Validación de Actualizaciones Microsoft Update Catalog con Python, en Equipos con Sistema Operativo Windows (Scripyt-kb)

Cristian Felipe Caro Sierra
Cristian Jacobo Riascos Cortes

Ingeniería de Sistemas - Modalidad Virtual, Fundación universitaria del Área Andina.
Diplomado Programación Python y DevNet
Bogotá, Colombia.

ccaro8@estudiantes.areandina.edu.co
criascos6@estudiantes.areandina.edu.co

Resumen

Este trabajo presenta el resultado de un Script que suple una tarea repetitiva de relación de parches conocidos como actualizaciones (KB) de equipos con Sistema Operativo Windows, almacenados en el catálogo de actualizaciones de Microsoft (Microsoft Update Catalog).

El propósito del Script, es relacionar los KB existentes en el catálogo de Microsoft y valida si hay una porción de código encapsulada en un parche, con mejoras en cuanto a configuraciones del sistema o remediaciones de vulnerabilidades.

Palabras clave

1. ***Vulnerabilidad:*** *Error de código dentro de un sistema, en ocasiones, estos pueden ser intencionados.*
2. ***Actualización:*** *Porción de código, que representa una mejora de un parche ya existente en configuración o remediación de una vulnerabilidad.*
3. ***Remediación:*** *Corregir los errores del sistema con una actualización, la instalación de parches (códigos empaquetados) o configuraciones.*
4. ***URL:*** *Esta sirve para ubicar de manera precisa recursos de un servidor. Esta ruta es la que se coloca en la caja superior de los navegadores.*
5. ***Catalogo:*** *Esta sirve para almacenar recursos con un índice que permite la ubicación rápida del contenido que este provee.*

Abstract

This work presents the result of a Script that supplements a repetitive task of listing patches known as updates (KB) of computers with Windows Operating System, stored in the Microsoft Update Catalog.

The Script itself, relates the existing KBs in the Microsoft catalog and validates if there is a portion of code encapsulated in a patch, with improvements in terms of system configurations or vulnerabilities remediation.

Keywords

1. ***Vulnerability:*** *Code errors within a system, sometimes these can be intentional.*
2. ***Update:*** *Portion of code, which represents an improvement of an existing patch in the configuration or remediation of a vulnerability.*
3. ***Remediation:*** *Fix system errors with an update, the installation of patches (bundled codes) or configurations.*
4. ***URL:*** *This is used to accurately locate server resources. This path is the one that is placed in the top box of the browsers.*
5. ***Catalog:*** *This is used to store resources with an index that allows the quick location of the content it provides.*

I. INTRODUCCIÓN

En la actualidad los sistemas de información han logrado facilitar de una u otra forma las actividades de administradores y usuarios; aun cuando existen plataformas que almacenan y que ante el usuario final presenta un ambiente sencillo de uso, en ocasiones la necesidad de buscar alternativas para acelerar procesos en los que se requiere una tarea repetitiva es considerado indispensable.

Añadido a lo anterior lograr esto requiere un trabajo fortuito, hasta hace poco debido a la evolución de los lenguajes de programación y como estos a lo largo de su historia cada vez tienden a ser más ligeros y sencillos en su sintaxis.

Para muchas áreas laborales, la posibilidad de hacer pequeños segmentos de código que faciliten u omitan tareas repetitivas se ha vuelto una constante, ya que esto es un recurso que permite ahorrar tiempo para distribuirlo en otras actividades que requieran más dedicación.

Este producto tiene como objeto, suplir la tarea repetitiva que valida en el catálogo de Windows las diferentes actualizaciones (KB's) de los equipos que tienen Sistema Operativo Windows.

II. METODOLOGÍA

Creación de ambiente de trabajo en Windows.

En esta sección se hace una breve descripción de las condiciones del ambiente ideal que debe tener el usuario para el desarrollo y ejecución del Script.

Instalación de Python

Se especifica que la instalación que se propone es en ambiente Windows.

Se descarga de la página oficial de Python el ejecutable y se instala en el equipo. Para el caso de este proyecto acepta Python 3 en cualquiera de sus versiones.

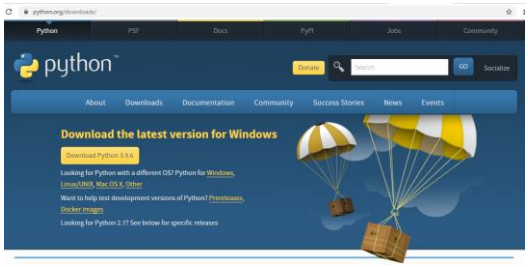

Ilustración 1. Sitio Oficial de Python, Descargas. Recuperada de: https://www.python.org/downloads/

Para esta actividad se utilizó Python 3.8.

Ambiente Virtual

Instalado Python, se crea una carpeta donde va a ir contenido el Script. La idea es generar un ambiente virtual controlado, para hacer la instalación de las librerías sin que estas afecten el funcionamiento del equipo en el resto de sus procesos.

Para hacer el ambiente virtual se ejecuta como administrador el símbolo del sistema y se procede a ir a la carpeta contenedora del Script. Allí se crea el entorno virtual.

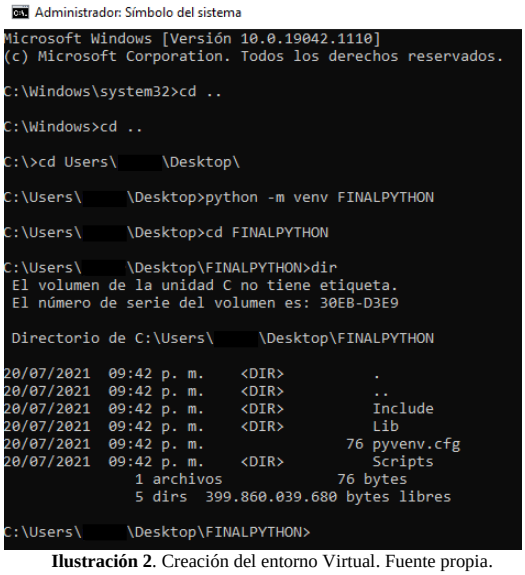

Ilustración 2. Creación del entorno Virtual. Fuente propia.

Una vez creado el ambiente, se ingresa a la carpeta y se activa el entorno con el comando Scripts\activate.

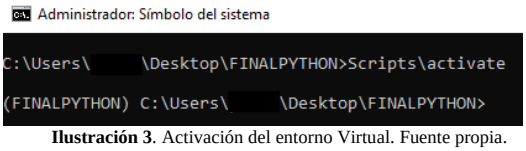

Ilustración 3. Activación del entorno Virtual. Fuente propia.

Como lo representa la imagen anterior, uno se encuentra dentro del entorno virtual cuando el nombre de la carpeta antecede la línea entre paréntesis.

Instalación de Librerías

Ya estando dentro del entorno virtual se deben instalar las librerías necesarias tanto para el desarrollo como para la ejecución del Script.

Pip

Gestor de paquetes, que se usa para la instalación y administración de software en Python. En caso de no tenerlo instalado, se debe descargar e instalar.

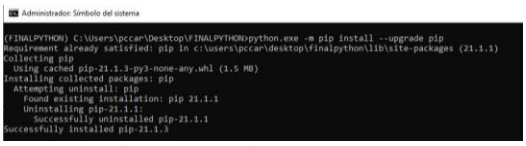

Ilustración 4. Actualización de pip. Fuente propia.

Librería bs4

Esta es una librería de Python para extraer datos html.

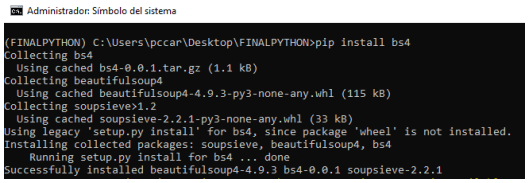

Ilustración 5. Instalación de la librería bs4. Fuente propia.

Librería pandas

Software para gestión de datos y análisis de estos.

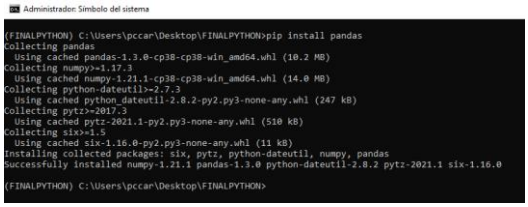

Ilustración 6. Instalación de la librería pandas. Fuente propia.

Librería requests

Librería de Python para hacer peticiones http.

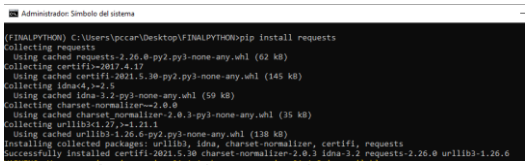

Ilustración 7. Instalación de la librería requests. Fuente propia.

Diagrama de bloques

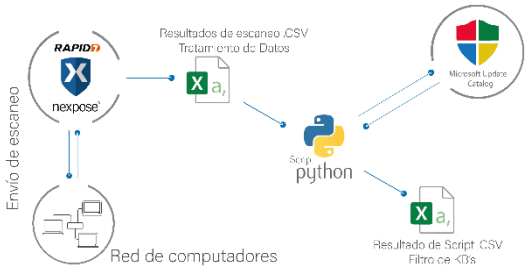

Ilustración 8. Diagrama General de Scripyt-kb. Fuente propia.

Manual de usuario

Scripyt-kb es una herramienta para todo tipo de usuario que requiera validar las actualizaciones de equipos que tengan Sistema Operativo Windows.

A continuación, se presenta el escenario en el cual se consulta el estado de las actualizaciones instaladas en el computador.

Ilustración 9. Historial de actualizaciones. Fuente propia.

Catálogo de actualizaciones de Windows

Este es el sitio oficial donde Microsoft almacena las actualizaciones para su relación, información y descarga.

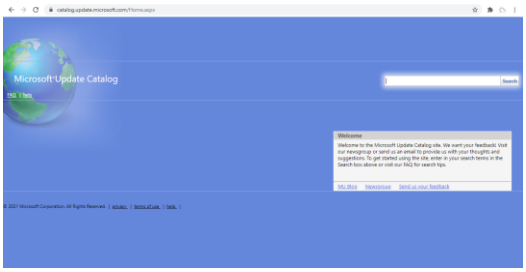

Ilustración 10. Microsoft Catalgo Update. Fuente propia.

Ejecución

Los diferentes pasos que requieren el símbolo del sistema es obligatorio ejecutarlo como administrador.

Para hacer la ejecución del Script el usuario debe tener los siguientes requisitos:

1. Instalar Python.
2. Crear el ambiente virtual.
3. Instalar pip y las librerías:

   - Librería requests
   - Librería bs4
   - Librería Pandas

4. Construir la lista de KB con formato JSON.
5. Instalar un editor de código que permita añadir la lista de KB al Script.
6. Ejecutar el Script.

Ilustración 11. Ejecución de Scripyt-kb. Fuente propia.

Manual programador

Selección del KB de la Lista

El ciclo for, permite la selección del KB siendo este el valor dentro del diccionario, la cual la llama después del ".format" para concatenarla a la URL y

acceder a la información del KB dentro del Catálogo de Microsoft.

```
for actual, kb in ListFind.items():

    url = 'https://www.catalog.update.microsoft.com/Search.aspx?q={}'.format(kb)
    pagina = requests.get(url) # --- Con esta linea se valida si la petición responde 200 ---

    soup = BeautifulSoup(pagina.content, 'html.parser')
```

Ilustración 12. Selección de KB del diccionario. Fuente propia.

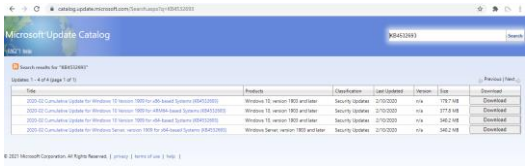

Ilustración 13. Información del KB en el Catálogo. Fuente propia.

Inspección de los recursos que contienen el KB

```
listkb = soup.find_all('td', class_='resultsbottomBorder resultspadding')   # Recorrido de las etiquetas

if listkb:

    # --- Este segmento de Código recorre cada titulo ---

    for i in listkb:
        if 'KB' in (i.text).strip():
            cadena = arreglarid(i['id'])
            diccionario[(i.text).strip()] = cadena # --- Alimenta un nuevo diccionario ---
```

Ilustración 14. Recorrido por los elementos del KB en el Catálogo. Fuente propia.

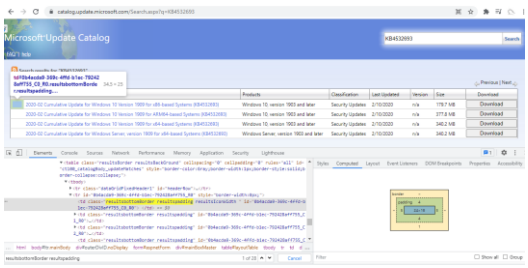

Ilustración 15. Modo gráfico del recorrido por los elementos del KB. Fuente propia.

Esta porción de código una vez hace el recorrido captura los elementos listándolos en un nuevo diccionario.

Recorrido del Id del KB

Una vez se tiene el listado de los elementos, el hace un recorrido por cada uno de los Id para separarlos de forma ordenada y en segundo plano crear un nuevo diccionario.

```
'''
--- Función que recorre los titulos de ListFind vector por vector e identifica los Id---
'''

def arreglarid(idd):
    title = 0
    positionnext = ""
    for i in range(len(idd)):
        for ii in list(idd[i]):
            if title == 0:
                if ii == '_':
                    title = 1
                else:
                    positionnext = positionnext + ii
                    print(positionnext)_# --- Verificación del recorrido del Id --- '''
            else:
                continue
    return positionnext
```

Ilustración 15. Función que recorre el Id del KB en el catálogo. Fuente propia.

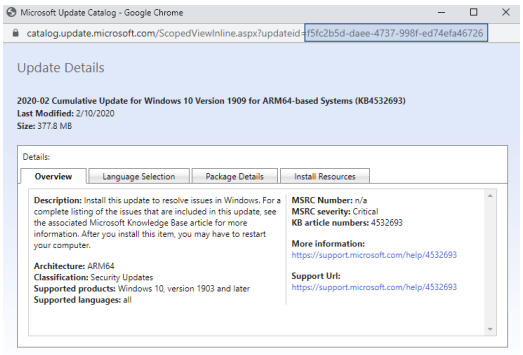

Ilustración 16. Id del KB en el catálogo. Fuente propia.

```
f
f5
f5f
f5fc
f5fc2
f5fc2b
f5fc2b5
f5fc2b5d
f5fc2b5d-
f5fc2b5d-d
f5fc2b5d-da
f5fc2b5d-dae
f5fc2b5d-daee
f5fc2b5d-daee-
f5fc2b5d-daee-4
f5fc2b5d-daee-47
f5fc2b5d-daee-473
f5fc2b5d-daee-4737
f5fc2b5d-daee-4737-
f5fc2b5d-daee-4737-9
f5fc2b5d-daee-4737-99
f5fc2b5d-daee-4737-998
f5fc2b5d-daee-4737-998f
f5fc2b5d-daee-4737-998f-
f5fc2b5d-daee-4737-998f-e
f5fc2b5d-daee-4737-998f-ed
f5fc2b5d-daee-4737-998f-ed7
f5fc2b5d-daee-4737-998f-ed74
f5fc2b5d-daee-4737-998f-ed74e
f5fc2b5d-daee-4737-998f-ed74ef
f5fc2b5d-daee-4737-998f-ed74efa
f5fc2b5d-daee-4737-998f-ed74efa4
f5fc2b5d-daee-4737-998f-ed74efa46
f5fc2b5d-daee-4737-998f-ed74efa467
f5fc2b5d-daee-4737-998f-ed74efa4672
f5fc2b5d-daee-4737-998f-ed74efa46726
```

Ilustración 17. Print del recorrido del Id del KB. Fuente propia.

Una vez terminado el comando y generado el diccionario, se procede a validar la arquitectura que está presentando el valor del ciclo.

```
for titleArq, idArq in diccionario.items():
    for ii, kbArq in ListFind.items():
        if titleArq == ii:
            urlArq = 'https://www.catalog.update.microsoft.com/ScopedViewInline.aspx?updateid={}'.format(idArq)
            arQuitect = titleArq

            #print(titleArq) # --- Mantener el titulo del KB que se esta consultado en la sección Arquitectura ---
            #print(idArq) # --- Mantener el Id del KB que se esta consultado en la sección Arquitectura ---
            #print(kbArq)  # --- Mantener el KB que se esta consultado en la sección Arquitectura ---

        elif kbArq in titleArq:
            urlArq = 'https://www.catalog.update.microsoft.com/ScopedViewInline.aspx?updateid={}'.format(idArq)
            arQuitect = titleArq
            #print(arQuitect)  # --- Mantener el KB que se esta consultado en la sección Arquitectura ---
        else:
            continue
```

Ilustración 18. Ciclo de validación por arquitectura. Fuente propia.

Se vuelve a generar otro diccionario con la información obtenida.

Ilustración 19. Print validación por arquitectura. Fuente propia.

Una vez obtenido el diccionario, el con el siguiente segmento de código, busca en los elementos del catálogo, la actualización más reciente lanzada por Microsoft en su catálogo.

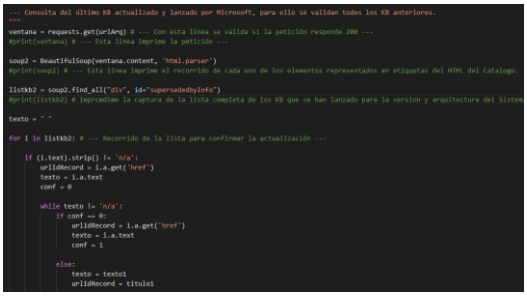

Ilustración 20. Recorrido de la última actualización del KB. Fuente propia.

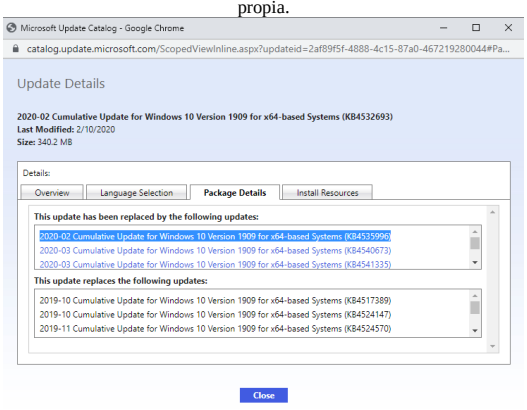

Ilustración 21. Inicio de recorrido de la última actualización del KB. Fuente propia.

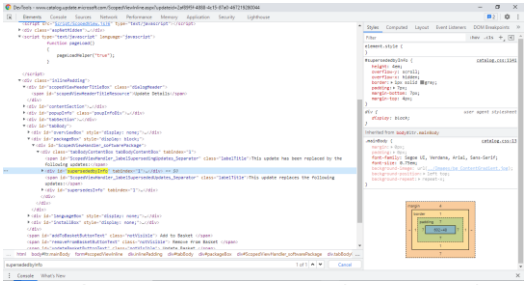

Ilustración 22. Etiqueta del recorrido de la última actualización del KB. Fuente propia.

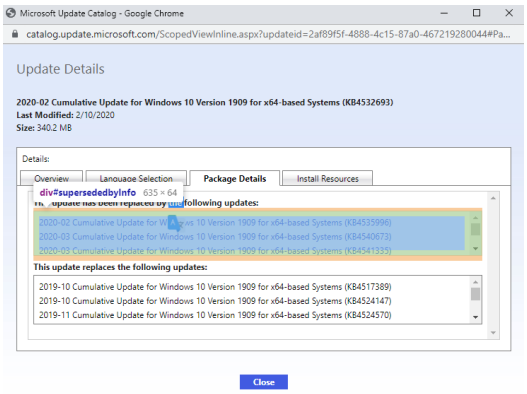

Ilustración 23. Etiqueta del recorrido de la última actualización del KB (Esquema). Fuente propia.

Selección de KB para anexar a .CSV

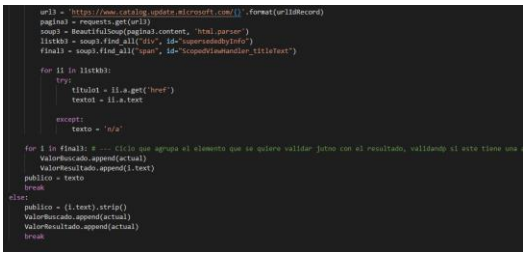

Ilustración 24. Registro de KB en nuevo archivo .CSV. Fuente propia.

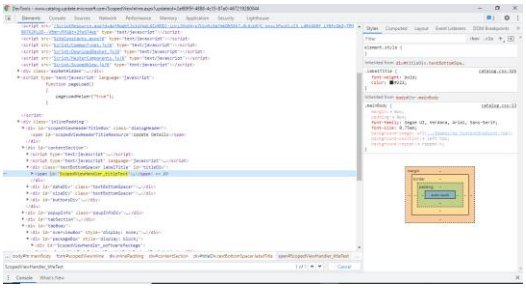

Ilustración 25. Etiqueta de selección del KB para informe .CSV. Fuente propia.

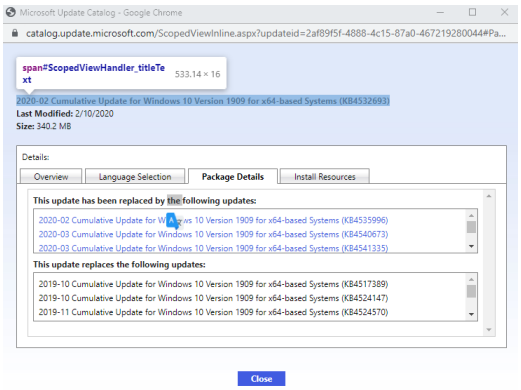

Ilustración 26. Etiqueta de selección del KB para informe .CSV (Esquema). Fuente propia.

Construcción de estructura .CSV

Ilustración 27. Estructura Archivo .CSV y ruta de ubicación para generarlo. Fuente propia.

Recursos

Microsoft Update Catalog

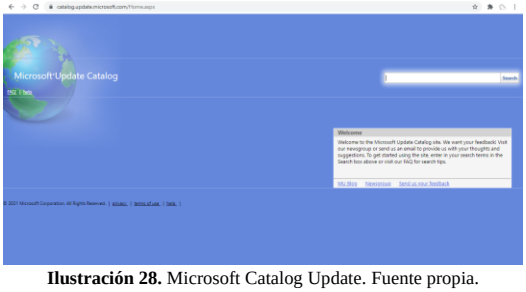

Ilustración 28. Microsoft Catalog Update. Fuente propia.

Símbolo del Sistema.

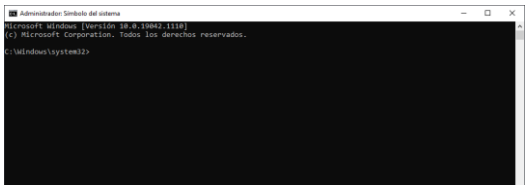

Ilustración 29. Símbolo del Sistema como Administrador. Fuente propia.

Editor de Código.

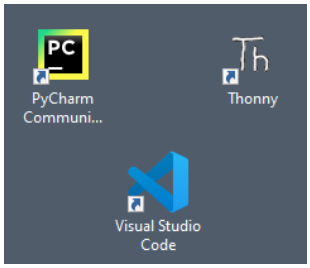

Ilustración 30. Editor de Código compatible con Python. Fuente propia.

Developer Tools

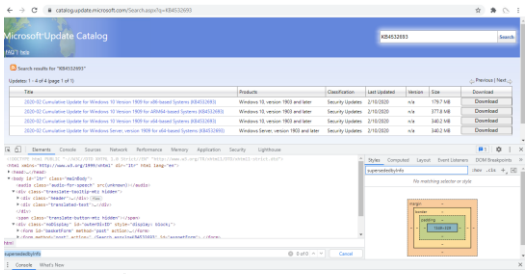

Ilustración 31. Developer Tools Chorme. Fuente propia.

Navegador de preferencia

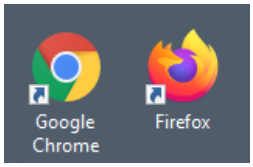

Ilustración 32. Navegador Web. Fuente propia.

III. RESULTADOS

Con el uso del Script se obtiene un informe con formato .CSV de 2 columnas, la primera con el KB que se desea validar y la segunda columna con el KB más actual que compete a la configuración o remediación de la primera columna.

Se obtiene un balance y validación de actualizaciones que necesitan los equipos con Sistema Operativo Windows.

IV. DISCUSIÓN

Haciendo un análisis de funcionalidad de Scripyt-kb se obtiene:

El Script reduce el tiempo de gestión de KB's notablemente al omitir un trabajo manual por cada uno de los elementos que conforman el Catálogo de Microsoft cuando se hace manualmente la actividad.

El nivel de exactitud, respecto a fallas provocadas por errores humanos es alto al ejecutar la tarea de modo manual.

Scripyt-kb hace un recorrido de cada uno de los elementos (Etiquetas) que contiene el catálogo de actualización de Microsoft haciendo un listado con información obtenida a través de scraping o extracción de información de sitios web.

Simultáneamente valida por virtudes de Python cada uno de los elementos extraídos haciendo una inspección de lo que requiere el usuario para la toma de decisiones.

V. CONCLUSIONES

De la actividad y el producto se puede concluir:

El uso de las diferentes alternativas que brinda Python para reducir tareas repetitivas tiene un nivel bastante alto además de su compatibilidad con cualquier Sistema Operativo catalogándolo como uno de los lenguajes de programación mas importantes en la actualidad.

La reducción de tiempos en actividades que requieren un recurso humano es útil para muchos administradores de infraestructuras tecnológicas, como cualquier área que pueda ser intervenida por este lenguaje de programación.

Se puede concluir del proyecto también la variedad de alternativas que ofrece Python con sus librerías y que estas cumplen un papel especifico, reduciendo así en muchas ocasiones la carga de procesos que requiere la ejecución, haciendo de este un lenguaje ligero.

Nombre del producto: **Scripyt-kb**

VI. REFERENCIAS BIBLIOGRÁFICAS

[1] Lozano, J. (2018-2020). Web scraping con Python. Extraer datos de una web. Guía de inicio de Beautiful Soup. [Blog – J2LOGO]. Recuperado de: https://j2logo.com/python/web-scraping-con-python-guia-inicio-beautifulsoup/

[2] Errodringer. (2018, septiembre 25). WEB SCRAPING (MUY FACIL) | PYTHON | 2021. Web scraping con Python en español. [Archivo de video]. Recuperado de: https://www.youtube.com/watch?v=rhnMvvmfBFI

[3] Kuffo, L. (2016, agosto 5). Introducción al Web Scraping | Web Scraping PARTE 1 [Archivo de video]. Recuperado de: https://www.youtube.com/watch?v=j2WF__eM-nE

[4] Watson, J. (2020, enero 4). Python CSV files - with PANDAS. [archivo de video]. Recuperado de: https://www.youtube.com/watch?v=ClNP-lTzKgI