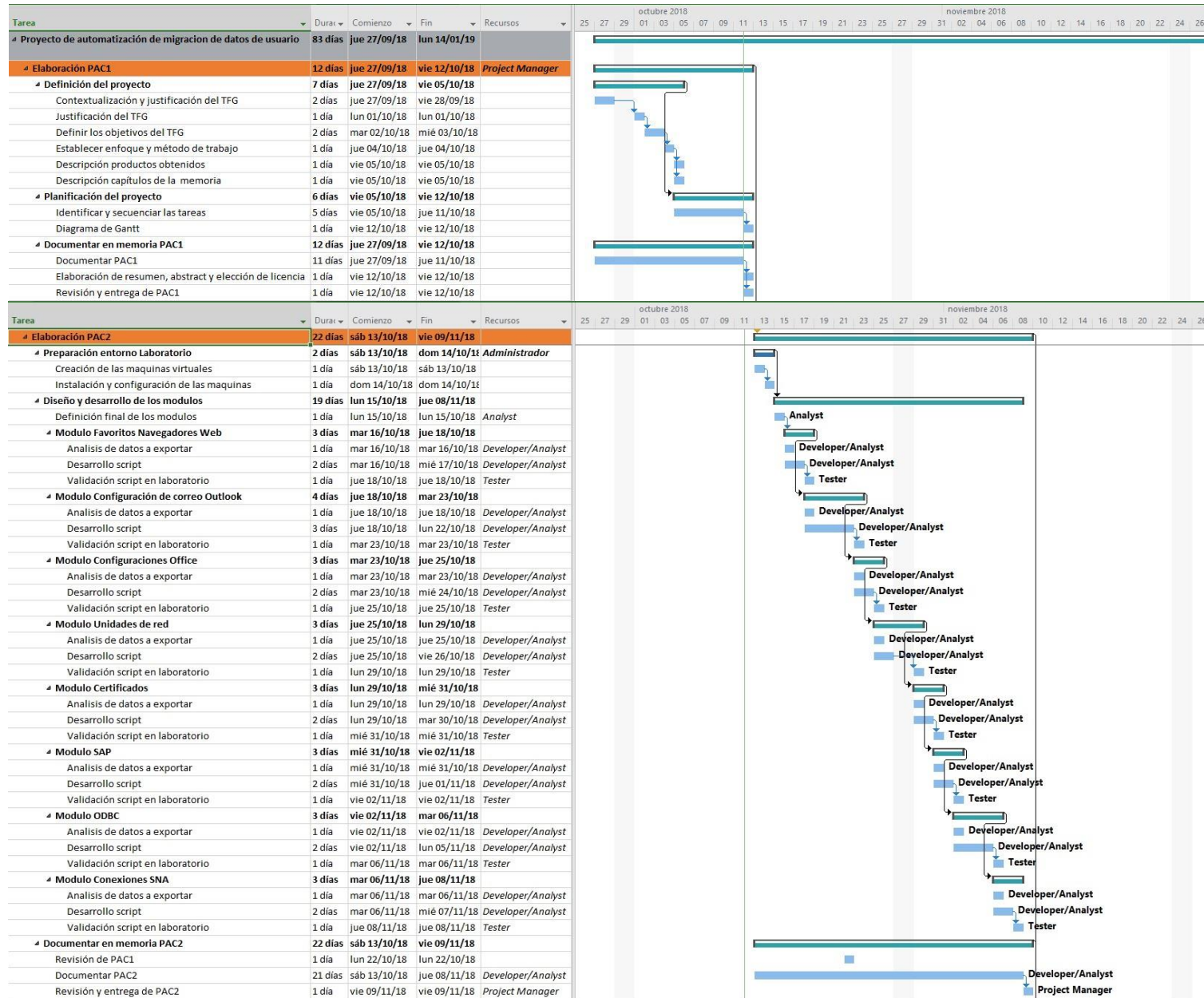


Figura 4. Diagrama Gantt III

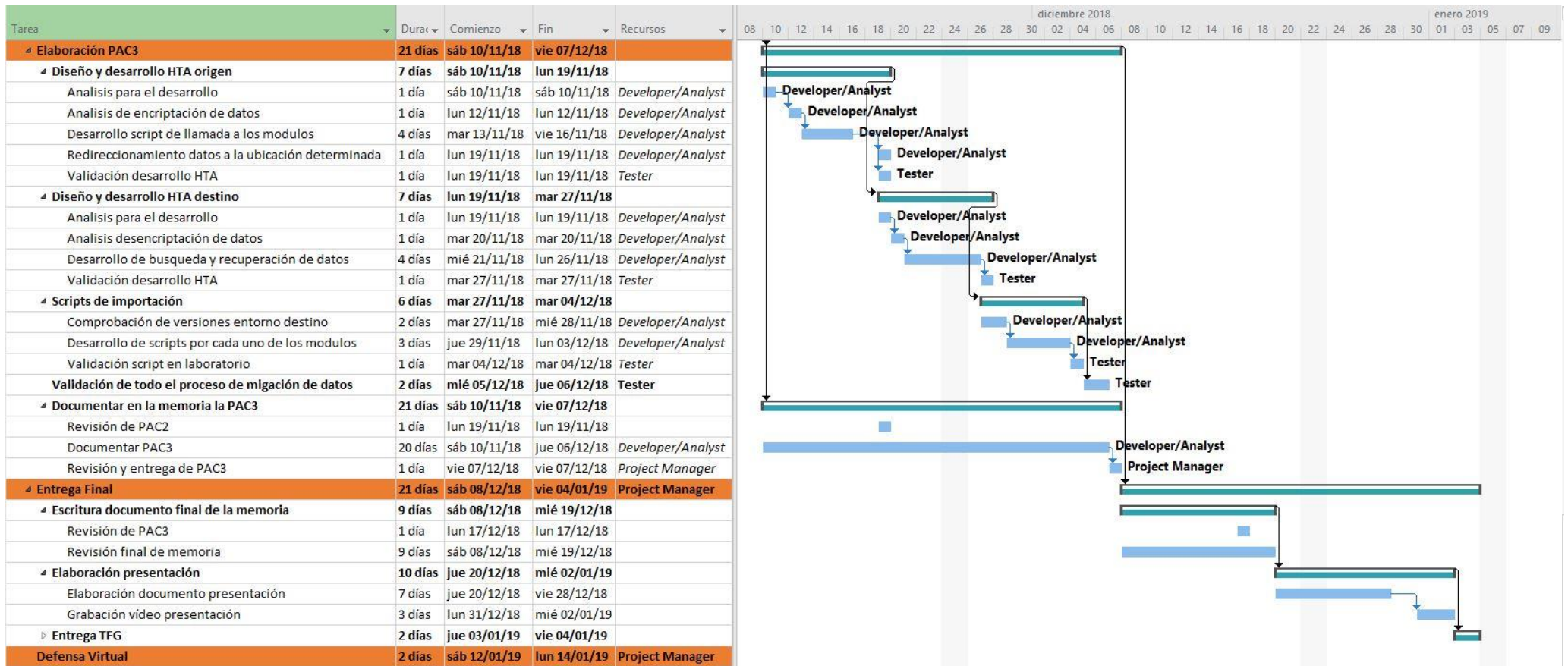


Figura 5. Diagrama Gantt IV

1.5 Breve resumen de productos obtenidos

Este proyecto propone la obtención de un producto principalmente, y es una solución basada en scripting que recabe e importe posteriormente los datos de usuario.

De manera enumerada los entregables serán:

- Scripts de captura y entrega de datos de cada módulo:
 - Configuración correo
 - Mapeo unidades de red
 - Certificados
 - Navegadores web
 - Configuraciones Office
 - SAP GUI
 - ODBC
 - Conexiones SNA

- HTA de:
 - Origen: Solicitud de creación de id y contraseña de usuario
 - Destino: Validación de credenciales para recuperación de datos

- Memoria, presentación y video final del TFG.

1.6 Breve descripción de los otros capítulos de la memoria

A modo resumen, se relacionan a continuación el resto de los capítulos que conforman la memoria.

En el capítulo 2.1 se define el entorno de pruebas que se utilizará para el desarrollo del proyecto.

En el capítulo 2.2 se presenta el diseño de los módulos de datos de usuarios y se desarrollan los diferentes scripts que faciliten su captura.

El capítulo 2.3 muestra el diseño e implementación del HTA de origen con los scripts asociados que permitirán al usuario crear su id y copiar sus datos al *share*.

El capítulo 2.4 muestra el diseño e implementación del HTA de destino con los scripts asociados que permitan al usuario recuperar la información.

El capítulo 2.5 aborda los scripts de importación de datos al nuevo entorno del usuario y que apuntan a los diferentes módulos.

El capítulo 2.6 muestra la validación de todo el proceso desarrollado de módulos.

En el capítulo 3 se incluye un análisis y reflexión final sobre el estado y consecución del proyecto a modo de conclusión.

El capítulo 4 define los diferentes términos utilizados a lo largo de la memoria.

El capítulo 5 recoge las referencias bibliográficas consultadas en el desarrollo del proyecto.

Por último, el capítulo 6 presenta aquellos ítems que no han podido incluirse en los diferentes capítulos de la memoria y que ofrecen en detalle ciertos puntos.

2. Diseño e implementación de la solución

2.1 Entorno Laboratorio

Para el desarrollo del proyecto se han contemplado diferentes escenarios con diferentes sistemas operativos o versiones de aplicativos (Office, SAP GUI, navegadores...).

Para ello, se han configurado 5 máquinas virtuales mediante VirtualBox, con los sistemas operativos: Windows XP, Windows 7, Windows 8.1, Windows 10 y Windows Server 2012 R2. Se han propuesto estos sistemas basando la decisión en esta comparativa sobre el uso de versiones de Windows de 3 analistas (Net Marquet Share, W3 Cuonter, Stat Cuonter) [\[4\]](#):

fuelle	Net Marquet Share	W3Cuonter	Stat Cuonter
Windows 10	35,27%	15,02%	51,94%
Windows 7	41,82%	13,50%	36,31%
Windows 8.1	5,30%	2,43%	7,02%
Windows XP	4,23%	ND	2,16%
Windows 8	1,10%	ND	1,94%
Windows Vista	0,31%	ND	0,55%
Windows 2000	0,01%	ND	0,01%

Figura 6. Versiones Windows más usadas

Por último, se ha configurado un Server 2012 R2 para poder poner en dominio alguno de los equipos clientes.

Por tanto, la validación del desarrollo se realiza sobre estos 4 clientes Windows, todo y que la solución debería ser transversal al resto de los sistemas operativos Windows, ya que el desarrollo implica registros y variables de entorno.

A continuación, se puede ver una captura de las 5 máquinas virtuales configuradas:

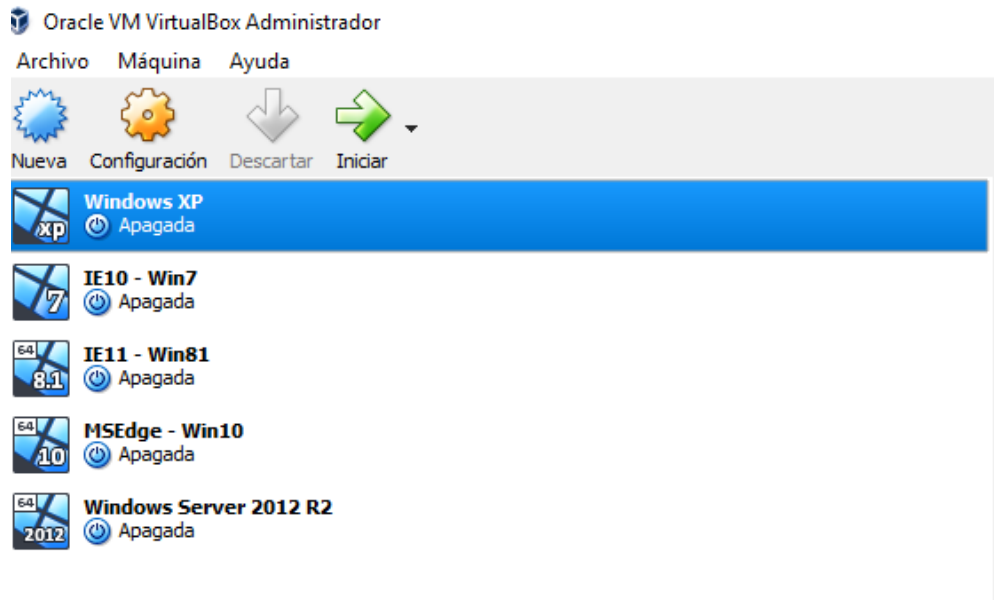


Figura 7. Máquinas virtuales

Considerando que Microsoft Office es una de las aplicaciones fundamentales como herramienta de trabajo en todas las empresas, y siendo uno de los puntos críticos de la solución planteada, se han instalado diferentes versiones en cada una de las máquinas para poder abarcar el impacto global en todo el proceso:

- **Windows XP:** MS Office 2003
- **Windows 7:** MS Office 2010
- **Windows 8.1:** MS Office 2013
- **Windows 10:** MS Office 2016

Así mismo, también se han tenido en cuenta las diferentes versiones de los navegadores o de SAP GUI a la hora de preparar el entorno de trabajo.

2.2 Diseño y desarrollo de Módulos exportación de datos

En este punto, se definen los módulos de datos de los usuarios que se van a trabajar a la hora de realizar las exportaciones durante la migración. En cada uno de ellos, se realiza un análisis de los datos requeridos, la implementación de un script de captura y la validación en el entorno de pruebas (en cada una de las máquinas configuradas).

El lenguaje utilizado a la hora de realizar estos guiones o scripts es **VBScript** (Visual Basic Script Edition). La elección del lenguaje de los guiones o scripts se ha basado en la transversalidad de la solución. Es por ello que se han tenido en consideración los siguientes puntos:

- Powershell requiere, en Windows XP, la instalación del Service Pack 2 o 3, dependiendo de la versión de éste (PowerShell 1.0 o 2.0) para poder ser interpretado de forma correcta, y esto ocasiona que no funcione en una solución transversal.

- Asimismo, VBScript es interpretado por Windows Script Host, que está instalado de forma predeterminada desde Windows 98, haciendo que sea interpretado en cualquier sistema operativo Windows. En el caso de PowerShell, es necesaria la instalación en Windows XP (requiriendo mínimo SP2).
- Batch (.bat) es un lenguaje obsoleto, de programación lineal y limitado a los comandos MSDos. Sin embargo, VBScript permite usar objetos y un amplio surtido de opciones.
- VBScript también es utilizado para la creación de HTAs, como se trata en este proyecto.

2.2.1 Módulo Favoritos Navegadores

Este módulo de datos recoge la información de los usuarios referente a los navegadores de Internet más utilizados en el mercado, como son Internet Explorer, Mozilla Firefox y Google Chrome (en diferentes versiones). Uno de los puntos esenciales a recoger son los denominados Favoritos, objetivo de este módulo, que refleja aquellas URLs o páginas más interesantes para el usuario y que son guardadas en el navegador para tener un acceso rápido y sencillo a posteriori.

Internet Explorer

Los favoritos de este navegador nativo de MS Windows, en todas sus versiones, se guardan en el perfil local del usuario del sistema operativo. Dependiendo de la versión del S.O. la ruta del perfil varía. Concretamente, en:

- WinXP – C:\Documents and Settings\%userprofile%\favoritos
- Posteriores – C:\Users\%userprofile%\favoritos

Utilizando la variable de entorno %userprofile%, por un lado, no hará falta tener en cuenta la ruta completa del perfil de usuario en cada sistema; y, por otro lado, permite seleccionar el usuario en ejecución (especialmente práctico en máquinas con más de un perfil de usuario).

Por otra parte, dependiendo del idioma del S.O. existen dos variantes en el nombre del directorio: "Favoritos" o "Favorites". En este caso, también se tiene en cuenta a la hora de hacer la exportación.

Para finalizar, se realiza una consulta WMI para comprobar el sistema operativo. Si éste es Windows 10, también se exportarán los favoritos del navegador Microsoft Edge. Para ello se utiliza la variable de entorno %localappdata% copiándose el contenido de la siguiente ruta:

- M. Edge – %LocalAppData%\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\MicrosoftEdge\User\Default\DataStore

Solucionados estos escollos, el contenido del directorio pasa a ser copiado. El script generado para este módulo es "IE.vbs".

```

Const Overwrite = True
Dim oFSO,objWShell
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")

CarpetaExportacion = WScript.Arguments.Item(0)
IE = "\IE"
Edge = "\Edge"
'Se comprueba si existe la carpeta para la exportación y se crea si es necesario
If Not oFSO.FolderExists(CarpetaExportacion) Then
    oFSO.CreateFolder CarpetaExportacion
    oFSO.CreateFolder CarpetaExportacion & IE
End If

Dim CarpetaUsuario
CarpetaUsuario = objWShell.expandEnvironmentStrings("%userprofile%")

'Se exportan los favoritos de IE, teniendo en cuenta el idioma. Si existe el directorio, copia el contenido
CarpetaUsuarioIE = CarpetaUsuario & "\favoritos"
If oFSO.FolderExists(CarpetaUsuarioIE) Then
    oFSO.CopyFolder CarpetaUsuarioIE, CarpetaExportacion & IE, Overwrite
End If
CarpetaUsuarioIE = CarpetaUsuario & "\favorites"
If oFSO.FolderExists(CarpetaUsuarioIE) Then
    oFSO.CopyFolder CarpetaUsuarioIE, CarpetaExportacion & IE, Overwrite
End If

'Se realiza consulta WMI para obtener la versión del SO
Set SystemSet = GetObject("winmgmts:").InstancesOf ("Win32_OperatingSystem")
for each System in SystemSet
    VersionSO = System.Version
Next
VersionSO = left (VersionSO, 2)

'En el caso que sea Windows 10 se exportan los Favoritos del Navegador Edge
If VersionSO = "10" then
    oFSO.CreateFolder CarpetaExportacion & Edge
    Dim LocalData
    LocalData = objWShell.expandEnvironmentStrings("%LocalAppData%")
    CarpetaEdge = LocalData &
"\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\MicrosoftEdge\User\Default"
    Store = "\DataStore"
    'Se comprueba que existe el directorio y se exporta
    If oFSO.FolderExists(CarpetaEdge & Store) Then
        oFSO.CopyFolder CarpetaEdge & Store, CarpetaExportacion & Edge & Store, Overwrite
    End If
End If

```

Mozilla Firefox

En el caso de Firefox, también en todas sus versiones, los favoritos se almacenan en un documento denominado "places.sqlite", que se encuentra en:

- WinXP - C:\Documents and Settings\%userprofile%\Datos de Programa\Mozilla\Firefox\Profiles\“aleatorio”.default
- Posteriores - C:\Users\%userprofile%\AppData\Roaming\Mozilla\Firefox\Profiles\“aleatorio”.default

En el desarrollo del script se ha utilizado otra variable de entorno, %AppData%, carpeta oculta donde se guardan los datos de aplicaciones del usuario dentro de la carpeta de usuario.

- WinXP - C:\Documents and Settings\%userprofile%\Datos de Programa
- Posteriores - C:\Users\%userprofile%\AppData\Roaming

Sin embargo, el documento "places.sqlite" está guardado en una carpeta el nombre de la cual se genera de manera aleatoria. Para poder copiar el fichero, se debe obtener este nombre. Para ello, el script extrae el nombre del archivo (profiles.ini) existente en la ruta %AppData%\Mozilla\Firefox. Dentro de este fichero .ini aparece el nombre aleatorio.

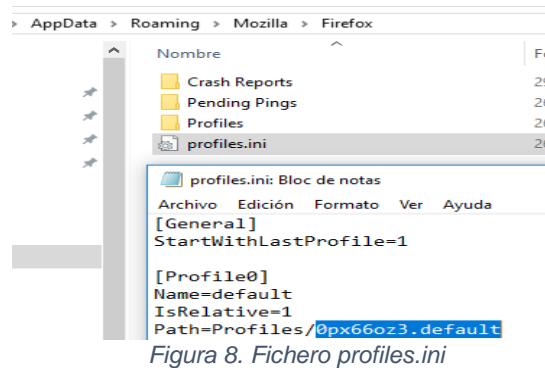


Figura 8. Fichero profiles.ini

Una vez obtenida la ruta completa, con todos los condicionantes, el contenido pasa a ser copiado. El script generado para este módulo es "Firefox.vbs".

```

Const Overwrite = True
Dim oFSO,objWShell,CarpetaExportacion
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = CreateObject("WScript.Shell")

CarpetaExportacion = WScript.Arguments.Item(0)
FirefoxProfileFileName = "profiles.ini"
FirefoxProfilePath = "\\Mozilla\Firefox\"

'Nombre documento favoritos
Dim places
places = "places.sqlite"

'Se comprueba si existe la carpeta para la exportaci  se crea si es necesario
If Not oFSO.folderExists(CarpetaExportacion) Then
    oFSO.CreateFolder(CarpetaExportacion)
End If

AppData = objWShell.ExpandEnvironmentStrings("%APPDATA%")
Temp_Path = AppData & FirefoxProfilePath & FirefoxProfileFileName

'Si existe el archivo .ini comienza a leerlo para extraer la ruta del perfil
If oFSO.FileExists(Temp_Path) Then
    oFSO.CopyFile AppData & FirefoxProfilePath & FirefoxProfileFileName, CarpetaExportacion & "\" &
    FirefoxProfileFileName
    Set objIniFile = oFSO.OpenTextFile(Temp_Path, 1, False)
    Do While objIniFile.AtEndOfStream = False
        Linia = Trim(objIniFile.ReadLine)

        'Valida la secci n la linea actual del .ini
        If LCase(Linia) = "[" & LCase("Profile0") & "]" Then
            Linia = Trim(objIniFile.ReadLine)

            'Analiza las lineas hasta llegar a la siguiente secci n del .ini
            Do While Left(Linia, 1) <> "["
                'Busca la posici n del signo igual en la linea
                intEqualPos = InStr(1, Linia, "=", 1)
                If intEqualPos > 0 Then
                    strLeftString = Trim(Left(Linia, intEqualPos - 1))
                    'Valida si la palabra path esta en la linea
                    If LCase(strLeftString) = LCase("Path") Then
                        FFoxPath = Trim(Mid(Linia, intEqualPos + 1))
                        'En el caso que exista "path" pero este vacio
                        If FFoxPath = "" Then
                            FFoxPath = " "
                        End If
                        'Salir del bucle en el caso que se haya obtenido el path
                        Exit Do
                    End If
                End If
            End If
            'Se aborta en el caso que se haya llegado al final del archivo
            If objIniFile.AtEndOfStream Then Exit Do

            'Continuar con la siguiente linea
            Linia = Trim(objIniFile.ReadLine)
        Loop
    Exit Do
    End If
    Loop
objIniFile.Close
End If

```

```
'Se reemplaza la barra para a rlo a la ruta completa
FFoxPath = Replace (FFoxPath, "/", "\")

'Crea archivo con la ruta del perfil de Firefox necesaria en la importaci n
Set fileVersion =
oFSO.CreateTextFile (CarpetaExportacion + "\path.txt", Overwrite)
fileVersion.WriteLine (FireFoxProfilePath & FFoxPath)
fileVersion.Close

'Se obtiene la ruta completa del Bookmarks y si existe, se copia
ProfilePath = AppData & FireFoxProfilePath & FFoxPath & "\" & places
If oFSO.FileExists(ProfilePath) Then
oFSO.CopyFile ProfilePath, CarpetaExportacion & "\" & places, Overwrite
End If
```

Google Chrome

Por  ltimo, Google Chrome, en todas sus versiones, almacena los favoritos en el fichero Bookmarks, ubicado en:

WinXP - C:\Documents and Settings\%userprofile%\Config~1\Datos de programa\Google\Chrome\User Data\Default

Posteriores - C:\Users\%userprofile%\AppData\Local\Google\Chrome\User Data\Default

Para las versiones posteriores a WinXP, existe una variable de entorno, %LOCALAPPDATA%, que equivale a C:\Users\%userprofile%\AppData\Local. No obstante, esta variable no existe en Windows XP, por lo tanto, son rutas diferenciadas. Por esta raz n, antes de realizar la copia de datos se debe conocer si el entorno origen es XP o no. Para obtener este dato, se realiza una consulta WMI, donde aparece el n mero de versi n de Windows a tratar [5].

Operating system	Version number
Windows 10	10.0*
Windows 8.1	6.3*
Windows 8	6.2
Windows 7	6.1
Windows Vista	6.0
Windows XP 64-Bit Edition	5.2
Windows XP	5.1
Windows 2000	5.0

Figura 9. Equivalencias sistema operativo/versi n

Una vez obtenida la ruta completa, el contenido pasa a ser copiado. El script generado para este m dulo es "Chrome.vbs".

```
Const Overwrite = True
Dim oFSO,objWShell
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")

CarpetaExportacion = WScript.Arguments.Item(0)

'Se comprueba si existe la carpeta para la exportaci n y se crea si es necesario
If Not oFSO.FolderExists(CarpetaExportacion) Then
oFSO.CreateFolder CarpetaExportacion
End If

'Se realiza consulta WMI para obtener la versi n del SO
Set SystemSet = GetObject("winmgmts:").InstancesOf ("Win32_OperatingSystem")
for each System in SystemSet
VersionSO = System.Version
Next
VersionSO = left (VersionSO, 3)
```

```

' En el caso que sea Windows XP apuntamos a una ruta diferente que con el resto de SO.
If VersionSO = "5.1" or VersionSO = "5.2" then
    Dim CarpetaUsuario
    CarpetaUsuario = objWShell.expandEnvironmentStrings("%userprofile%")
    CarpetaChrome = CarpetaUsuario & "\Config~1\Datos de programa\Google\Chrome\User Data\Default"
    'Si existe la ruta se copia el Bookmarks
    If oFSO.FolderExists(CarpetaChrome) Then
        oFSO.CopyFile CarpetaChrome & "\Bookmarks", CarpetaExportacion & "\Bookmarks", Overwrite
    End If
Else
    Dim AppData
    AppData = objWShell.expandEnvironmentStrings("%LOCALAPPDATA%")
    CarpetaChrome = AppData & "\Google\Chrome\User Data\Default"
    'Si existe la ruta se copia el Bookmarks
    If oFSO.FolderExists(CarpetaChrome) Then
        oFSO.CopyFile CarpetaChrome & "\Bookmarks", CarpetaExportacion & "\Bookmarks", Overwrite
    End If
End If

```

2.2.2 Módulo Configuración de correo Outlook

El módulo de configuración de correo Outlook recoge la información de los usuarios del cliente de correo Outlook de MS Office y sus versiones, uno de los más utilizados a nivel organizativo. Este módulo permite exportar esa configuración exclusiva de cada organización sin tener que realizar acciones manuales posteriormente.

Dentro de los datos a los que se accede, aparte de la configuración del buzón de correo, está la firma personalizada de cada usuario, los diseños de fondo generados y los archivos ".nk2" que contiene la lista de contactos "cacheados" en el cliente (autocompleta la dirección de correo de los contactos a los que alguna vez se ha enviado un correo).

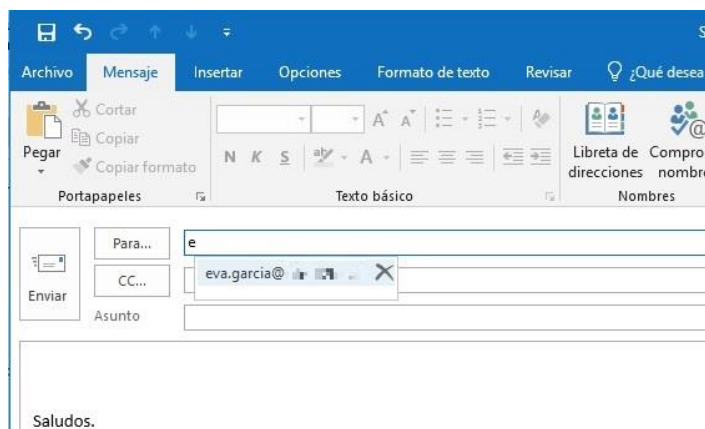


Figura 10. Ejemplo contacto "cacheado" en Outlook

Para la realización del script, se ha de tener en cuenta la versión de MS Office de la máquina, ya que, dependiendo de ésta, la ruta del registro de Windows al que apuntar es diferente. Por consiguiente, se hace la consulta de la versión mediante HKCR\Outlook.Application\CurVer\

Suite Office	Version number
Office 2016	16.0
Office 2013	15.0
Office 2010	14.0
Office 2007	12.0
Office 2003	11.0
Office XP	10.0
Office 2000	9.0

Figura 11. Equivalencia Office/Versión

Y se extrae un árbol de claves de registro u otro:

- Versión 16 o 15: HKEY_CURRENT_USER\Software\Microsoft\Office\
%versionNumber%\Outlook\Profiles
- Versiones anteriores: HKEY_CURRENT_USER\Software\Microsoft\Windows
NT\CurrentVersion\Windows Messaging Subsystem

El script genera un fichero con la versión de Office del equipo origen para posterior tratamiento en la importación a la máquina destino (se puede dar el caso que se requiera modificaciones si la versión de origen/destino es diferente).

La clave de registro en la cual se guardan opciones del cliente a nivel de Exchange es la misma para todas las versiones:

- HKEY_CURRENT_USER\Software\Microsoft\Exchange\Client\Options

Las firmas de cada usuario se guardan en la siguiente ruta, utilizando como variable de entorno %AppData%, ya comentada en apartados anteriores, y se hace una copia del contenido:

- %AppData%\Microsoft\“Signatures” o “Firmas” (teniendo en cuenta el idioma)

También se exportará lo que se conoce como Stationery o diseños de fondo. Es cuando el usuario se configura el fondo del cuerpo del correo con temas o efectos haciendo servir cambios de fuentes, colores de fondo, líneas, imágenes, etc. Es muy personalizable, hasta la posibilidad de incluir el logotipo de la organización. Se realiza la copia de esta configuración que está contenida en la siguiente ruta:

- %AppData%\ Microsoft\“Stationery” o diseños de fondo (como en el caso anterior, teniendo en cuenta el idioma)

El .nk2 se guarda (haciendo uso también de la variable %AppData%) en la ruta siguiente. Se crea un fichero .nk2 por cada perfil de Outlook configurado. Por tanto, lo que se realiza es una búsqueda de todos los ficheros existentes .nk2.

- %AppData%\Microsoft\Outlook

De modo que, obtenidos todos los datos, el contenido se copia para su exportación. El script generado para este módulo es “Outlook.vbs”.

```
Const Overwrite = True
Dim oFSO,objWShell,strKey,NumVersion, fileVersion
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")

CarpetaExportacion = WScript.Arguments.Item(0)
'Se comprueba si existe la carpeta para la exportación y se crea si es necesario
If Not oFSO.FolderExists(CarpetaExportacion) Then
    oFSO.CreateFolder CarpetaExportacion
End If

'Se consulta la versión de Outlook
strKey = "HKCR\Outlook.Application\CurVer\"
NumVersion = objWShell.RegRead(strKey)
NumVersion = right (NumVersion, 2)
```

```

'Se genera fichero con la versión de Office para el posterior tratamiento
Set fileVersion = oFSO.CreateTextFile (CarpetaExportacion + "\version.txt", Overwrite)
fileVersion.WriteLine (NumVersion)
fileVersion.Close

'Dependiendo de la versión de Office, se exporta un arbol de claves o otro
If NumVersion = 16 or NumVersion = 15 Then
    regCmd = "cmd.exe /c reg export HKEY_CURRENT_USER\Software\Microsoft\Office\" + NumVersion +
    ".0\Outlook\Profiles " + CarpetaExportacion + "\SessionData2.reg"
    objWShell.run regCmd, Overwrite
Else
    WindowsMessaging = ("\"HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Windows Messaging
Subsystem\Profiles")
    regCmd2 = "cmd.exe /c reg export " + WindowsMessaging + " " + CarpetaExportacion + "\SessionData2.reg"
    objWShell.run regCmd2, Overwrite
End If

'Se exporta arbol de claves común a todas las versiones
regCmd = "cmd.exe /c reg export HKEY_CURRENT_USER\Software\Microsoft\Exchange\Client\Options " +
CarpetaExportacion + "\SessionData1.reg"
objWShell.run regCmd, Overwrite

'Se exportan las firmas, teniendo en cuenta el idioma. Si existe el directorio, copia el contenido
AppData = objWShell.expandEnvironmentStrings("%APPDATA%")
Firma = "\Signatures"
CarpetaFirma = AppData & "\Microsoft" & Firma
If oFSO.FolderExists(CarpetaFirma) Then
    oFSO.CopyFolder CarpetaFirma, CarpetaExportacion & Firma, Overwrite
End If

Firma = "\Firmas"
CarpetaFirma = AppData & "\Microsoft" & Firma
If oFSO.FolderExists(CarpetaFirma) Then
    oFSO.CopyFolder CarpetaFirma, CarpetaExportacion & "\Signatures", Overwrite
End If

'Se exportan los diseños de fondo, teniendo en cuenta el idioma. Si existe el directorio, copia el
contenido
Stationery = "\Stationery"
CarpetaStationery = AppData & "\Microsoft" & Stationery
If oFSO.FolderExists(CarpetaStationery) Then
    oFSO.CopyFolder CarpetaStationery, CarpetaExportacion & Archivos & Stationery, Overwrite
End If

Stationery = "\diseños de fondo"
CarpetaStationery = AppData & "\Microsoft" & Stationery
If oFSO.FolderExists(CarpetaStationery) Then
    oFSO.CopyFolder CarpetaStationery, CarpetaExportacion & Archivos & "\Stationery", Overwrite
End If

'nk2
'Si existe el directorio se copian los .nk2 que existan
CarpetaNk2 = AppData & "\Microsoft\Outlook"
If oFSO.FolderExists(CarpetaNk2) Then
    Set pathNk2 = oFSO.GetFolder(CarpetaNk2).Files
    for each file in pathNk2
        If Mid(file.NAME,len(file.NAME)-3,4) = ".NK2" then
            oFSO.CopyFile CarpetaNk2 + "\" + file.NAME, CarpetaExportacion + "\" + file.NAME, Overwrite
        end if
    next
End If

```

2.2.3 Módulo Configuraciones Office

El módulo de configuraciones Office obtiene toda la información referente a la personalización de los usuarios de toda la suite de MS Office. Concretamente se trata de la configuración existente cuando, en cualquier producto de la suite, se accede a: Archivo>Opciones. Por tanto, se capturan opciones visuales (por ejemplo: apariencia del *ribbon*), ruta de guardado por defecto o temporales, complementos activos, barra de herramientas de acceso rápido, etc.

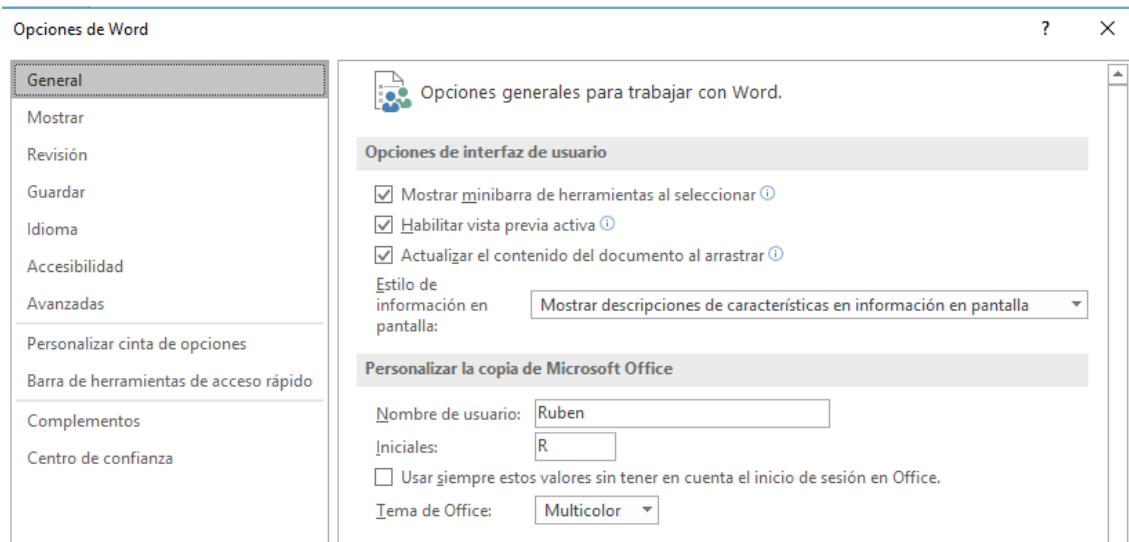


Figura 12. Opciones configuración MS Office

Como en el caso anterior, también se genera un archivo donde se guarda la versión del equipo de origen para el posterior tratamiento en la importación si fuese necesario.

Para realizar esta tarea se extrae el árbol de registro de Windows donde se almacena toda esta información, cabe indicar que es la misma ruta para todas las versiones:

- HKEY_CURRENT_USER\Software\Microsoft\Office

Con el siguiente árbol de registro de Windows se consigue obtener la información referente a la configuración del corrector de ortografía y gramática:

- HKEY_CURRENT_USER\Software\Microsoft\Shared Tools\Proofing Tools

Las siguientes configuraciones se obtienen de realizar la copia de archivos concretos. Todos ellos se localizan en %AppData%\Microsoft:

- Diccionarios Personalizados: *Actuales hasta versión 2007:* %AppData%\Microsoft\Uproof. *Versiones anteriores a 2007:* %AppData%\Microsoft\Proof
- Plantillas generadas: Teniendo en cuenta el idioma: %AppData%\Microsoft\“Templates” o “Plantillas”

De modo que, obtenidos todos los datos, tanto la exportación del registro como el contenido se copia para su exportación. El script generado para este módulo es “Office.vbs”.

```

Const Overwrite = True
Dim oFSO,objWShell,strKey,NumVersion, fileVersion
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")
CarpetaExportacion = WScript.Arguments.Item(0)
'Capturar Versión de Office
strKey = "HKCR\Outlook.Application\CurVer\"
NumVersion = objWShell.RegRead(strKey)
NumVersion = right (NumVersion, 2)

```

```

'Se comprueba si existe la carpeta para la exportación y se crea si es necesario
If Not oFSO.FolderExists(CarpetaExportacion) Then
  oFSO.CreateFolder CarpetaExportacion
End If

'Crear archivo de la versión de Office Origen necesaria en la importación
Set fileVersion = oFSO.CreateTextFile (CarpetaExportacion + "\version.txt", Overwrite)
fileVersion.WriteLine (NumVersion)
fileVersion.Close

'Exportar settings Office
regCmd = "cmd.exe /c reg export HKEY_CURRENT_USER\Software\Microsoft\Office " + CarpetaExportacion +
"\Settings1.reg"
objWShell.run regCmd, Overwrite

ProofingReg = ("\"HKEY_CURRENT_USER\Software\Microsoft\Shared Tools\Proofing Tools")
regCmd2 = "cmd.exe /c reg export " + ProofingReg + " " + CarpetaExportacion + "\Settings2.reg"
objWShell.run regCmd2, Overwrite

'Exportar archivos configuración: Diccionarios y plantillas, teniendo en cuenta idioma y versiones
AppData = objWShell.expandEnvironmentStrings("%APPDATA%")

Archivos = "\Archivos"
If Not oFSO.FolderExists(CarpetaExportacion & Archivos) Then
  oFSO.CreateFolder CarpetaExportacion & Archivos
End If

Proof = "\Proof"
CarpetaProof = AppData & "\Microsoft" & Proof
If oFSO.FolderExists(CarpetaProof) Then
  oFSO.CopyFolder CarpetaProof, CarpetaExportacion & Archivos & "\UProof", Overwrite
End If

UProof = "\UProof"
CarpetaUProof = AppData & "\Microsoft" & UProof
If oFSO.FolderExists(CarpetaUProof) Then
  oFSO.CopyFolder CarpetaUProof, CarpetaExportacion & Archivos & UProof
End If

Templates = "\Templates"
CarpetaTemplates = AppData & "\Microsoft" & Templates
If oFSO.FolderExists(CarpetaTemplates) Then
  oFSO.CopyFolder CarpetaTemplates, CarpetaExportacion & Archivos & Templates, Overwrite
End If

Templates = "\Plantillas"
CarpetaTemplates = AppData & "\Microsoft" & Templates
If oFSO.FolderExists(CarpetaTemplates) Then
  oFSO.CopyFolder CarpetaTemplates, CarpetaExportacion & Archivos & "\Templates", Overwrite
End If

```

2.2.4 Módulo Unidades de Red

El módulo de unidades de red captura toda la información en lo que hace referencia a esta configuración del sistema operativo. Hay que tener en cuenta que la configuración que se captura es el tipo de conexión de la unidad, letra utilizada para el mapeo, *path* remoto, etc. En ningún caso se gestionan los permisos de accesos a estos *path* remotos.

▾ Ubicaciones de red (2)



Figura 13. Unidades de red mapeadas

En este caso, la captura de datos se realiza mediante el registro de Windows en la sección de "Current User", es decir, la configuración del usuario que ha iniciado sesión. No se ha de tener en cuenta la versión del sistema operativo ya que, en todas, la ruta del registro es la misma:

- HKEY_CURRENT_USER\Network\

El script generado para este módulo es "Unidades.vbs".

```
Const Overwrite = True
Dim oFSO,objWShell,strKey
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")

CarpetaExportacion = WScript.Arguments.Item(0)

'Arbol de registro donde se guardan las unidades de red del usuario
strKey = "HKEY_CURRENT_USER\Network\"

'Se comprueba si existe la carpeta para la exportación y se crea si es necesario
If Not oFSO.FolderExists(CarpetaExportacion) Then
    oFSO.CreateFolder CarpetaExportacion
End If

'Se exporta el arbol de registro de Windows de las unidades
regCmd = "cmd.exe /c reg export " + strKey + " " + CarpetaExportacion + "\Unidades.reg"
objWShell.run regCmd, Overwrite
```

2.2.5 Módulo Certificados

El módulo de certificados exporta los certificados digitales del usuario, instalados previamente en el equipo origen. Obviamente están contenidos en el almacén de certificados de Windows, esta ubicación consta de claves en el registro y estas claves en el registro corresponden a ficheros. Es importante indicar que los certificados a exportar son públicos, es decir, no contienen la clave privada ya que conllevaría un error de seguridad.

Como se ha comentado, únicamente se exportarán los certificados digitales del usuario – se presupone que los certificados de equipo ya estarán en el equipo destino – todo y que si el cliente solicita exportarlos únicamente habría que añadir en el script una clave de registro y un directorio a copiar. Lo mismo ocurre con los certificados desplegados por GPO.

Por norma general, estos certificados se administran desde la consola MMC, agregando el complemento de Certificados, o directamente desde *Certificate Manager Tool* (certmgr) o mediante comandos como certutil. En este caso se automatiza este proceso mediante los siguientes pasos.

Una vez aclarados los términos, el árbol del registro de Windows a exportar es:

- HKEY_CURRENT_USER\Software\Microsoft\SystemCertificates\

Se debe indicar, que esta clave de registro es común para todas las versiones de Windows. En estas claves se hace referencia a las rutas donde están ubicados los certificados y que se deberán copiar. A todas estas rutas se accede a través de la variable de entorno %AppData%:

- %AppData%\Microsoft\SystemCertificates
- %AppData%\Microsoft\Crypto
- %AppData%\Microsoft\Credentials
- %AppData%\Microsoft\Protect

De modo que, obtenidos todos los datos, el contenido se copia para su exportación. El script generado para este módulo es "CertificadosUsuario.vbs".

```
Const Overwrite = True
Dim oFSO,objWShell,strKey,AppData
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")

CarpetaExportacion = WScript.Arguments.Item(0)

'Se comprueba si existe la carpeta para la exportación y se crea si es necesario
If Not oFSO.FolderExists(CarpetaExportacion) Then
    oFSO.CreateFolder CarpetaExportacion
End If

'Se exporta el arbol de registro de Windows relacionado con los certificados
strKey = "HKEY_CURRENT_USER\Software\Microsoft\SystemCertificates\"
regCmd = "cmd.exe /c reg export " + strKey + " " + CarpetaExportacion + "\Certificados.reg"
objWShell.run regCmd, Overwrite

'Se copian las carpetas del AppData del usuario que contiene sus certificados
AppData = objWShell.expandEnvironmentStrings("%APPDATA%")
Certificado = "\SystemCertificates"
CarpetaCertificados = AppData & "\Microsoft" & Certificado
If oFSO.FolderExists(CarpetaCertificados) Then
    oFSO.CopyFolder CarpetaCertificados, CarpetaExportacion & Certificado, Overwrite
End If

Certificado = "\Crypto"
CarpetaCertificados = AppData & "\Microsoft" & Certificado
If oFSO.FolderExists(CarpetaCertificados) Then
    oFSO.CopyFolder CarpetaCertificados, CarpetaExportacion & Certificado, Overwrite
End If

Certificado = "\Credentials"
CarpetaCertificados = AppData & "\Microsoft" & Certificado
If oFSO.FolderExists(CarpetaCertificados) Then
    oFSO.CopyFolder CarpetaCertificados, CarpetaExportacion & Certificado, Overwrite
End If

Certificado = "\Protect"
CarpetaCertificados = AppData & "\Microsoft" & Certificado
If oFSO.FolderExists(CarpetaCertificados) Then
    oFSO.CopyFolder CarpetaCertificados, CarpetaExportacion & Certificado, Overwrite
End If
```

2.2.6 Módulo SAP GUI

SAP GUI es una aplicación que se instala en una máquina cliente para tratar los datos que tiene la base de datos en el servidor. Se trata de un sistema informático alimentado de datos, donde el objetivo es poder explotarlos para convertirlos en información útil para la toma de decisiones. Es común que una mediana empresa haga servir este sistema y es seguro en el caso de grandes organizaciones como pueden ser entidades bancarias o públicas. Con este módulo del sistema se pretende poder exportar la configuración de todas las conexiones al servidor de forma automática.

En el caso de esta aplicación hay que tener en cuenta dos aspectos importantes: si la arquitectura del sistema operativo es de 32 o 64 bits y que versión del cliente SAP GUI está instalado en el equipo, ya que en el caso que sea igual o superior a 7.40 *patch* 5 el archivo de configuración de conexión cambia de un .ini a .xml. Para poder realizar este análisis se realizan diferentes consultas al registro y se ha desarrollado una función. Se ha desarrollado esta función ya que se debe consultar en diversas ocasiones.

La función (ExistKey) recibe como parámetro una clave de registro de Windows por dos motivos: para saber si existe esa clave (de esta manera se obtendrá si el SO es de 32 o 64 bits y para poder conocer la ruta del archivo de configuración), y en el caso de existir devuelve el valor de la misma. Por último, indicar que se ha utilizado la sentencia "On Error" para capturar y gestionar los errores y excepciones, ya que si se realiza una consulta a un registro inexistente se generará una excepción, de este modo se controla.

Se realiza la consulta al registro para saber si el sistema operativo es de 32 o 64 bits, ya que la ruta del ejecutable "SAPGui.exe" varía según la arquitectura. De esta forma se consigue el *path* de instalación que se necesita para obtener la versión actual del SAP GUI.

- SO x86: HKEY_LOCAL_MACHINE\SOFTWARE\SAP\SAP Shared\SAPsysdir
- SO x64: HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\SAP\SAP Shared\ SAPsysdir

El script genera un fichero con la versión de SAP GUI del equipo origen para posterior tratamiento en la importación a la máquina destino (se puede dar el caso que se requiera modificaciones si la versión de origen/destino es diferente).

Por último, Si la versión de SAP es inferior a 7.50 se han de consultar unas claves de registro y si la versión es igual o superior, se consultan otras. El objetivo es conseguir la ruta del archivo de configuración de las conexiones (se ha podido modificar mediante variable de entorno) y exportarlo:

>= 7.40 *patch 5*:

- HKEY_CURRENT_USER\Software\SAP\SAPLogon\LandscapeFilesLastUsed\LandscapeFile
- HKEY_CURRENT_USER\Software\SAP\SAPLogon\LandscapeFilesLastUsed\LandscapeFileGlobal

< 7.40 *patch 5*:

- HKEY_CURRENT_USER\Software\SAP\SAPLogon\ConfigFilesLastUsed\ConnectionConfigFile
- HKEY_CURRENT_USER\Software\SAP\SAPLogon\ConfigFilesLastUsed\TreeConfigFile

El script generado para este módulo es "SAPGUI.vbs".

```
Const Overwrite = True
Dim oFSO,objWShell,AppData
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")

CarpetaExportacion = WScript.Arguments.Item(0)

'La funcion ExistKey sirve para saber si existe una clave de registro, en el caso que exista devuelve el
valor de la misma.
Function ExistKey(Key)
    on error resume next
        ExistKey = objWShell.regread(Key)
        bFound = (err.number = 0)
    on error goto 0

    If Not bFound Then
        ExistKey = ""
    End if
End Function

'Se comprueba si existe la carpeta para la exportación y se crea si es necesario
If Not oFSO.FolderExists(CarpetaExportacion) Then
    oFSO.CreateFolder CarpetaExportacion
End If
```

```

'Se consulta el registro para saber la ruta del SAPGui.exe. Se tiene en cuenta que el SO pueda ser de 32 o 64
bits
keySAP = "HKEY_LOCAL_MACHINE\SOFTWARE\SAP\SAP Shared\SAPsysdir"
pathSAP = ExistKey (keySAP)
if pathSAP = "" Then
    keySAP = "HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\SAP\SAP Shared\SAPsysdir"
    pathSAP = ExistKey (keySAP)
End If

'Conocer la versión de SAP GUI
If oFSO.FileExists (pathSAP & "\SAPGui.exe") Then
    SAPGuiVer = oFSO.GetFileVersion(pathSAP & "\SAPGui.exe")
    Version = split(SAPGuiVer, ".")
    SAPGuiVer = Version(0)
    patch = Version(1)
End If

'Se genera fichero con la versión de SAP GUI para el posterior tratamiento
Set fileVersion = oFSO.CreateTextFile (CarpetaExportacion + "\version.txt", Overwrite)
fileVersion.WriteLine (SAPGuiVer)
fileVersion.WriteLine (patch)
fileVersion.Close

'Si la versión es 7.40 con parche 5 o superior se exportan los xml de configuracion de la conexion
If (SAPGuiVer = 7400 and patch > 4) or (SAPGuiVer >= 7500)Then
    keySAPLogon = "HKEY_CURRENT_USER\Software\SAP\SAPLogon\LandscapeFilesLastUsed\LandscapeFile"
    pathSAPLogon = ExistKey (keySAPLogon)
    If oFSO.FileExists (pathSAPLogon) Then
        oFSO.CopyFile pathSAPLogon, CarpetaExportacion & "\SAPUILandscape.xml", Overwrite
    End If
    keySAPLogon = "HKEY_CURRENT_USER\Software\SAP\SAPLogon\LandscapeFilesLastUsed\LandscapeFileGlobal"
    pathSAPLogon = ExistKey (keySAPLogon)
    If oFSO.FileExists (pathSAPLogon) Then
        oFSO.CopyFile pathSAPLogon, CarpetaExportacion & "\SAPUILandscapeGlobal.xml", Overwrite
    End If
End If

'En el caso que la version sea inferior se exporta el .ini y xml de configuracion de la conexion
Else
    keySAPLogon = "HKEY_CURRENT_USER\Software\SAP\SAPLogon\ConfigFilesLastUsed\ConnectionConfigFile"
    pathSAPLogon = ExistKey (keySAPLogon)
    If oFSO.FileExists (pathSAPLogon) Then
        oFSO.CopyFile pathSAPLogon, CarpetaExportacion & "\saplogon.ini", Overwrite
    End If
    keySAPLogon = "HKEY_CURRENT_USER\Software\SAP\SAPLogon\ConfigFilesLastUsed\TreeConfigFile"
    pathSAPLogon = ExistKey (keySAPLogon)
    If oFSO.FileExists (pathSAPLogon) Then
        oFSO.CopyFile pathSAPLogon, CarpetaExportacion & "\SapLogonTree.xml", Overwrite
    End If
End If

```

2.2.7 Módulo ODBC

El módulo ODBC realiza la exportación de los *Open DataBase Connectivity*, básicamente se trata de un estándar de acceso a las bases de datos con el objetivo de poder acceder a éstas desde cualquier aplicación, independientemente de su DBMS.

Es muy usual que las aplicaciones utilicen estas configuraciones para conectar con BBDD, por ejemplo: Access, Excel, desarrollos a medida, AutoCAD, etc. Esta configuración se encuentra en Panel de control > Herramientas administrativas > Orígenes de datos ODBC o desde el terminal con el comando `odbcad32`.

Es importante indicar que con la exportación únicamente se traspasa la configuración de la conexión; el driver o controlador utilizado debe estar instalado en ambos equipos, ya que son librerías del equipo. En el siguiente ejemplo el equipo debe tener instalado el driver de MS Access para trabajar correctamente:

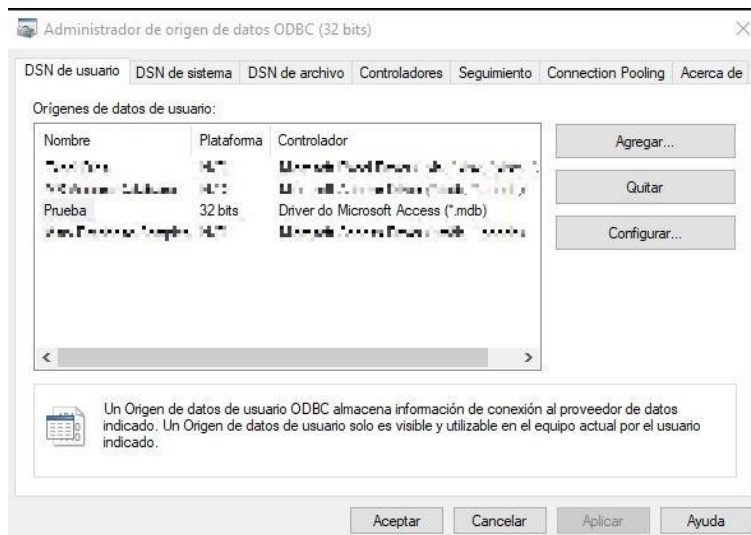


Figura 14. Administrador de ODBC

Por último, en este caso se exportarán las configuraciones tanto de usuario como de máquina, ya que se considera que en la instalación de algunas aplicaciones configura estos orígenes de datos a nivel de equipo. En este caso, no es necesario distinguir entre las diferentes versiones de sistemas operativos, ya que se almacenan en la misma ruta del registro de Windows en todas ellas. Las rutas de registro a exportar son las siguientes:

- ODBC Equipo: HKEY_LOCAL_MACHINE\Software\ODBC
- ODBC Usuario: HKEY_CURRENT_USER\Software\ODBC

El script generado para realizar las acciones de este módulo es "ODBC.vbs".

```

Const Overwrite = True
Dim oFSO,objWShell
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")

CarpetaExportacion = WScript.Arguments.Item(0)

'Se comprueba si existe la carpeta para la exportación y se crea si es necesario
If Not oFSO.FolderExists(CarpetaExportacion) Then
    oFSO.CreateFolder CarpetaExportacion
End If

'Se exporta el arbol de registro de Windows que contiene la configuración de los ODBC
'ODBC de Usuario
regCmd = "cmd.exe /c reg export HKEY_CURRENT_USER\Software\ODBC " + CarpetaExportacion +
"\SessionDataUser.reg"
objWShell.run regCmd, Overwrite

'ODBC de maquina
regCmd = "cmd.exe /c reg export HKEY_LOCAL_MACHINE\Software\ODBC " + CarpetaExportacion +
"\SessionDataMachine.reg"
objWShell.run regCmd, Overwrite

```

2.2.8 Módulo Conexiones SNA

El módulo conexiones SNA tiene como objetivo exportar la configuración de la conexión SNA (*Systems Network Architecture*) utilizada por aplicativos como AS/400 o HIS (*Host Integration Server*). Se trata de una arquitectura de red usada para conectar con *mainframe*. Para los sistemas operativos Windows existen "SNA Server" y HIS que trabaja como un *Gateway* para

realizar el enlace entre la red de mainframes en arquitectura SNA y una red TCP/IP con Windows.

Estas aplicaciones, en las máquinas cliente, emulan la conexión 3270 de IBM, que trabajan como una conexión mediante Telnet. Es muy usual encontrarse este tipo de aplicaciones en organizaciones bancarias y públicas, ya que por una parte son capaces de trabajar con una gran cantidad de datos y transacciones, y por otra se considera una arquitectura más segura que el modelo TCP/IP. Estas aplicaciones son conocidas popularmente como aplicaciones de pantalla verde.



Figura 15. Aplicativo bajo conexión SNA

En este caso, la exportación de las configuraciones de la conexión SNA se realiza mediante el registro de Windows, común a todas las versiones del Sistema Operativo. Como en el caso anterior, se realiza la exportación tanto de las configuraciones de máquina como las de usuario:

- SNA Equipo: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\SnaBase\Parameters
- SNA Usuario: HKEY_CURRENT_USER\SOFTWARE\Microsoft\SnaBase\Parameters

El script generado para realizar las acciones de este módulo es "SNA.vbs".

```
Const Overwrite = True
Dim oFSO,objWShell
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")

CarpetaExportacion = WScript.Arguments.Item(0)

'Se comprueba si existe la carpeta para la exportación y se crea si es necesario
If Not oFSO.FolderExists(CarpetaExportacion) Then
    oFSO.CreateFolder CarpetaExportacion
End If

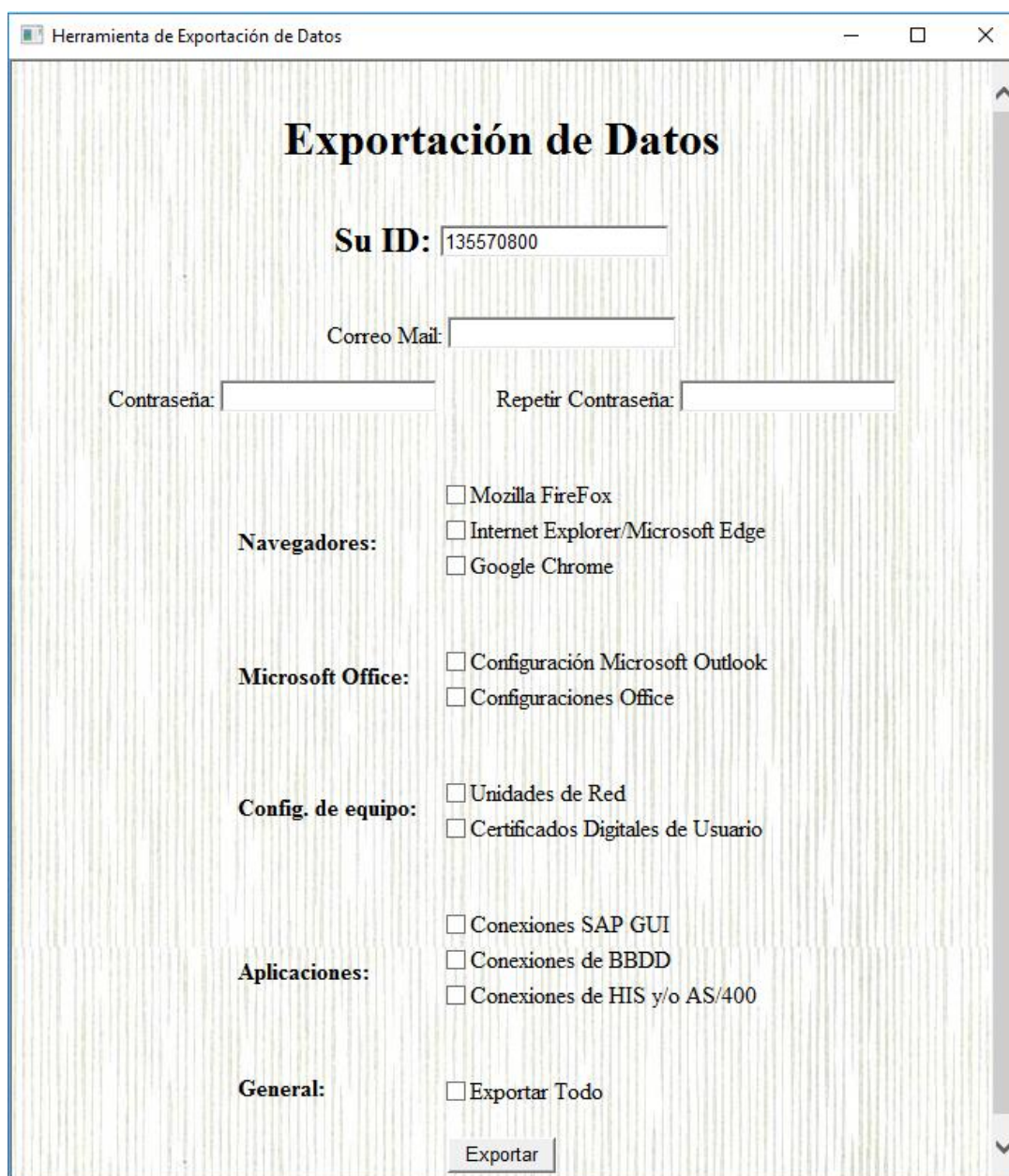
'Se exporta el arbol de registro de Windows que contiene la configuración de las conexiones SNA
'SNA de Usuario
regCmd = "cmd.exe /c reg export HKEY_CURRENT_USER\SOFTWARE\Microsoft\SnaBase\Parameters " +
CarpetaExportacion + "\SessionDataUser.reg"
objWShell.run regCmd, Overwrite

'SNA de maquina
regCmd = "cmd.exe /c reg export HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\SnaBase\Parameters " +
CarpetaExportacion + "\SessionDataMachine.reg"
objWShell.run regCmd, Overwrite
```

2.3 Diseño y desarrollo del HTA de exportación de datos

Una vez implementados los diferentes módulos de exportación de datos que se usarán durante la migración de la plataforma, el HTA (*HTML Application* o Aplicación HTML) ofrece al usuario la posibilidad de escoger los módulos que necesita, mediante un id aleatorio y una contraseña que facilita la compresión y securiza los datos a exportar y la solicitud de un correo electrónico donde se enviará el id aleatorio a cada usuario.

Esta aplicación presenta una interfaz gráfica al usuario, donde se recogen los diferentes módulos de configuración especificados y susceptibles de ser exportados. Se ha añadido la opción de "exportar todo" para facilitar el proceso. Existe la posibilidad de marcar módulos obligatorios a requerimiento de la organización.



The screenshot shows a web browser window titled "Herramienta de Exportación de Datos". The main heading is "Exportación de Datos". Below the heading, there are several input fields: "Su ID:" with the value "135570800", "Correo Mail:" (empty), "Contraseña:" (empty), and "Repetir Contraseña:" (empty). Below these are several sections of checkboxes:

- Navegadores:**
 - Mozilla FireFox
 - Internet Explorer/Microsoft Edge
 - Google Chrome
- Microsoft Office:**
 - Configuración Microsoft Outlook
 - Configuraciones Office
- Config. de equipo:**
 - Unidades de Red
 - Certificados Digitales de Usuario
- Aplicaciones:**
 - Conexiones SAP GUI
 - Conexiones de BBDD
 - Conexiones de HIS y/o AS/400
- General:**
 - Exportar Todo

At the bottom, there is an "Exportar" button.

Figura 16. HTA Exportación

2.3.1 Identificación Usuario

Para asegurar la información que se gestiona durante todo el proceso, se establece la obligatoriedad de crear un usuario y contraseña que, a posteriori, recupere dicha información. Para la creación del usuario se ha definido que se genere un código automático y aleatorio (no modificable). Se han valorado diferentes opciones como el dni, correo, código empleado..., pero no permiten asegurar la privacidad o anonimato completo del usuario y sus datos.

Este ID deberá ser recordado por el usuario para poder recuperar las configuraciones y para ello se ha incorporado un envío de este al correo que cada usuario indique en el formulario. Este correo se enviará desde la cuenta de administrador de la migración para así, por un lado, tener un registro de cada id/usuario manteniendo la privacidad de los datos al no disponer de la contraseña; y por otro lado, el usuario dispone de su id para recordarlo en cualquier momento. Al tratarse de un dato crítico, deberá ser tenido en cuenta por los responsables de la Gestión del cambio. El id será comprobado en el *share* mediante la aplicación para evitar duplicidades, ya que debe ser único. Se realiza de forma automática ya que el id aleatorio se genera dentro de un bucle y únicamente sale de él si el id generado no existe en el *path* remoto.

Asimismo, este id está relacionado con una contraseña con un mínimo de 8 caracteres, escogido por el mismo usuario y que habrá de ser recordado a la hora de la importación.

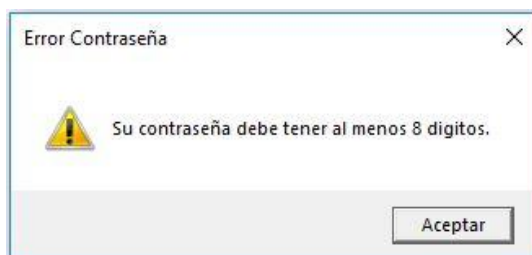


Figura 17. Error contraseña I

También se realiza una doble comprobación de contraseña, para asegurar la corrección de ésta.

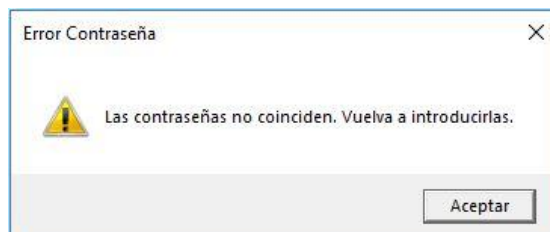


Figura 18. Error contraseña II

Esta contraseña es la utilizada para comprimir y descomprimir en fichero 7z de manera securizada los datos del usuario en el *share*.

Queda patente la importancia de estos dos datos, id y contraseña, a la hora de poder recuperar los datos del usuario, y es por ello por lo que deben ser recordados y guardados por el usuario de forma responsable

2.3.2 Gestión de los datos

Los datos que se exportan de la máquina origen han de ser tratados antes de ser guardados, para conservar la privacidad de estos, teniendo en cuenta la LOPD. Pasan un proceso de compresión de datos utilizando la aplicación *opensource* y gratuita 7zip, que tiene la funcionalidad de uso mediante línea de comandos. Esta aplicación encripta los datos de usuario utilizando la contraseña escogida por el usuario.

Al comienzo de la exportación, los datos se guardan en la carpeta temporal del usuario, por dos motivos:

- Asegurar que el usuario tenga permisos de escritura sobre ese directorio.
- Al realizarse en el perfil local del usuario, únicamente tiene acceso él.

Una vez realizada toda la exportación, los datos son encriptados en local (con 7z) para subirlos posteriormente al *share*. Al encriptarse antes de depositarlos en el repositorio, evitamos los posibles ataques de captura de datos de la red (sniffers), ya que estos ya estarán securizados. Estos datos se suben con el atributo de "oculto". Para los usuarios, el repositorio está oculto, todo y tener permisos; no es necesario que tengan mapeado el recurso. Por último, todos los datos exportados y guardados en local en el equipo de origen son borrados, tanto los encriptados como los que no lo están. La configuración del usuario en el equipo origen no se modifica, de esta manera puede seguir trabajando una vez realizada la exportación o realizar el proceso a posterior por si se ha dejado algún dato por exportar.

2.3.3 Proceso del HTA

El HTA es un acceso directo que apunta directamente al HTA del repositorio, y al tratarse de una solución transversal, en entornos con dominio, este acceso directo se copiaría en el escritorio público, de esta forma aparecerá a todos los usuarios que hagan uso de ese equipo; por otro lado, en entornos *workgroup*, la distribución del acceso directo podría realizarse mediante correo.

El proceso que ejecuta el HTA es el siguiente:

En primer lugar, se ejecuta el acceso directo y se abre HTA donde el usuario, con ID aleatorio y automatizado, una contraseña y una dirección de correo, indica los módulos a exportar (uno, varios o todos) y clicla en el botón Exportar. En caso de que el usuario no indique ningún módulo, aparece mensaje de error:

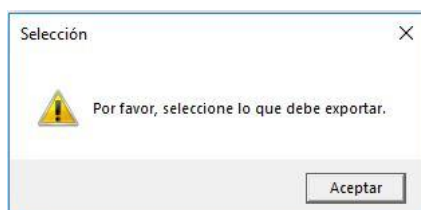


Figura 19. Error Selección

Se genera un fichero donde se indican las selecciones realizadas por el usuario y que se guarda juntamente con los datos, este fichero será utilizado en el proceso de importación.

Seguidamente, empieza el proceso de exportación de datos (ejecuta *scripts* de exportación, guarda en local, encripta, sube al repositorio y borra datos en local):

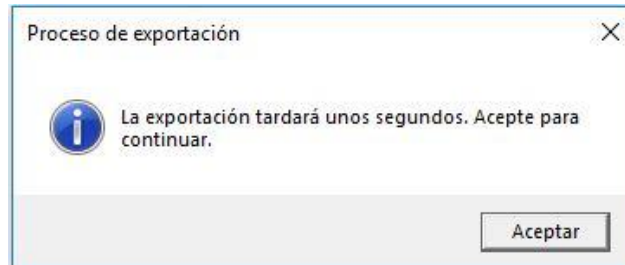


Figura 20. Inicio proceso

Durante las pruebas de validación no se han detectado errores de exportación al mantener las aplicaciones abiertas, excepto en Microsoft Edge. En el caso que el usuario marque la opción de Favoritos Internet Explorer, se valida si el proceso MicrosoftEdge está en ejecución; en caso afirmativo se le indica al usuario que cierre dicha aplicación (para tener en cuenta por Gestión del Cambio).

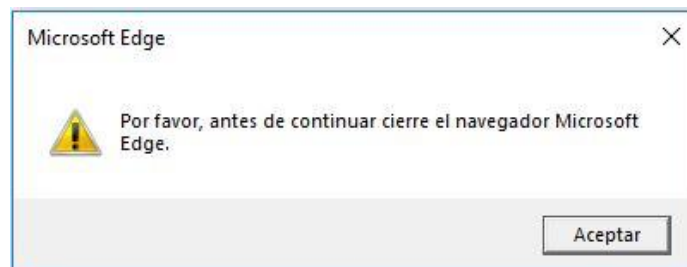


Figura 21. Error Microsoft Edge

Una vez finalizado el proceso completo, aparece un mensaje de finalización y recordatorio de ID para el usuario:

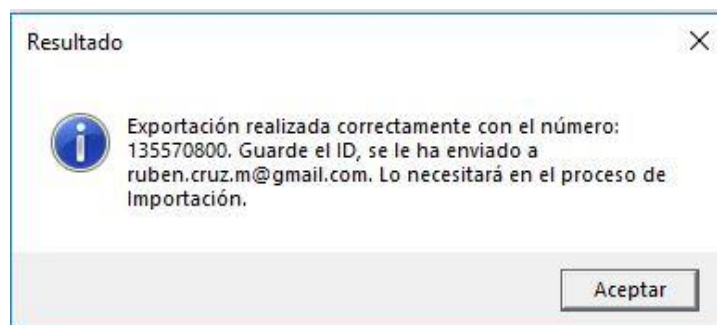


Figura 22. Exportación realizada

El usuario recibirá asimismo un correo electrónico de finalización del proceso con su id.

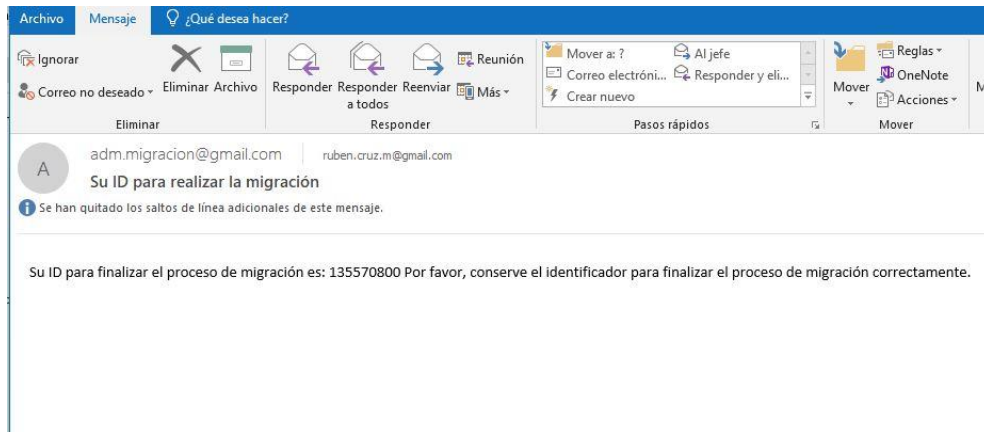


Figura 23. Correo a usuario con su ID

Para realizar este envío se ha desarrollado un script al cual se le facilitan los argumentos: buzón de usuario e id asociado. Como en este script ha de aparecer la contraseña del buzón de administrador se ha encriptado el script, pasando de extensión .vbs a .vbe. De este modo, si alguien accediera a este, no podría obtener ningún tipo de información.

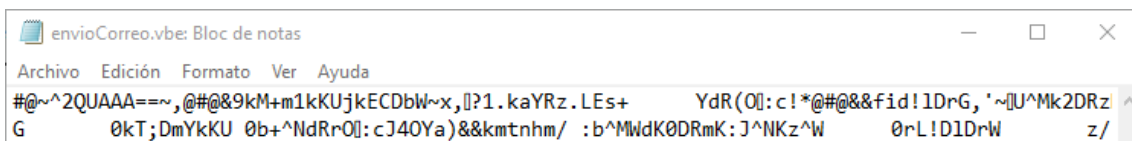


Figura 24. Contenido envioCorreo.vbe

El script sin encriptar generado es:

```

direccionUsuario = WScript.Arguments.Item(0)
IDUsuario = WScript.Arguments.Item(1)

set objCDO = createobject("cdo.message")
direccionADM = "adm.migracion@gmail.com"
psADM = "TFGracruzma"
objCDO.subject = "Su ID para realizar la migraci" + chr(243) + "n"
objCDO.from = direccionADM
objCDO.to = direccionUsuario
objCDO.textbody = "Su ID para finalizar el proceso de migraci" + chr(243) + "n es: " +
IDUsuario + vbCrLf + "Por favor, conserve el identificador para finalizar el proceso de migraci"
+ chr(243) + "n correctamente."
objCDO.configuration.fields.item("http://schemas.microsoft.com/cdo/configuration/sendusing")
= 2
objCDO.configuration.fields.item("http://schemas.microsoft.com/cdo/configuration/smtpserver")
= "smtp.gmail.com"
objCDO.configuration.fields.item("http://schemas.microsoft.com/cdo/configuration/smtpserverport")
= 465
objCDO.configuration.fields.item("http://schemas.microsoft.com/cdo/configuration/smtpauthenticate")
= 1
objCDO.configuration.fields.item("http://schemas.microsoft.com/cdo/configuration/smtpconnectionti
meout") = 30
objCDO.configuration.fields.item("http://schemas.microsoft.com/cdo/configuration/sendusername") =
direccionADM
objCDO.configuration.fields.item("http://schemas.microsoft.com/cdo/configuration/sendpassword") =
psADM
objCDO.configuration.fields.item("http://schemas.microsoft.com/cdo/configuration/smtpusessl")
= 1
objCDO.configuration.fields.update
objCDO.send

```



```

Function CheckAplicacionAbierta(aplicacion)
    Dim objWMIService, colItems, objItem

    On Error Resume Next
    CheckAplicacionAbierta = False
    Set objWMIService = GetObject("winmgmts:\\.\root\CIMV2")
    Set colItems = objWMIService.ExecQuery("Select * from Win32_Process",,48)

    For Each objItem in colItems
        If left(objItem.Name, 13) = aplicacion Then
            CheckAplicacionAbierta = True
        End If
    Next

End Function

Sub Start_Button()
    Set oFSO = CreateObject("Scripting.FileSystemObject")
    Set objWShell = CreateObject("WScript.Shell")
    Const PathRemoto = "\\192.168.1.30\Datos\Configs"
    Const PathScripts = "\\192.168.1.30\Datos\Export"
    Const zip = "\\192.168.1.30\Datos\Export\7za"
    temp = objWShell.expandEnvironmentStrings("%temp%")
    'Se controla que la contraseÃa tenga como minimo 8 digitos
    If Len(Pass1.Value) < 8 Then
        x=MsgBox ("Su contraseÃa debe tener al menos 8 digitos.", 48, "Error ContraseÃa")
    Else
        'Se comprueba que las dos contraseÃas sean iguales. Si no lo son, se borran
        If Pass1.Value <> Pass2.Value Then
            x=MsgBox ("Las contraseÃas no coinciden. Vuelva a introducirlas.", 48, "Error ContraseÃa")
            Pass1.Value = ""
            Pass2.Value = ""
        Else
            If chkIexplorer.Checked or chkAll.Checked Then
                If CheckAplicacionAbierta("MicrosoftEdge") Then
                    x=MsgBox ("Por favor, antes de continuar cierre el navegador Microsoft Edge.", 48,
"Microsoft Edge")
                End If
            End If
            If (chkFirefox.Checked or chkIexplorer.Checked or chkChrome.checked or chkOutlook.Checked
or chkOffice.Checked or chkUnidades.Checked or chkCertificados.Checked or chkSAP.Checked or
chkODBC.Checked or chkSNA.Checked or chkAll.Checked) Then
                x=Msgbox ("La exportaciÃn tardarÃ; unos segundos. Acepte para continuar.", 64, "Proceso
de exportaciÃn")
            Else
                x=MsgBox ("Por favor, seleccione lo que debe exportar.", 48, "SelecciÃn")
                Exit Sub
            End If

            'Se crea la carpeta de exportacion con el ID del usuario
            CarpetaExportacion = temp & "\" & ID.value
            If Not oFSO.FolderExists(CarpetaExportacion) Then
                oFSO.CreateFolder CarpetaExportacion
            End If

            'Crear archivo para indicar las selecciones de usuario
            Set FileSelections = oFSO.CreateTextFile (CarpetaExportacion + "\Seleccion.txt", Overwrite)

            'Se comprueba si se han seleccionado cada uno de los modulos.
            'En caso afirmativo se exportan los seleccionados
            If chkFirefox.Checked or chkAll.Checked Then
                cmd = "cmd.exe /c wscript.exe " + PathScripts + "\Firefox.vbs " + CarpetaExportacion +
"\Firefox"
                objWShell.run cmd, 0, True
                FileSelections.WriteLine ("Firefox")
            End If
            If chkIexplorer.Checked or chkAll.Checked Then
                cmd = "cmd.exe /c wscript.exe " + PathScripts + "\IE.vbs " + CarpetaExportacion +
"\Iexplorer"
                objWShell.run cmd, 0, True
                FileSelections.WriteLine ("Iexplorer")
            End If
            If chkChrome.checked or chkAll.Checked Then
                cmd = "cmd.exe /c wscript.exe " + PathScripts + "\Chrome.vbs " + CarpetaExportacion +
"\Chrome"
                objWShell.run cmd, 0, True
                FileSelections.WriteLine ("Chrome")
            End If
            If chkOutlook.Checked or chkAll.Checked Then
                cmd = "cmd.exe /c wscript.exe " + PathScripts + "\Outlook.vbs " + CarpetaExportacion +
"\Outlook"
                objWShell.run cmd, 0, True
                FileSelections.WriteLine ("Outlook")
            End If
        End If
    End Sub
End Sub

```

```

        If chkOffice.Checked or chkAll.Checked Then
            cmd = "cmd.exe /c wscript.exe " + PathScripts + "\Office.vbs " + CarpetaExportacion +
"\Office"
            objWShell.run cmd, 0, True
            FileSelections.WriteLine ("Office")
        End If
        If chkUnidades.Checked or chkAll.Checked Then
            cmd = "cmd.exe /c wscript.exe " + PathScripts + "\Unidades.vbs " + CarpetaExportacion
+ "\Unidades"
            objWShell.run cmd, 0, True
            FileSelections.WriteLine ("Unidades")
        End If
        If chkCertificados.Checked or chkAll.Checked Then
            cmd = "cmd.exe /c wscript.exe " + PathScripts + "\CertificadosUsuario.vbs " +
CarpetaExportacion + "\CertificadosUsuario"
            objWShell.run cmd, 0, True
            FileSelections.WriteLine ("CertificadosUsuario")
        End If
        If chkSAP.Checked or chkAll.Checked Then
            cmd = "cmd.exe /c wscript.exe " + PathScripts + "\SAPGUI.vbs " + CarpetaExportacion +
"\SAPGUI"
            objWShell.run cmd, 0, True
            FileSelections.WriteLine ("SAPGUI")
        End If
        If chkODBC.Checked or chkAll.Checked Then
            cmd = "cmd.exe /c wscript.exe " + PathScripts + "\odbc.vbs " + CarpetaExportacion +
"\ODBC"
            objWShell.run cmd, 0, True
            FileSelections.WriteLine ("odbc")
        End If
        If chkSNA.Checked or chkAll.Checked Then
            cmd = "cmd.exe /c wscript.exe " + PathScripts + "\sna.vbs " + CarpetaExportacion +
"\SNA"
            objWShell.run cmd, 0, True
            FileSelections.WriteLine ("sna")
        End If

        FileSelections.Close

        'Se zipean los archivos de configuraci3n con el ID y la contrase1a facilitada por el
usuario
        cmd = "cmd.exe /c " & zip & " a " & CarpetaExportacion & ".7z " & CarpetaExportacion & " -
p" & Pass1.Value
        objWShell.run cmd, 0, True

        If oFSO.FileExists(CarpetaExportacion & ".7z") Then
            oFSO.MoveFile CarpetaExportacion & ".7z", PathRemoto & "\"
            set objFile = oFSO.GetFile(PathRemoto & "\" & ID.value & ".7z")
            If Not objFile.Attributes AND 2 then
                objFile.Attributes = objFile.Attributes + 2
            End If
            cmd = "cmd.exe /c rmdir /s /q "" " & CarpetaExportacion & "" "
            objWShell.run cmd, 0, True
            cmd = "cmd.exe /c wscript.exe " + PathScripts + "\envioCorreo.vbe " +
CorreoUsuario.Value + " " + ID.value
            objWShell.run cmd, 0, True
            x=MsgBox ("Exportaci3n realizada correctamente con el n1mero: " & IDRandom.value & ".
Guarde el ID, se le ha enviado a " & CorreoUsuario.Value & ". Lo necesitar1 en el proceso de
Importaci3n.", 64, "Resultado")
        End If

    End If
End If
End Sub
</script>

```

2.4 Dise1o y desarrollo del HTA de importaci3n de datos

En este apartado se desarrolla la aplicaci3n HTML que ejecuta los scripts de retorno de los datos de los usuarios que han sido capturados en el apartado anterior. Utilizando el ID de usuario aleatorio facilitado en el HTA de exportaci3n, as1 como, la contrase1a requerida, los ficheros de datos son descifrados y descomprimidos en la m1quina destino del usuario.

El entorno gráfico que se muestra al usuario es similar al descrito en el apartado anterior, en el cual se visualiza las casillas de ID y contraseña que han de rellenarse para que comience el proceso de importación de datos.

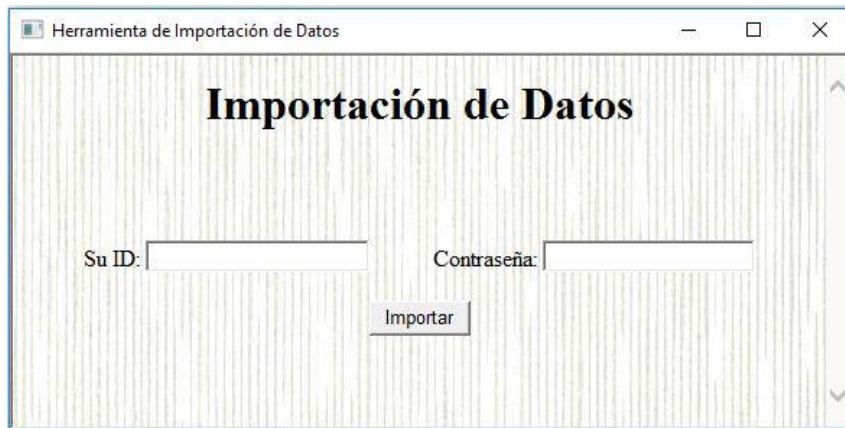


Figura 25. HTA Importación

2.4.1 Identificación usuario

Para poder recuperar la información guardada y encriptada en el *share*, el usuario debe identificarse de manera unívoca en el HTA, indicando la clave ID que le ha sido facilitada durante la exportación y la contraseña que escogió a tal efecto. Se realiza una comprobación de la validez del ID en el repositorio (si existe):

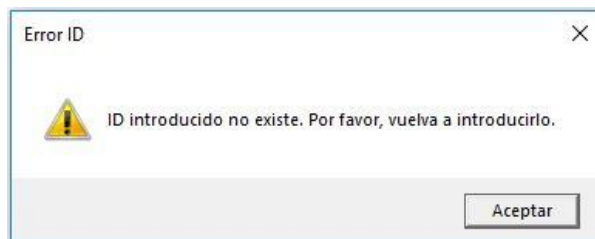


Figura 26. Error ID

Y se comprueba si la contraseña cumple los requisitos (la validez de este *password* se verifica a posteriori al descomprimir los datos recuperados, que se explica en el apartado siguiente):

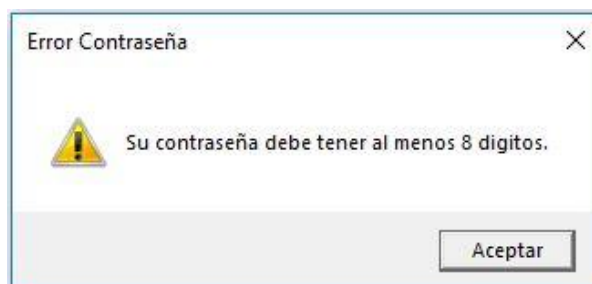


Figura 27. Error contraseña III

2.4.2 Gestión de datos

Una vez se ha accedido al repositorio, se descargan los datos relacionados al ID indicado y se borra la información del *share*. A continuación, se ha de verificar que la contraseña del proceso de descryptado es la correcta. Para ello, se ejecuta un script de validación:

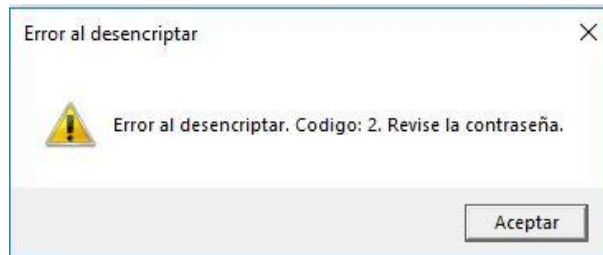


Figura 28. Error contraseña IV

En caso de que la contraseña sea incorrecta, se realiza un borrado de los datos descargados en el directorio temporal del usuario como método de seguridad y se vuelven a cargar en el *share*. En otras palabras, se realiza un *rollback* del proceso.

Para finalizar, se resetea el HTA para indicar nuevamente los datos de recuperación.

Si el ID y contraseña son correctos el proceso continúa y se importan los datos, borrando, una vez finalizado, los datos descryptados y dejando únicamente el fichero 7zip codificado en local por si han de ser recuperados en cualquier momento, ya que no existen en el repositorio.

2.4.3 Proceso del HTA

La distribución del HTA a los usuarios será la misma que el HTA de exportación: escritorios públicos o distribución por correo.

El proceso que ejecuta en HTA de importación es el siguiente:

Se ejecuta el acceso directo y se abre HTA donde el usuario ha de indicar su ID y la contraseña creados durante el proceso de captación de datos. Si todo el proceso es correcto (id y contraseña válidos) aparece mensaje de proceso en ejecución:

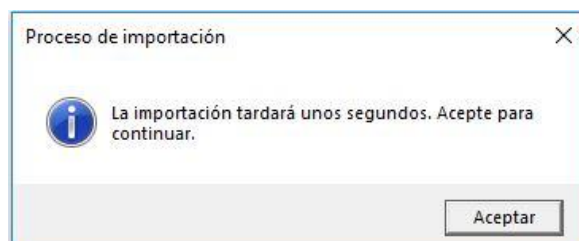


Figura 29. Inicio proceso II

vbScript

```
<script language="VBScript">

Sub Window_OnLoad
window.resizeto 600,300
End Sub

Function CheckAplicacionAbierta(aplicacion)
Dim objWMIService, colItems, objItem
On Error Resume Next

CheckAplicacionAbierta = False
Set objWMIService = GetObject("winmgmts:\\.\root\CIMV2")
Set colItems = objWMIService.ExecQuery("Select * from Win32_Process",,48)

For Each objItem in colItems
If left(objItem.Name, 13) = aplicacion Then
CheckAplicacionAbierta = True
End If
Next
End Function

Sub Start_Button()
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = CreateObject("WScript.Shell")
temp = objWShell.expandEnvironmentStrings("%temp%")
Const PathRemoto = "\\192.168.1.30\Datos\Configs"
Const PathScripts = "\\192.168.1.30\Datos\Import"
Const zip = "\\192.168.1.30\Datos\Import\7za"

'Se busca si existe el fichero exportado por el ID indicado
If Not oFSO.FileExists (PathRemoto & "\" & ID.value & ".7z") Then
x=MsgBox ("ID introducido no existe. Por favor, vuelva a introducirlo.", 48, "Error ID")
ID.value = ""
Pass1.Value = ""
Exit Sub
End If

'Se controla que la contraseña tenga como minimo 8 digitos
If Len(Pass1.Value) < 8 Then
x=MsgBox ("Su contraseña debe tener al menos 8 digitos.", 48, "Error Contraseña")
Pass1.Value = ""
Exit Sub
End If
x=Msgbox ("La importación tardará unos segundos. Acepte para continuar.", 64, "Proceso de importación")
oFSO.MoveFile PathRemoto & "\" & ID.value & ".7z", temp & "\"
CarpetaImportacion = temp & "\" & ID.value
'Se zipean los archivos de configuración con el ID y la contraseña facilitada por el usuario
cmd = "cmd.exe /c " & zip & " x " & CarpetaImportacion & ".7z -o" & temp & " -p" & Pass1.Value
nReturnErrorLevel = objWShell.run (cmd, 0, True)

If nReturnErrorLevel <> 0 Then
oFSO.MoveFile CarpetaImportacion & ".7z", PathRemoto & "\"
cmd = "cmd.exe /c rmdir /s /q "" & CarpetaImportacion & "" "
objWShell.run cmd, 0, True
x=MsgBox ("Error al descomprimir. Código: " & nReturnErrorLevel & ". Revise la contraseña.", 48, "Error al descomprimir")
ID.value = ""
Pass1.Value = ""
Exit Sub
End If

'Se lee el fichero seleccion para saber las importaciones que se han de realizar
If oFSO.FileExists(CarpetaImportacion & "\Seleccion.txt") Then
Set FileSelections = oFSO.OpenTextFile(CarpetaImportacion & "\Seleccion.txt", 1, False)
Do
Seleccion = Trim(FileSelections.ReadLine)
If Seleccion = "explorer" Then
If CheckAplicacionAbierta("MicrosoftEdge") Then
x=MsgBox ("Por favor, cierre el navegador Microsoft Edge y vuelva a seleccionar Importar" , 48, "Microsoft Edge")
FileSelections.Close
oFSO.MoveFile CarpetaImportacion & ".7z", PathRemoto & "\"
cmd = "cmd.exe /c rmdir /s /q "" & CarpetaImportacion & "" "
objWShell.run cmd, 0, True
Exit Sub
End If
End If
cmd = "cmd.exe /c wscript.exe " + PathScripts + "\" + Seleccion + "_import.vbs " + CarpetaImportacion + "\" + Seleccion
objWShell.run cmd, 0, True
Loop While Not FileSelections.AtEndOfStream
End If
End Function
End Sub
End Script
```

```

FileSelections.Close
cmd = "cmd.exe /c rmdir /s /q "" & CarpetaImportacion & "" "
objWShell.run cmd, 0, True
x=MsgBox ("Importación realizada correctamente.", 64, "Importació Finalizada")
End If
End Sub
</script>

```

2.5 Diseño y desarrollo de Módulos importación de datos

En este punto, se definen los módulos de importación de datos de los usuarios desarrollados para finalizar el proceso de migración. Como en el caso de los módulos de exportación, en cada uno de ellos se realiza un análisis de los datos necesarios y las modificaciones requeridas, la propia implementación del script de importación y la validación en el entorno de pruebas construido.

Por los motivos explicados con anterioridad y por homogeneidad, los scripts están desarrollados en lenguaje VBScript. Una vez claros estos conceptos se pasa a describir cada uno de los guiones implantados. A tener en cuenta que, todos los scripts de importación de los diferentes módulos son llamados por la HTA pasándoles como argumento el directorio donde se encuentran los datos a importar.

2.5.1 Módulo Favoritos Navegadores

Una vez recogidos todos los datos de los favoritos de los diferentes navegadores, mediante este script se procede a la importación de éstos a la máquina destino.

Internet Explorer

Al igual que en el módulo de exportación, se realiza el mismo proceso de importación de favoritos pero a la inversa: utilizando la misma variable de entorno %userprofile%, teniendo en cuenta el idioma del sistema operativo y en el caso de ser Windows 10 y disponer de Microsoft Edge, también se importan.

La finalidad del proceso es el de copiar la ruta de los favoritos en la localización correcta de destino. El script generado para este módulo es "Iexplorer_import.vbs"

```

Const Overwrite = True
Dim oFSO,objWShell
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")

CarpetaImportacion = WScript.Arguments.Item(0)
IE = "\IE"
Edge = "\Edge"

'Se comprueba si existe la carpeta para la importación, si existe comienza el proceso.
If oFSO.FolderExists(CarpetaImportacion & IE) Then
    Dim CarpetaUsuario
    CarpetaUsuario = objWShell.expandEnvironmentStrings("%userprofile%")
    'Se importa el contenido, teniendo en cuenta el idioma.
    CarpetaUsuarioIE = CarpetaUsuario & "\favoritos"
    If oFSO.FolderExists(CarpetaUsuarioIE) Then
        oFSO.CopyFolder CarpetaImportacion & IE, CarpetaUsuarioIE, Overwrite
    End If

```

```

CarpetaUsuarioIE = CarpetaUsuario & "\favorites"
If oFSO.FolderExists(CarpetaUsuarioIE) Then
    oFSO.CopyFolder CarpetaImportacion & IE, CarpetaUsuarioIE, Overwrite
End If

'Si se exportaron los favoritos de Edge, se importan.
If oFSO.FolderExists(CarpetaImportacion & Edge) Then
    Dim LocalData
    LocalData = objWShell.expandEnvironmentStrings("%LocalAppData%")
    CarpetaEdge = LocalData &
"\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC\MicrosoftEdge\User\Default"
    Store = "\DataStore"
    'Se comprueba que existe el directorio y se importa
    If oFSO.FolderExists(CarpetaEdge & Store) Then
        oFSO.CopyFolder CarpetaImportacion & Edge & Store, CarpetaEdge & Store, Overwrite
    End If
End If
End If

```

Mozilla Firefox

Durante la exportación de los favoritos de Firefox, se generan tres ficheros: places.sqlite (fichero que contiene los favoritos), profiles.ini (fichero con el perfil de Firefox) y path.txt (contiene la ruta o estructura de carpetas donde se ha de depositar el fichero places.sqlite). Por tanto, el script lee en primer lugar el fichero path.txt y comprueba si existen cada una de las carpetas de la ruta, y si no existen, las crea. Una vez creada esta ruta, se copia el fichero places.sqlite.

Por otro lado, el archivo profiles.ini se copia en la ruta correspondiente, y de esta forma se fuerza a la aplicación a utilizar el perfil indicado con los favoritos adecuados.

El script generado para este módulo es "Firefox_import.vbs".

```

Dim oFSO,objWShell
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = CreateObject("WScript.Shell")
Const Overwrite = True

CarpetaImportacion = WScript.Arguments.Item(0)

FireFoxProfileFileName = "\profiles.ini"
FireFoxProfilePath = "\Mozilla\Firefox"

'Nombre documento favoritos
Dim places, path
places = "\places.sqlite"
path = "\path.txt"

'Se comprueba si existe la carpeta para la importacion, si existe comienza el proceso.
If oFSO.folderExists(CarpetaImportacion) Then
    'Si existe el archivo .txt comienza a leerlo para extraer la ruta del perfil
    If oFSO.FileExists(CarpetaImportacion & path) Then
        'Se extrae la ruta donde se han de importar los datos
        Set objIniFile = oFSO.OpenTextFile(CarpetaImportacion & path, 1, False)
        ruta = Trim(objIniFile.ReadLine)
        objIniFile.Close
        AppData = objWShell.ExpandEnvironmentStrings("%APPDATA%")
        ruta = AppData & ruta

        'Se revisa que las carpetas de la ruta existan, en caso negativo se crean
        SubRutas = Split(Ruta,"",-1,1)
        aDIR = ""
        For Each sRutas in SubRutas
            aDIR = aDIR & sRutas & "\"
            If not oFSO.FolderExists(aDIR) Then
                oFSO.CreateFolder aDIR
            End If
        Next
        'Se copia el archivo que contiene los Favoritos
        If oFSO.FileExists(CarpetaImportacion & places) Then
            oFSO.CopyFile CarpetaImportacion & places, ruta & places, Overwrite
        End If
    End If
End If

```



```

        'Se copia el fichero .ini para forzar que vaya a buscar esos favoritos.
        If oFSO.FileExists(CarpetaImportacion & FireFoxProfileFileName) Then
            oFSO.CopyFile CarpetaImportacion & FireFoxProfileFileName, AppData & FireFoxProfilePath
            & FireFoxProfileFileName, Overwrite
        End If
    End If
End If

```

Google Chrome

Como en el caso de la exportación, se realiza una consulta wmi para obtener la versión del sistema operativo, ya que en el caso de ser Windows XP la ruta es diferente al resto de sistemas. Una vez obtenidas las rutas, se realiza la copia del fichero *bookmarks*, que contiene los favoritos de Chrome.

El script generado para este módulo es "Chrome_import.vbs"

```

Const Overwrite = True
Dim oFSO,objWShell
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")

CarpetaImportacion = WScript.Arguments.Item(0)

'Se comprueba si existe la carpeta para la importacion, si existe comienza el proceso.
If oFSO.FolderExists(CarpetaImportacion) Then
    'Se realiza consulta WMI para obtener la versión del SO
    Set SystemSet = GetObject("winmgmts:").InstancesOf ("Win32_OperatingSystem")
    for each System in SystemSet
        VersionSO = System.Version
    Next
    VersionSO = left (VersionSO, 3)
    Perfil = "\Default"
    'En el caso que sea Windows XP apuntamos a una ruta diferente que con el resto de SO.
    If VersionSO = "5.1" or VersionSO = "5.2" then
        Dim CarpetaUsuario
        CarpetaUsuario = objWShell.expandEnvironmentStrings("%userprofile%")
        CarpetaChrome = CarpetaUsuario & "\Config~1\Datos de programa\Google\Chrome\User Data"
        'Si existe la ruta se copia el Bookmarks
        If oFSO.FolderExists(CarpetaChrome) Then
            If Not oFSO.FolderExists(CarpetaChrome & Perfil) Then
                oFSO.CreateFolder CarpetaChrome & Perfil
            End If
            oFSO.CopyFile CarpetaImportacion & "\Bookmarks", CarpetaChrome & Perfil & "\Bookmarks",
Overwrite
        End If
    Else
        Dim AppData
        AppData = objWShell.expandEnvironmentStrings("%LOCALAPPDATA%")
        CarpetaChrome = AppData & "\Google\Chrome\User Data"
        'Si existe la ruta se copia el Bookmarks
        If oFSO.FolderExists(CarpetaChrome) Then
            If Not oFSO.FolderExists(CarpetaChrome & Perfil) Then
                oFSO.CreateFolder CarpetaChrome & Perfil
            End If
            oFSO.CopyFile CarpetaImportacion & "\Bookmarks", CarpetaChrome & Perfil & "\Bookmarks",
Overwrite
        End If
    End If
End If

```

2.5.2 Módulo Configuración de correo Outlook

En el script de correo Outlook se utilizan dos procedimientos Subrutina (SUB), el ReplaceReg y OSTReg. El primero, ReplaceReg, recibe 4 parámetros: 2 archivos .reg (antiguo y nuevo) y 2 *strings* o cadenas (antigua y nueva). Es decir, lo que realiza el procedimiento es recibir un .reg y en este archivo debe encontrar la antigua cadena y reemplazarla por la nueva. Por último, se guarda en el nuevo fichero .reg.

Básicamente se utiliza para editar las cadenas de registro de Windows, modificando por las adecuadas según la versión. El segundo, OSTReg, recibe 1 parámetro: el registro a editar. Lo que realiza este procedimiento es modificar dos claves de registro que hacen referencia a la ubicación del OST del correo de origen. En concreto se editan el registro 001f6610 (se elimina) y el 00036601 (se modifica el valor de la clave). Esto debe realizarse para que a posteriori de la importación no vaya a buscar el OST a la ruta del equipo origen, ya que esta, con toda probabilidad, cambiará.

El funcionamiento del script es el siguiente:

Al inicio, consulta la versión de Outlook del equipo destino. Después importa las firmas y los diseños de fondo, y como siempre teniendo en cuenta el idioma del sistema operativo. A continuación, copia todos los ficheros nk2 que se encuentran en la ruta de importación en el *path* adecuado del equipo destino.

A continuación, se comprueba si la versión del Outlook del equipo destino es 2016. Eso se debe a que, en esta versión, el correo Exchange únicamente se puede configurar mediante *autodiscover*. Es decir, no se puede migrar mediante la exportación/importación de los registros, ya que ni siquiera es posible configurarlo manualmente [6][7][8][9]. En el caso que sea 2016, se extrae de la exportación del registro el nombre del perfil de Outlook del equipo origen. Esto es para forzar que la configuración del perfil de correo que se cree sea el mismo que en origen, ya que será necesario que se trate del mismo nombre para poder importar el fichero nk2.

Con este nombre conseguido, se llevan a cabo las siguientes operaciones para preparar el entorno y que el usuario no tenga posibilidad de error:

- Se genera el nombre del perfil a nivel de registro donde irán los *settings* una vez el buzón esté configurado:
HKEY_CURRENT_USER\Software\Microsoft\Office\16.0\Outlook\Profiles\ProfileName
- Se fuerza que este perfil creado sea por defecto:
HKEY_CURRENT_USER\Software\Microsoft\Office\16.0\Outlook\DefaultProfile
- Se impide que al usuario le aparezca la ventana de selección de perfil de correo:
HKEY_CURRENT_USER\Software\Microsoft\Exchange\Client\Options\PickLogonProfile

Posteriormente, se revisa si Outlook es de 32 o 64 bits con el objetivo de conseguir la ruta del ejecutable. En este momento, se muestra un mensaje al usuario:

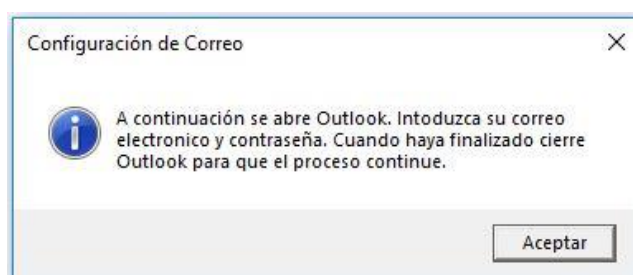


Figura 32. Apertura Outlook

Y se ejecuta el Outlook parametrizado, donde el usuario deberá indicar su cuenta de correo y contraseña.



Figura 33. Cuenta correo usuario

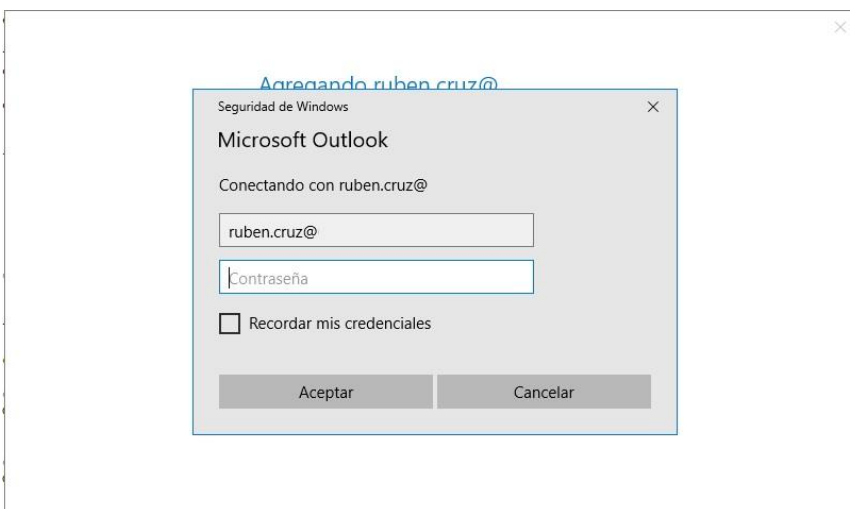


Figura 34. Contraseña correo usuario

En referencia a la versión Outlook 2016, importante para el usuario cerrar el Outlook tal y como aparece en el mensaje, para continuar con la ejecución correcta del script. En el caso que el usuario no cierre Outlook y para evitar que se queden configuraciones por importar, en el desarrollo de la solución se fuerza a que este módulo sea el último en ejecutarse. De esta forma, se asegura que se realizan todas las importaciones.

En caso de no tratarse de versión 2016, se lee el fichero versión.txt generado en la exportación, para obtener la versión de Outlook de equipo origen. Esto se hace para comparar versiones origen y destino, y si fuese necesario, llamar a ReplaceReg. Independientemente del resultado de la comparación, se llamará a la función OSTReg.

Para finalizar, se importa el árbol de las claves de registro común a todas las versiones.

El script generado para este módulo es "Outlook_import.vbs".

```
Const Overwrite = True
Dim oFSO,objWShell,strKey,NumVersion, fileVersion
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")

'Se le facilita un archivo .reg de inicio y otro final, y una cadena que ha de substituir por otra.
Sub ReplaceReg (OldReg, NewReg, OldKey, NewKey)
  Set InPutfile = oFSO.OpenTextFile(OldReg, 1, False, -1)
  ContentFile = InPutfile.ReadAll
  InPutfile.Close

  oFSO.CreateTextFile NewReg
  NewContent = Replace(ContentFile, OldKey, NewKey)

  set OutPutfile = oFSO.OpenTextFile (NewReg,2,False, -1)
  OutPutfile.write NewContent
  OutPutfile.close
End Sub

'Edita los registros que hacen referencia al OST
Sub OSTReg (RegeditOST)
  registro = ""001f6610""
  Set InPutfile = oFSO.OpenTextFile(RegeditOST, 1, False, -1)

  Do While Not InPutfile.AtEndOfStream
    Caracter = True
    Line = InPutfile.ReadLine
    If Instr(Line, registro) = 0 Then
      Content = Content & Line & vbCrLf
    Else 'Lo ha encontrado
      Do While Caracter
        Line = InPutfile.ReadLine
        If Instr(Line, "\") = 0 Then
          Caracter = False
        End If
      Loop
    End If
  Loop
  InPutfile.Close

  registro = ""00036601""
  NewContent = Replace(Content, registro & "=hex:84,01,00,00", registro & "=hex:04,10,00,00")

  Set OutPutfile = oFSO.OpenTextFile(RegeditOST, 2, False, -1)
  OutPutfile.write NewContent
  OutPutfile.close
End Sub

CarpetaImportacion = WScript.Arguments.Item(0)
'Se consulta la versión de Outlook
strKey = "HKCR\Outlook.Application\CurVer\"
NumVersion = objWShell.RegRead(strKey)
NumVersion = right (NumVersion, 2)

'Se importan las firmas, teniendo en cuenta el idioma. Si existe el directorio, copia el contenido
AppData = objWShell.expandEnvironmentStrings("%APPDATA%")
Firma = "\Signatures"
CarpetaFirma = AppData & "\Microsoft" & Firma
If oFSO.FolderExists(CarpetaImportacion & Firma) Then
  If Not oFSO.FolderExists(CarpetaFirma) Then
    oFSO.CreateFolder CarpetaFirma
    oFSO.CopyFolder CarpetaImportacion & Firma, CarpetaFirma, Overwrite
  End If
If Not oFSO.FolderExists(AppData & "\Microsoft\Firmas") Then
  oFSO.CreateFolder AppData & "\Microsoft\Firmas"
  oFSO.CopyFolder CarpetaImportacion & Firma, AppData & "\Microsoft\Firmas", Overwrite
End If
End If

'Se importan los diseños de fondo, teniendo en cuenta el idioma. Si existe el directorio, copia el contenido
Stationery = "\Stationery"
CarpetaStationery = AppData & "\Microsoft" & Stationery
If oFSO.FolderExists(CarpetaImportacion & Stationery) Then
  If oFSO.FolderExists(CarpetaStationery) Then
    oFSO.CopyFolder CarpetaImportacion & Stationery, CarpetaStationery, Overwrite
  End If
  If oFSO.FolderExists(AppData & "\Microsoft\diseños de fondo") Then
    oFSO.CopyFolder CarpetaImportacion & Stationery, AppData & "\Microsoft\diseños de fondo",
Overwrite
  End If
End If
```

```

'nk2
'Si existe el directorio y ficheros .nk2 se copian
CarpetaNk2 = AppData & "\Microsoft\Outlook"
Set pathNk2 = oFSO.GetFolder(CarpetaImportacion).Files
For each file in pathNk2
    If Mid(file.NAME,len(file.NAME)-3,4) = ".nk2" Then
        oFSO.CopyFile CarpetaImportacion + "\" + file.NAME, CarpetaNk2 + "\" + file.NAME, Overwrite
    End If
Next

'Se comprueba si existe la carpeta para la importacion, si existe comienza el proceso.
If oFSO.FolderExists(CarpetaImportacion) Then
    If NumVersion = 16 Then
        Set InPutfile = oFSO.OpenTextFile(CarpetaImportacion + "\SessionData2.reg", 1, False, -1)
        Profile = False
        Do While (Not InPutfile.AtEndOfStream) and (Profile = False)
            Line = InPutfile.ReadLine
            If Instr(Line, "Profiles\") <> 0 Then
                posicion = Instr(1, Line, "Profiles\")
                numCaracteres = len(Line)
                posicion = posicion + 8
                ProfileName = Right(Line, numCaracteres - posicion)
                ProfileName = left(ProfileName,len(ProfileName)-1)
                Profile = True
            End If
        Loop
        InPutfile.Close
        regCmd = "cmd.exe /c reg add HKEY_CURRENT_USER\Software\Microsoft\Office\16.0\Outlook\Profiles\"
+ ProfileName
        objWShell.run regCmd, Overwrite
        DefaultProfile = "HKEY_CURRENT_USER\Software\Microsoft\Office\16.0\Outlook\DefaultProfile"
        objWShell.RegWrite DefaultProfile,ProfileName,"REG_SZ"
        DontAskProfile = "HKEY_CURRENT_USER\Software\Microsoft\Exchange\Client\Options\PickLogonProfile"
        objWShell.RegWrite DontAskProfile, "0", "REG_SZ"

        If oFSO.FileExists ("C:\Program Files\Microsoft Office\root\Office16\OUTLOOK.EXE") Then
            ejecutable = "C:\Program Files\Microsoft Office\root\Office16\OUTLOOK.EXE"
        End If

        If oFSO.FileExists ("C:\Program Files (x86)\Microsoft Office\root\Office16\OUTLOOK.EXE") Then
            ejecutable = "C:\Program Files (x86)\Microsoft Office\root\Office16\OUTLOOK.EXE"
        End If

        cmd = "cmd.exe /c "" + ejecutable + """" + " /importnk2"
        wscript.echo cmd
        x=MsgBox ("A continuaci" + chr(243) + "n se abre Outlook. Intoduzca su correo electronico y
contrase" + chr(241) + "a. Cuando haya finalizado cierre Outlook para que el proceso continue.", 64,
"Configuraci" + chr(243) + "n de Correo")
        objWShell.Run cmd, 0, True

    Else
        'Si existe el archivo .txt lo lee para extraer la version de Office del origen
        Set OfficeFile = oFSO.OpenTextFile(CarpetaImportacion + "\version.txt", 1, False)
        OldNumVersion = Trim(OfficeFile.ReadLine)
        OfficeFile.Close

        reg = CarpetaImportacion + "\SessionData2.reg"

        If NumVersion <> OldNumVersion Then
            'Dependiendo de la versión de Office, se exporta un arbol de claves o otro
            If Not (OldNumVersion = 16 or OldNumVersion = 15) Then
                If NumVersion = 16 or NumVersion = 15 Then
                    OldRegValue = "HKEY_CURRENT_USER\Software\Microsoft\Windows
NT\CurrentVersion\Windows Messaging Subsystem\Profiles"
                    NewRegValue = "HKEY_CURRENT_USER\Software\Microsoft\Office\" + NumVersion +
".0\Outlook\Profiles"

                    If oFSO.FileExists(reg) Then
                        sNewFileName = CarpetaImportacion + "\NewSessionData2.reg"
                        ReplaceReg reg, sNewFileName, OldRegValue, NewRegValue
                        'Se tratan los registros de OST
                        OSTReg sNewFileName
                        'Importar NewSessionData2.reg
                        regCmd = "cmd.exe /c reg import " + sNewFileName
                        objWShell.run regCmd, Overwrite
                    End If
                Else
                    'Importar registro sin tratar ya que tanto version origen como destino son
inferiores a la v.15
                    If oFSO.FileExists(reg) Then
                        'Se tratan los registros de OST
                        OSTReg reg
                        'Importar SessionData2.reg
                        regCmd = "cmd.exe /c reg import " + reg
                        objWShell.run regCmd, Overwrite
                    End If
                End If
            End If
        End If
    End If
End If

```

```

        End If
    Else
        'Cambiar en el registro 15 por 16
        NumVersion = NumVersion + ".0"
        OldNumVersion = OldNumVersion + ".0"
        If oFSO.FileExists(reg) Then
            sNewFileName = CarpetaImportacion + "\NewSessionData2.reg"
            ReplaceReg reg, sNewFileName, OldNumVersion, NumVersion
            'Se tratan los registros de OST
            OSTReg sNewFileName
            'Importar NewSessionData2.reg
            regCmd = "cmd.exe /c reg import " + sNewFileName
            objWShell.run regCmd, Overwrite
        End If
        WindowsMessaging = ("\"HKEY_CURRENT_USER\Software\Microsoft\Windows
NT\CurrentVersion\Windows Messaging Subsystem\Profiles\"")
        regCmd2 = "cmd.exe /c reg export " + WindowsMessaging + " " + CarpetaExportacion +
"\SessionData2.reg"
        objWShell.run regCmd2, Overwrite
    End If

    Else
        'Importar registro sin tratar ya que las versiones de Office en destino y origen es la
misma.
        If oFSO.FileExists(reg) Then
            'Se tratan los registros de OST
            OSTReg reg
            'Importar SessionData2.reg
            regCmd = "cmd.exe /c reg import " + reg
            objWShell.run regCmd, Overwrite
        End If
    End If
End If

If oFSO.FileExists(CarpetaImportacion + "\SessionData1.reg") Then
    'Se importa arbol de claves común a todas las versiones
    regCmd = "cmd.exe /c reg import " + CarpetaImportacion + "\SessionData1.reg"
    objWShell.run regCmd, Overwrite
End If
End If

```

2.5.3 Módulo Configuraciones Office

En este script se utiliza un procedimiento Subrutina (SUB), que es un método de programación que no retorna un valor, a diferencia del procedimiento Función, que si lo retorna. En este caso, el procedimiento se llama ReplaceReg, que recibe 4 parámetros: 2 archivos .reg (antiguo y nuevo) y 2 *strings* o cadenas (antigua y nueva). Es decir, lo que realiza el procedimiento es recibir un .reg y en este archivo debe encontrar la antigua cadena y reemplazarla por la nueva. Por último, se guarda en el nuevo fichero .reg. Básicamente se utiliza para editar las cadenas de registro de Windows, modificando el número de versión del Office.

Por tanto, lo primero que realiza el script es capturar la versión de Office del equipo destino. A continuación, lee el fichero versión.txt, que está en la carpeta de importación para conocer la versión de Office del equipo origen. Una vez se tienen ambas versiones se comparan; en caso de ser diferentes, se llama a ReplaceReg para hacer los cambios de versión en el registro exportado (puede hacerlo 1 o 2 veces, según los archivos .reg que se hayan exportado). Una vez realizado el cambio de las cadenas, se importan estos nuevos registros. Por el contrario, si ambas versiones son iguales, no es necesario llamar a ReplaceReg ya que no se ha de modificar y se importan los registros originales.

En referencia los registros de Windows no hay que realizar más acciones, y se pasaría a tratar los ficheros de configuración, tanto diccionarios como plantillas. Para ello, sólo hay que realizar la copia de los ficheros exportados

previamente en la ruta adecuada del Appdata del Usuario. Como en el caso de la exportación, se tiene en cuenta tanto la versión del sistema operativo como el idioma de éste.

El script generado para este módulo es "Office_import.vbs".

```

Const Overwrite = True
Dim oFSO,objWShell,strKey,NumVersion, fileVersion
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")

'Se le facilita un archivo .reg de inicio y otro final, y una cadena que ha de substituir por otra.
Sub ReplaceReg (OldReg, NewReg, OldKey, NewKey)
Set InPutfile = oFSO.OpenTextFile(OldReg, 1, False, -1)
ContentFile = InPutfile.ReadAll
InPutfile.Close

oFSO.CreateTextFile NewReg
NewContent = Replace(ContentFile, OldKey, NewKey)

set OutPutfile = oFSO.OpenTextFile (NewReg,2,False, -1)
OutPutfile.write NewContent
OutPutfile.close
End Sub

CarpetaImportacion = WScript.Arguments.Item(0)

'Capturar Versión de Office
strKey = "HKCR\Outlook.Application\CurVer\"
NumVersion = objWShell.RegRead(strKey)
NumVersion = right (NumVersion, 2)

'Se comprueba si existe la carpeta para la importacion, si existe comienza el proceso.
If oFSO.FolderExists(CarpetaImportacion) Then
'Si existe el archivo .txt lo lee para extraer la version de Office del origen
If oFSO.FileExists(CarpetaImportacion + "\version.txt") Then
Set OfficeFile = oFSO.OpenTextFile(CarpetaImportacion + "\version.txt", 1, False)
OldNumVersion = Trim(OfficeFile.ReadLine)
OfficeFile.Close
'Si las versiones de Office son diferentes en origen y destino
If NumVersion <> OldNumVersion Then
NumVersion = NumVersion + ".0"
OldNumVersion = OldNumVersion + ".0"
reg = CarpetaImportacion + "\Settings1.reg"
'Se reemplaza en los registros las versiones de Office
If oFSO.FileExists(reg) Then
sNewFileName = CarpetaImportacion + "\NewSettings1.reg"
ReplaceReg reg, sNewFileName, OldNumVersion, NumVersion
End If

reg = CarpetaImportacion + "\Settings2.reg"
If oFSO.FileExists(reg) Then
sNewFileName = CarpetaImportacion + "\NewSettings2.reg"
ReplaceReg reg, sNewFileName, OldNumVersion, NumVersion
End If

'Se importan los nuevos registros tratados
regCmd = "cmd.exe /c reg import " + CarpetaImportacion + "\NewSettings1.reg"
objWShell.run regCmd, Overwrite

regCmd = "cmd.exe /c reg import " + CarpetaImportacion + "\NewSettings2.reg"
objWShell.run regCmd, Overwrite

'En el caso que las versiones de origen y destino sea la misma, se importa el registro sin
editar
Else
regCmd = "cmd.exe /c reg import " + CarpetaImportacion + "\Settings1.reg"
objWShell.run regCmd, Overwrite

regCmd = "cmd.exe /c reg import " + CarpetaImportacion + "\Settings2.reg"
objWShell.run regCmd, Overwrite
End If
End If

'Importar archivos configuración: Diccionarios y plantillas, teniendo en cuenta idioma y
versiones
Archivos = "\Archivos"
UProof = "\UProof"
CarpetaUProof = AppData & "\Microsoft" & UProof
If oFSO.FolderExists(CarpetaImportacion & Archivos & UProof) Then
If oFSO.FolderExists(CarpetaUProof) Then
oFSO.CopyFolder CarpetaImportacion & Archivos & UProof, CarpetaUProof
End If

```

```

        If oFSO.FolderExists(AppData & "\Microsoft" & "\Proof") Then
            oFSO.CopyFolder CarpetaImportacion & Archivos & UProof, AppData & "\Microsoft" &
"\Proof"
        End If
    End If

    Templates = "\Templates"
    CarpetaTemplates = AppData & "\Microsoft" & Templates
    If oFSO.FolderExists(CarpetaImportacion & Archivos & Templates) Then
        If oFSO.FolderExists(CarpetaTemplates) Then
            oFSO.CopyFolder CarpetaImportacion & Archivos & Templates, CarpetaTemplates
        End If
        If oFSO.FolderExists(AppData & "\Microsoft" & "\Plantillas") Then
            oFSO.CopyFolder CarpetaImportacion & Archivos & Templates, AppData & "\Microsoft" &
"\Plantillas"
        End If
    End If

End If

```

2.5.4 Módulo Unidades de Red

Como en todos los scripts de importación, primero se realiza la comprobación de la existencia de la carpeta de exportación del módulo. En este caso, únicamente debe haber un fichero llamado unidades.reg. Realizada la comprobación, se importa el árbol de registro del sistema operativo. Como este árbol no cambia independientemente del sistema, no se requiere ninguna validación previa.

Para hacer efectivo el cambio, es necesario el reinicio del equipo (a contemplar en la Gestión del Cambio).

El script generado para este módulo es "Unidades_import.vbs"

```

Const Overwrite = True
Dim oFSO,objWShell
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")

CarpetaImportacion = WScript.Arguments.Item(0)

'Se comprueba si existe la carpeta para la importacion, si existe comienza el proceso.
If oFSO.FolderExists(CarpetaImportacion) Then

    'Se importa el arbol de registro de Windows de las unidades
    regCmd = "cmd.exe /c reg import " + CarpetaImportacion + "\Unidades.reg"
    objWShell.run regCmd, Overwrite

End If

```

2.5.5 Módulo Certificados

En este script de importación también se comprueba que la carpeta de exportación existe en el directorio indicado. Comentar que todos los pasos realizados en la importación no requieren ningún tipo de edición ya que las rutas tanto de directorios como de carpetas son las misma en todos los sistemas operativos Windows.

En primer lugar, se realiza la importación del árbol de registro que se exportó en el origen. Los siguientes pasos realizan la copia de ficheros contenidos en los directorios indicados, haciendo uso (como en la exportación) de la variable de entorno %AppData%. Las rutas destinos son las siguientes:

- %AppData%\Microsoft\SystemCertificates
- %AppData%\Microsoft\Crypto
- %AppData%\Microsoft\Credentials
- %AppData%\Microsoft\Protect

Cabe indicar que los archivos de estos directorios son considerados como ficheros de sistema y dependiendo de las sentencias o aplicaciones de terceros usadas para tratarlos, hay que utilizar parámetros especiales.

El script generado para este módulo es "CertificadosUsuario_import.vbs".

```

Const Overwrite = True
Dim oFSO,objWShell,AppData
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")
CarpetaImportacion = WScript.Arguments.Item(0)

'Se comprueba si existe la carpeta para la importacion, si existe comienza el proceso.
If oFSO.FolderExists(CarpetaImportacion) Then

    'Se importa el arbol de registro de Windows relacionado con los certificados
    regCmd = "cmd.exe /c reg import " + CarpetaImportacion + "\Certificados.reg"
    objWShell.run regCmd, Overwrite

    'Se copian las carpetas al AppData del usuario que contiene sus certificados
    AppData = objWShell.expandEnvironmentStrings("%APPDATA%")
    Certificado = "\SystemCertificates"
    CarpetaCertificados = AppData & "\Microsoft" & Certificado
    If oFSO.FolderExists(CarpetaImportacion & Certificado) Then
        oFSO.CopyFolder CarpetaImportacion & Certificado, CarpetaCertificados, Overwrite
    End If

    Certificado = "\Crypto"
    CarpetaCertificados = AppData & "\Microsoft" & Certificado
    If oFSO.FolderExists(CarpetaImportacion & Certificado) Then
        oFSO.CopyFolder CarpetaImportacion & Certificado, CarpetaCertificados, Overwrite
    End If

    Certificado = "\Credentials"
    CarpetaCertificados = AppData & "\Microsoft" & Certificado
    If oFSO.FolderExists(CarpetaImportacion & Certificado) Then
        oFSO.CopyFolder CarpetaImportacion & Certificado, CarpetaCertificados, Overwrite
    End If

    Certificado = "\Protect"
    CarpetaCertificados = AppData & "\Microsoft" & Certificado
    If oFSO.FolderExists(CarpetaImportacion & Certificado) Then
        oFSO.CopyFolder CarpetaImportacion & Certificado, CarpetaCertificados, Overwrite
    End If

End If

```

2.5.6 Módulo SAP GUI

En este script del módulo de importación de SAP se utiliza la misma función que se usa en el de exportación, ExistKey. Como ya se ha indicado anteriormente, se usa para saber si una clave de registro existe, y en caso afirmativo, devuelve el valor de ésta.

En primer término, se declaran dos variables (xml e ini inicializadas en *false*) para saber qué tipo de archivo -donde están configuradas las cadenas de conexión- se ha de importar. Antes de comenzar a tratar la información hay que saber si la ruta destino existe (%AppData\SAP\Common); si no existe la ruta de carpetas donde se encuentran los ficheros de configuración, se generan automáticamente. Esto se puede dar si en el equipo destino no se ha ejecutado el SAP en ningún momento.

En segundo término, se comprueba la versión del SAP GUI instalada en el equipo destino, tanto en el versionado como en el parche. Si en la carpeta de importación existe el fichero "SAPUILandscape.xml" se cambia el valor de xml a *true* (significa que la versión de origen ya utilizaba este tipo de archivo). Por tanto, si la versión de destino es superior a 740.4 y el xml tiene este valor, únicamente se copian los archivos exportados. En el caso que tengan la versión indicada pero el xml este en *false*, se revisa si en la carpeta de importación existe el fichero "saplogon.ini"; en caso afirmativo, se cambia el valor de la variable ini = *true* y se prepara el entorno destino.

Cuando se habla de preparar el entorno destino, significa que la versión del SAP GUI de origen utiliza el fichero ini y en la de destino el xml. En este punto hay dos maneras de actuar sobre el registro de Windows para que sea compatible:

- HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\SAP\SAPLogon\LandscapeFormatEnabled: Si se cambia el valor de la clave a 0, la aplicación apuntaría al fichero ini. Aplicaría a todos los usuarios que utilizarasen el equipo.
- HKEY_CURRENT_USER\Software\SAP\SAPLogon\LandscapeMigrationstate: Si se cambia el valor de la variable a 0, la próxima vez que el usuario inicie SAP realizará la conversión del fichero ini al xml de forma automática. Por tanto, aplica únicamente a ese usuario y convierte el archivo de las cadenas de conexión a la nueva versión utilizada por el fabricante.

Se ha considerado mejor solución la segunda opción, por lo que ha sido la escogida. Para preparar el entorno, aparte del cambio del valor de la clave de registro, también se renombran los ficheros xml en el caso que existan, para que cuando se ejecute el proceso de migración no se den problemas de sobrescritura.

Por último, en el caso de que la versión de la aplicación destino sea inferior a las indicadas anteriormente o la variable ini sea igual a *true* (éste significa que hemos preparado el entorno anteriormente), se realiza la copia de los ficheros adecuados en la ruta destino del equipo.

El script generado para este módulo es "SAPGUI_import.vbs".

```
Const Overwrite = True
Dim oFSO,objWShell,AppData
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")

CarpetaImportacion = WScript.Arguments.Item(0)

'La funcion ExistKey sirve para saber si existe una clave de registro, en el caso que exista
devuelve el valor de la misma.
Function ExistKey(Key)
    on error resume next
    ExistKey = objWShell.regread(Key)
    bFound = (err.number = 0)
    on error goto 0

    If Not bFound Then
        ExistKey = ""
    End if
End Function
```

```

'Se comprueba si existe la carpeta para la importacion, si existe comienza el proceso.
If oFSO.FolderExists(CarpetaImportacion) Then
    xml = false
    ini = false

    'Se consulta el registro para saber la ruta del SAPGui.exe. Se tiene en cuenta que el SO pueda
ser de 32 o 64 bits
    keySAP = "HKEY_LOCAL_MACHINE\SOFTWARE\SAP\SAP Shared\SAPsysdir"
    pathSAP = ExistKey (keySAP)
    If pathSAP = "" Then
        keySAP = "HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\SAP\SAP Shared\SAPsysdir"
        pathSAP = ExistKey (keySAP)
    End If

    'Conocer la versión de SAP GUI
    If oFSO.FileExists (pathSAP & "\SAPGui.exe") Then
        SAPGuiVer = oFSO.GetFileVersion(pathSAP & "\SAPGui.exe")
        Version = split(SAPGuiVer, ".")
        SAPGuiVer = Version(0)
        patch = Version(1)
    End If

    If oFSO.FileExists (CarpetaImportacion & "\SAPUILandscape.xml") Then
        xml = true
    End If

    AppData = objWShell.expandEnvironmentStrings("%APPDATA%")
    pathSAPLogon = AppData & "\SAP\Common"

    SubRutas = Split(pathSAPLogon, "\", -1, 1)
    aDIR = ""
    For Each sRutas in SubRutas
        aDIR = aDIR & sRutas & "\"
        If not oFSO.FolderExists(aDIR) Then
            oFSO.CreateFolder aDIR
        End If
    Next

    'Si la versión es 7.40 con parche 5 o superior se importan los xml de configuracion de la
conexion
    If (SAPGuiVer = 7400 and patch > 4) or (SAPGuiVer >= 7500) Then
        If xml Then
            oFSO.CopyFile CarpetaImportacion & "\SAPUILandscape.xml", pathSAPLogon &
"\SAPUILandscape.xml", Overwrite
            oFSO.CopyFile CarpetaImportacion & "\SAPUILandscapeGlobal.xml", pathSAPLogon &
"\SAPUILandscapeGlobal.xml", Overwrite
        Else
            'Si existe saplogon.ini para importar
            If oFSO.FileExists (CarpetaImportacion & "\saplogon.ini") Then
                ini = true
                'Se renombra los archivos de configuracion actual y se modifica la clave para que
realice la iportación
                If oFSO.FileExists (pathSAPLogon & "\SAPUILandscape.xml") Then
                    oFSO.MoveFile pathSAPLogon & "\SAPUILandscape.xml", pathSAPLogon &
"\SAPUILandscape_old.xml"
                    oFSO.MoveFile pathSAPLogon & "\SAPUILandscapeGlobal.xml", pathSAPLogon &
"\SAPUILandscapeGlobal_old.xml"
                    MigrateKey = "HKEY_CURRENT_USER\Software\SAP\SAPLogon\LandscapeMigrationState"
                    objWShell.RegWrite MigrateKey, "0", "REG_DWORD"
                End If
            End If
        End If
    End If

    'En el caso que la version sea inferior o ini = true se importa el .ini y xml de configuracion
de la conexion
    If ((SAPGuiVer = 7400 and patch <= 4) or (SAPGuiVer < 7500)) or ini Then
        oFSO.CopyFile CarpetaImportacion & "\saplogon.ini", pathSAPLogon & "\saplogon.ini",
Overwrite
        oFSO.CopyFile CarpetaImportacion & "\SapLogonTree.xml", pathSAPLogon & "\SapLogonTree.xml",
Overwrite
    End If
End If

```

2.5.7 Módulo ODBC

Este módulo tiene mucho paralelismo con el módulo de unidades de red; esto es debido a que, lo único que se ha de realizar en el script es la importación de los ficheros .reg que se puedan encontrar (puede haber de 0 a 2). La causa es que el equipo de origen puede tener o no configurados ODBCs, y en el caso que los tenga pueden estar configurados a nivel de máquina y/o de usuario.

Por tanto, se comprueba que existan estos ficheros .reg y se importan. Como en el caso comentado, las rutas de estos árboles de registro son invariables independientemente el sistema operativo, por consiguiente, no se requiere de más comprobaciones.

El script generado para este módulo es "Odbc_import.vbs".

```
Const Overwrite = True
Dim oFSO,objWShell
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")

CarpetaImportacion = WScript.Arguments.Item(0)

'Se comprueba si existe la carpeta para la importacion, si existe comienza el proceso.
If oFSO.FolderExists(CarpetaImportacion) Then

    'Se importa el arbol de registro de Windows que contiene la configuración de los ODBC
    'ODBC de Usuario
    regCmd = "cmd.exe /c reg import " + CarpetaImportacion + "\SessionDataUser.reg"
    objWShell.run regCmd, Overwrite

    'ODBC de maquina
    regCmd = "cmd.exe /c reg import " + CarpetaImportacion + "\SessionDataMachine.reg"
    objWShell.run regCmd, Overwrite

End If
```

2.5.8 Módulo Conexiones SNA

Este módulo también se asemeja al módulo de ODBC dado que, lo único que realiza el script es la importación de los ficheros .reg que puedan encontrarse (puede haber de 0 a 2). El motivo es que el equipo de origen puede tener o no configuradas las conexiones SNA, y en el caso que las tenga pueden estar configurados en máquina y/o en usuario.

Se comprueba que existan los ficheros .reg y se realiza la importación. Como en el caso anterior, las rutas de los árboles de registro son invariables independientemente el sistema operativo, lo que significa que no se requieren más comprobaciones.

El script generado para este módulo es "sna_import.vbs".

```

Const Overwrite = True
Dim oFSO,objWShell
Set oFSO = CreateObject("Scripting.FileSystemObject")
Set objWShell = WScript.CreateObject("WScript.Shell")
CarpetaImportacion = WScript.Arguments.Item(0)
'Se comprueba si existe la carpeta para la importacion, si existe comienza el proceso.
If oFSO.FolderExists(CarpetaImportacion) Then

    'Se importa el arbol de registro de Windows que contiene la configuración de las conexiones SNA
    'SNA de Usuario
    regCmd = "cmd.exe /c reg import " + CarpetaImportacion + "\SessionDataUser.reg"
    objWShell.run regCmd, Overwrite

    'SNA de maquina
    regCmd = "cmd.exe /c reg import " + CarpetaImportacion + "\SessionDataMachine.reg"
    objWShell.run regCmd, Overwrite
End If

```

2.6 Validación de la solución

Una vez finalizado el desarrollo de la aplicación de migración, se ha detectado la necesidad de incorporar *logs* de esta forma se podrá monitorizar si existen errores, y en el caso de que los hubiera, poder analizar donde se han producido para poder buscar una solución. Se crea un *log* por cada una de las exportaciones e importaciones. Para ello, se ha generado la siguiente función que se encuentra en todos los scripts de exportación e importación.

Otra modificación realizada, en este caso en las HTAs, es que al llamar a cada uno de los scripts del módulo, se les pasa un argumento más el cual contiene la información del *path* donde se debe generar y escribir el fichero, que se ha determinado sea el *share*.

Función Log generada:

```

LogFile = WScript.Arguments.Item(1)
'Escritura del Log
Sub Log (Lin, Filelog)
    Dim sFileLog, fsolog
    Set fsolog = CreateObject("Scripting.FileSystemObject")

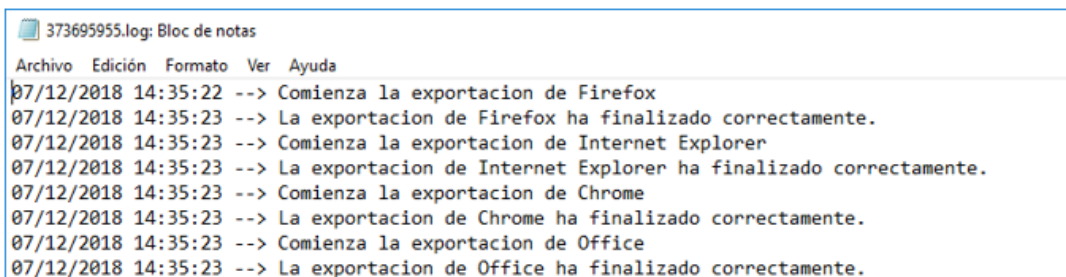
    If not fsolog.FileExists(Filelog) Then
        Set sFileLog = fsolog.CreateTextFile(Filelog, True)
    Else
        Set sFileLog = fsolog.OpenTextFile(Filelog, 8, True)
    End If

    sFileLog.WriteLine (CStr(Now()) & " --> " & lin)
    sFileLog.Close

    Set sFileLog = Nothing
    Set fsolog = Nothing
End Sub

```

Ejemplo Log exportación:



```

373695955.log: Bloc de notas
Archivo Edición Formato Ver Ayuda
07/12/2018 14:35:22 --> Comienza la exportacion de Firefox
07/12/2018 14:35:23 --> La exportacion de Firefox ha finalizado correctamente.
07/12/2018 14:35:23 --> Comienza la exportacion de Internet Explorer
07/12/2018 14:35:23 --> La exportacion de Internet Explorer ha finalizado correctamente.
07/12/2018 14:35:23 --> Comienza la exportacion de Chrome
07/12/2018 14:35:23 --> La exportacion de Chrome ha finalizado correctamente.
07/12/2018 14:35:23 --> Comienza la exportacion de Office
07/12/2018 14:35:23 --> La exportacion de Office ha finalizado correctamente.

```

Figura 35. Ejemplo Log Exportación

Ejemplo Log importación:

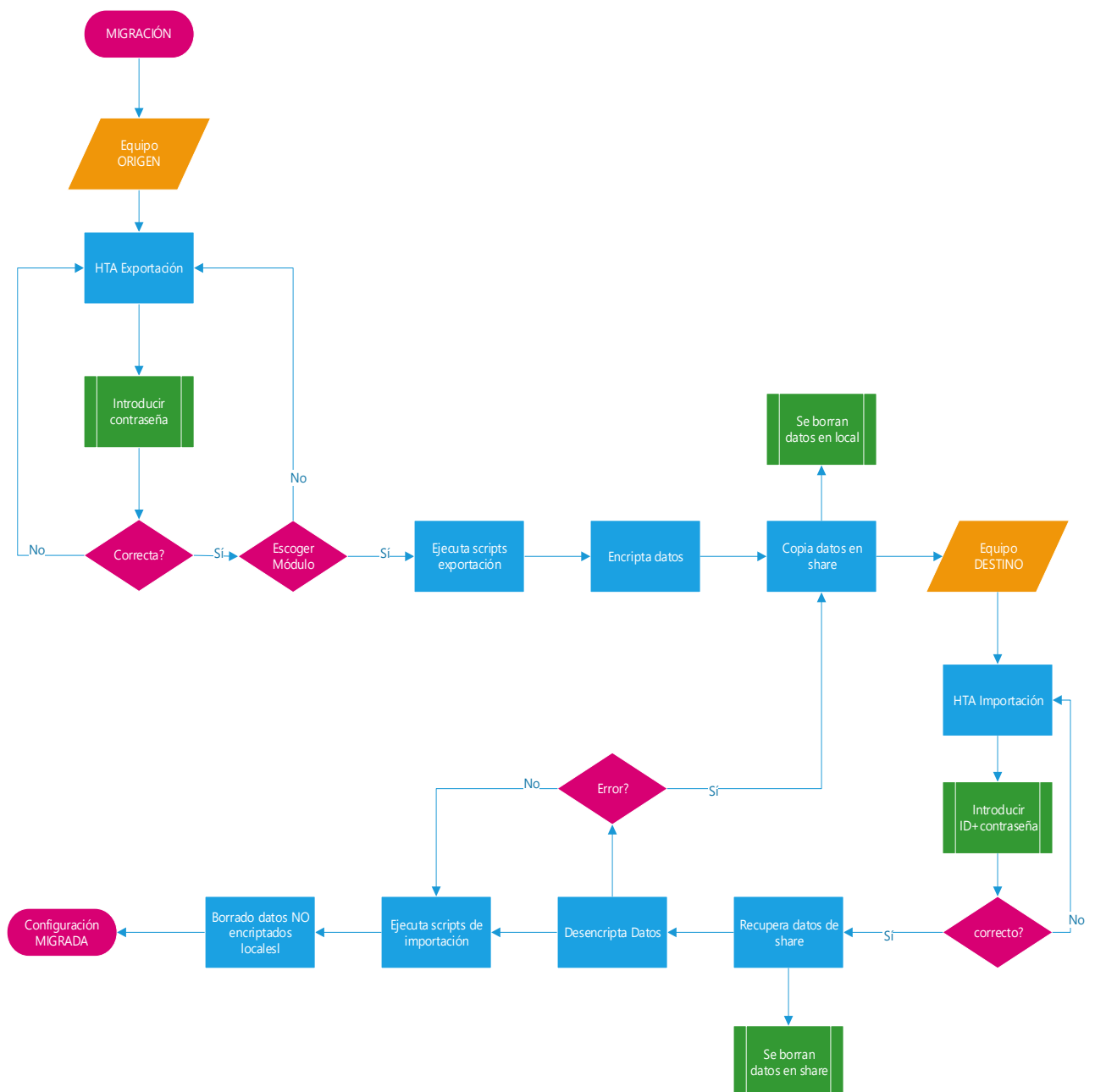
```

384399515_import.log: Bloc de notas
Archivo Edición Formato Ver Ayuda
07/12/2018 14:50:01 --> Comienza la importacion de Firefox
07/12/2018 14:50:01 --> La importacion de Firefox ha finalizado correctamente.
    
```

Figura 36. Ejemplo Log Importación

Para poder ofrecer una visión global del funcionamiento de la solución propuesta, se realiza el siguiente flujo explicativo:

FLUJO PROCESO MIGRACIÓN



3. Conclusiones

En el transcurso de este proyecto se ha podido apreciar una evolución en cuanto a la definición de la solución inicial y la mejora continua que se ha ido aplicando a cada fase de desarrollo del producto. Una buena planificación del proyecto y tener unos conceptos básicos claros, son primordiales para la consecución de un diseño de calidad, y éstas son lecciones que se han de aprender desde el primer momento.

Durante el desarrollo del proyecto se ha profundizado en la versatilidad e innumerables posibilidades que ofrece la programación, en este caso concreto vbScript y HTML, proporcionando sistemas de control de datos o elaboración de aplicaciones. Ha sido todo un reto, dada la naturaleza autodidacta que se ha tenido que aplicar para la realización.

Asimismo, investigar y conocer las interioridades o particularidades de cada producto (sistemas operativos, paquetes ofimáticos, ...) ha sido una lección que aporta un conocimiento a un nivel superior; al igual que aplicar una visión más allá, conectando con el usuario y aplicando usabilidad a la solución (por ejemplo, mensajes de información del proceso al usuario, control de errores de contraseñas incorrectas, ...).

Una vez implementados los diferentes módulos y HTAs y analizado el funcionamiento de toda la solución en su conjunto, se ha constatado que los objetivos planteados en un inicio se han cubierto en su totalidad, todo y encontrar algunos escollos en el camino que finalmente han sido solventados. En algún caso, con una propuesta alternativa de la pensada inicialmente.

Respecto a la planificación del proyecto ha sido la correcta y se ha conseguido cada hito estimado, sin tener que realizar modificaciones durante el desarrollo, así como haciendo uso de la metodología ágil que ha supuesto un avance y agilidad en la realización de la propuesta. El problema que ha sido detectado ha entrado dentro de las previsiones realizadas y no ha habido que introducir cambios de "*timings*" para asumirlo.

La falta de tiempo y recursos no ha posibilitado la excelencia del producto, pudiendo haber incorporado nuevas líneas de trabajo al producto:

- Entornos gráficos más visuales a la hora de presentar los HTA de exportación/importación
- Creación de nuevos módulos
- Depurar más el código, ya que es optimizable
- Ampliar los logs con más detalle para un análisis más eficiente.

El resultado final de la solución transversal ha sido muy satisfactorio, así como todos los conocimientos adquiridos en el transcurso del proyecto. Y la modularidad y transversalidad del producto ofrece una herramienta de apoyo para muchos profesionales del sector, así como un punto de apoyo para las organizaciones en cuanto a coste y retorno de la inversión se refiere. El enfoque que se ha dado a la solución está fijado en una migración de

plataforma, pero una funcionalidad extra que aporta este producto diseñado es la virtualización de las configuraciones de usuario, permitiendo que un mismo usuario pueda tener sus *settings* en cualquier equipo de la organización con "dos *clicks*". Es decir, aporta la funcionalidad de movilidad.

4. Glosario

7zip: programa de código libre utilizado para la compresión de datos.

Autodiscover: es un servicio de Microsoft que permite que los clientes Outlook puedan localizar y configurar de manera automática el acceso al buzón de un usuario.

Batch: archivo de texto con extensión .bat para ejecutar ordenes de DOS de forma secuencial.

Bookmarks: fichero que contiene los favoritos del navegador Google Chrome.

Certificado digital: fichero firmado electrónicamente y emitido por organismos autorizados, que garantiza de manera legal y técnica la identidad de una persona en Internet.

Certmgr: *Certificate Manager tool* se trata de la herramienta de Windows para administrar los certificados de Windows.

Certutil: Se ejecuta el programa en línea de comandos que permite administrar certificados digitales.

Clave registro Windows: Se trata de una base de datos de los sistemas operativos Windows donde se almacena los ajustes de configuración a bajo nivel, tanto de usuario como equipo.

Conexión SNA: *Systems Network Architecture*, se trata de una arquitectura de red usada para conectar con mainframe.

Consola MMC: Consola gráfica de Administración de Windows Server para facilitar las tareas de administración del sistema.

Consulta WMI: *Windows Management Instrumentation* se trata de una base de datos del sistema operativo de Microsoft que contiene información de bajo nivel en un modelo de objetos.

DBMS: Abreviatura de *Data Base Management System*, conjunto de programas que ayudan en la gestión de una base datos de manera sencilla, haciendo de interfaz entre el usuario, la base de datos y las aplicaciones que la hacen servir.

Dominio: red creada por varios ordenadores gestionados por uno o varios servidores dentro de una base de datos central y que autentican los accesos de todos los usuarios.

Encriptación/Desencriptación: proceso de cifrado/descifrado de datos para asegurar la privacidad de éstos. La información es ilegible y solo puede accederse mediante una contraseña.

Exchange: servicio de correo electrónico de Microsoft mediante su programa Microsoft Exchange Server. Normalmente es utilizado por las organizaciones.

Gateway: del inglés, Puerta de enlace es un dispositivo que permite la conexión entre redes con diferentes arquitecturas o protocolos de comunicación.

GPO: Abreviatura de Directivas o Políticas de Grupo en Windows, que establecen una configuración o reglas del objeto afectado (usuarios o equipos).

HTA: Aplicaciones en HTML, son programas que funcionan como aplicaciones mediante un navegador de Internet y que pueden ejecutar diversas tareas preprogramadas.

Log: denominado "registro", se trata de un archivo de texto en el que, de manera cronológica, muestra los sucesos que se han dado en el sistema o aplicación.

Mainframe: Ordenador potente con gran capacidad de procesamiento de datos.

MS: Abreviatura de Microsoft Office.

NK2: archivo de Outlook que almacena las listas de contactos que han sido utilizados para enviar correos, autocompletando envíos posteriores.

ODBC: acrónimo de *Open Data Base Connectivity*, se trata de un estándar de acceso a las bases de datos con el objetivo de poder acceder a éstas desde cualquier aplicación.

OST: (Offline Storage Table) es un archivo de almacenamiento de los mensajes y datos de Outlook en local y fuera de línea.

PowerShell: Lenguaje de programación de scripting basado en instrucciones, ayudando en la automatización de tareas y procesos de los S.O.

Ribbon: Interfaz gráfica de usuario en la parte superior de una ventana compuesta de botones que ejecutan las funciones disponibles de un programa.

Rollback: marcha atrás durante un proceso volviendo al estado inicial.

S.O.: Abreviatura de Sistema Operativo.

SAP: *Systems, Applications, Products in Data Processing* es un software de gestión empresarial que organiza y gestiona los recursos a muy alto nivel.

Script: término utilizado en programación para referirse a un texto con instrucciones escrito en algún tipo de lenguaje interpretado y que ejecuta diversas funciones.

Service Pack: abreviado como SP, engloba un conjunto de actualizaciones de corrección y mejoras de aplicaciones y sistemas operativos, utilizado por Windows para los parches de sus sistemas.

Settings: traducido del inglés “Configuraciones o Ajustes”, en este caso, de usuario.

Share: espacio de uso compartido y colaborativo de información ubicado en la intranet de una organización.

Sniffer: programa informático que captura información de una red, así como la actividad de datos y el uso de los puertos del sistema.

Telnet: protocolo de red que funciona de manera remota accediendo a otras máquinas.

URL: abreviatura de *Uniform Resource Locator*, compuesto por una serie de caracteres siguiendo un estándar y que permite designar o denominar recursos en una red.

Variable de entorno: es un valor dinámico que se añade en los scripts y que permite acceder de una manera simple y sencilla a rutas complejas.

VBScript: *Visual Basic Script Edition*, lenguaje de programación utilizado en scripting e interpretado por Windows Scripting Host de Microsoft.

Workgroup: traducido del inglés “Grupo de Trabajo”. Hace referencia a un equipo dentro de un grupo, pero de manera independiente y sin ser controlado por un servidor, por lo que no hay accesos de administradores.

5. Bibliografía

- [1] <http://web.laplink.com/esp/product/pcmover-express-esp/#tabs2> (30/09/18)
- [2] <https://www.forensit.com/es/index.html> (30/09/18)
- [3] https://es.easeus.com/backup-software/tb-advanced-server-buy.html?linkid=es_product_tbas (30/09/18)
- [4] https://es.wikipedia.org/wiki/Microsoft_Windows (02/10/18)
- [5] <https://docs.microsoft.com/es-es/windows/desktop/SysInfo/operating-system-version> (02/10/18)
- [6] <https://www.codetwo.com/kb/recreate-outlook-profiles-after-migration/> (03/12/18)
- [7] <https://www.quickstart.com/blog/how-to-manually-configure-outlook-on-ms-exchange-server-2016/> (03/12/18)
- [8] <https://support.hostway.com/hc/en-us/articles/115001261650-How-Do-I-Configure-Outlook-2016-To-Connect-To-Exchange> (03/12/18)
- [9] <https://support.microsoft.com/en-us/help/3073002/after-migration-to-office-365-outlook-doesn-t-connect-or-web-services> (03/12/18)
- <https://gallery.technet.microsoft.com/scriptcenter/df14b0d5-d8d3-4502-8250-dfff7e2f66fe> (15/10/18)
- <https://docs.microsoft.com/es-es/windows/desktop/WmiSdk/wmi-tasks--operating-systems> (15/10/18)
- <https://stackoverflow.com/questions/4317794/a-vbscript-to-find-windows-version-name-and-the-service-pack> (16/10/18)
- <https://stackoverflow.com/questions/4542284/how-to-determine-windows-version-from-a-vb-script> (16/10/18)
- <https://support.office.com/en-us/article/create-stationery-for-email-messages-b5552ece-8f09-49ce-81a1-c1b7d347914f> (16/10/18)
- <https://support.office.com/en-us/article/apply-stationery-backgrounds-or-themes-to-email-messages-e24fc237-62e1-4361-82a3-d8a7467d2b7e> (16/10/18)
- <https://norfipc.com/utiles/como-usar-diccionario-word-otro-equipo-instalacion.html> (18/10/18)
- <https://social.technet.microsoft.com/Forums/en-US/7022763a-d39d-4d39-a850-9bc7e7ea0529/registry-setting-for-office-2016-spell-grammar-checker?forum=Office2016setupdeploy> (24/10/18)
- <https://www.mail-signatures.com/articles/outlook-email-signature-location-and-backup/> (25/10/18)
- <https://www.tutorialspoint.com/vbscript/> (26/10/18)
- <https://www.experts-exchange.com/questions/26165311/How-to-make-WMI-filter-to-check-for-installed-software.html> (27/10/18)
- <https://docs.microsoft.com/en-us/sysinternals/downloads/procmon> (29/10/18)
- <https://whatsabyte.com/mac/find-chrome-profile-folders-windows-mac-linux/> (30/10/18)
- [https://support.winshuttle.com/s/article/How-to-find-the-location-of-the-saplogon-ini-file-in-SAP-GUI\(01/11/18\)](https://support.winshuttle.com/s/article/How-to-find-the-location-of-the-saplogon-ini-file-in-SAP-GUI(01/11/18))
- <http://www.mundosap.com/foro/showthread.php?t=73993> (01/11/18)
- <http://kb.mit.edu/confluence/pages/viewpage.action?pageId=3902995> (01/11/18)
- <https://support.microsoft.com/en-sg/help/913111/how-to-use-the-filever-exe-tool-to-obtain-specific-information-about-a> (03/11/18)
- <https://todosapisu.com/sap-variables-del-sistema/>
- <https://social.technet.microsoft.com/Forums/en-US/6e4394b4-16f2-4185-8738-6a7a6774a387/running-32bit-his-2009-client-in-64bit-windows-7?forum=biztalkhis> (03/11/18)
- https://supportline.microfocus.com/documentation/books/reUZE_Server_60/ccms30.htm (03/11/18)
- <https://docs.microsoft.com/en-us/host-integration-server/core/host-integration-server-client-and-sna-communications2> (04/11/18)
- <https://support.microsoft.com/es-pr/help/2449299> (04/11/18)
- https://community.spiceworks.com/how_to/22726-exportar-odbc-de-windows (04/11/18)
- <https://superuser.com/questions/419832/how-can-i-open-the-32-bit-odbc-data-source-administrator-in-windows-7-64-bit> (05/11/18)
- <https://social.technet.microsoft.com/Forums/en-US/821e6848-3b3d-4263-8efa-eea1d660c9e9/vbscript-check-if-registry-key-exists?forum=ITCG> (09/11/18)

<https://docs.microsoft.com/en-us/dotnet/framework/tools/certmgr-exe-certificate-manager-tool> (09/11/18)

<https://social.technet.microsoft.com/Forums/en-US/7022763a-d39d-4d39-a850-9bc7e7ea0529/registry-setting-for-office-2016-spell-grammar-checker?forum=Office2016setupdeploy> (09/11/18)

<https://social.technet.microsoft.com/Forums/en-US/6e4394b4-16f2-4185-8738-6a7a6774a387/running-32bit-his-2009-client-in-64bit-windows-7?forum=biztalkhis> (11/11/18)

[https://docs.microsoft.com/en-us/previous-versions/tn-archive/ee156618\(v=technet.10\)](https://docs.microsoft.com/en-us/previous-versions/tn-archive/ee156618(v=technet.10)) (16/11/18)

<https://www.experts-exchange.com/questions/27690166/VBscript-Call-another-VBscript-with-Arguments.html> (16/11/18)

<http://courses.cs.vt.edu/~cs1204/compression/pkzip.html> (18/11/18)

<https://support.pkware.com/display/PKZIP/Command+Line+Installation+Options> (18/11/18)

<https://social.technet.microsoft.com/Forums/scriptcenter/en-US/bf5ea554-916a-4ab0-a7fd-806fbfa46faf/windowsettimeout-quit-working-in-hta?forum=ITCG> (18/11/18)

<https://www.itninja.com/blog/view/how-to-create-a-customised-popup-notification-window-using-hta> (19/11/18)

<https://www.intowindows.com/how-to-backup-or-export-edge-favorites-in-windows-10/> (19/11/18)

<http://stahlworks.com/dev/index.php?tool=zipunzip#zipexamp> (20/11/18)

https://sevenzip.osdn.jp/chm/cmdline/commands/extract_full.htm (20/11/18)

<https://sourceforge.net/p/sevenzip/discussion/45797/thread/d9c0cef4/> (22/11/18)

<https://github.com/chocolatey/choco/issues/775> (23/11/18)

<https://archive.sap.com/discussions/thread/3810989> (25/11/18)

<https://sevenzip.osdn.jp/chm/cmdline/switches/spf.htm> (25/11/18)

http://content1585.rssing.com/chan-10666611/all_p138.html (26/11/18)

<https://webcache.googleusercontent.com/search?q=cache:QkDuHmNOfuQJ:https://archive.sap.com/discussions/thread/3827734+&cd=1&hl=ca&ct=clnk&gl=es> (26/11/18)

<https://es.scribd.com/document/329478394/SAP-GUI-Administration-Guide-pdf> (27/11/18)

<https://archive.sap.com/discussions/thread/3810989> (28/11/18)

<https://apps.support.sap.com/sap/support/knowledge/mimes/call.htm?number=2638551> (28/11/18)

<https://support.microsoft.com/ca-es/help/2774167/fix-the-sabase-service-may-stop-with-an-access-violation-when-a-tcp-i> (29/11/18)

<https://www.wintips.org/backup-restore-outlook-account-settings/> (30/11/18)

https://answers.microsoft.com/en-us/windows/forum/windows_7-networking/exporting-importing-outlook-account-settings/744f06db-34e6-4a4b-b5fb-26cc09368d4a (30/11/18)

<https://support.office.com/es-es/article/resolver-problemas-de-sincronizaci%C3%B3n-en-las-aplicaciones-correo-y-calendario-en-windows-10-0dd86c69-18f3-4f73-9d3d-375bdc9c3e34> (30/11/18)

<https://docs.microsoft.com/es-es/DeployOffice/oct/oct-2016-help-overview> (30/11/18)

<https://community.spiceworks.com/topic/1993791-configuring-outlook-or-creating-prf-for-office-2016> (01/12/18)

<https://support.pdq.com/knowledge-base/1237> (01/12/18)

<https://support.microsoft.com/en-gb/help/3211279/outlook-2016-implementation-of-autodiscover> (01/12/18)

<https://docs.microsoft.com/en-us/exchange/architecture/client-access/autodiscover?view=exchserver-2019> (01/12/18)

<https://www.jamf.com/jamf-nation/discussions/24417/outlook-2016-autodiscover> (01/12/18)

<https://community.spiceworks.com/topic/2097780-automating-outlook-2016-for-first-time-use-with-oct> (01/12/18)

<http://sbcpro.de/customize-outlook-profiles-by-using-an-outlook-profile-prf-file> (01/12/18)

<https://newsignature.com/articles/bug-alert-exterminate-autodiscover-bug-windows-10-microsoft-office/> (01/12/18)

<https://social.technet.microsoft.com/Forums/en-US/a427d233-d4b9-4d60-96be-ae6c21eff191/how-to-control-placement-of-imappst-files?forum=outlook> (01/12/18)

<https://www.slipstick.com/outlook/command-lines-for-outlook/> (02/12/18)

<https://help.bittitan.com/hc/en-us/articles/115008263508-How-do-I-find-the-main-AutoDiscover-URL-and-Certificate-Principal-Name-for-my-Hosted-Exchange-provider-> (02/12/18)

<https://www.slipstick.com/outlook/command-lines-for-outlook/> (02/12/18)

6. Anexos

6.1 Estructura del share

Junto a la memoria, se adjunta en un fichero llamado "Datos.rar", la estructura existente en el *share* donde se ejecutan los procesos:

- Scripts
- HTAs
- Configuraciones de usuario
- Logs

Nota: Recordar que toda la estructura tiene los archivos con el atributo oculto.

6.2 Script para encriptar .vbs

A continuación se detalla el script (Encoder.vbs) utilizado para encriptar el envíoCorreo.vbs, se ha extraído de:

<https://gallery.technet.microsoft.com/scriptcenter/16439c02-3296-4ec8-9134-6eb6fb599880>

```
Option Explicit

dim oEncoder, oFilesToEncode, file, sDest
dim sFileOut, oFile, oEncFile, oFSO, i
dim oStream, sSourceFile

set oFilesToEncode = WScript.Arguments
set oEncoder = CreateObject("Scripting.Encoder")
For i = 0 to oFilesToEncode.Count - 1
    set oFSO = CreateObject("Scripting.FileSystemObject")
    file = oFilesToEncode(i)
    set oFile = oFSO.GetFile(file)
    Set oStream = oFile.OpenAsTextStream(1)
    sSourceFile=oStream.ReadAll
    oStream.Close
    sDest = oEncoder.EncodeScriptFile(".vbs",sSourceFile,0,"")
    sFileOut = Left(file, Len(file) - 3) & ".vbe"
    Set oEncFile = oFSO.CreateTextFile(sFileOut)
    oEncFile.Write sDest
    oEncFile.Close
Next
```