UNIVERSIDAD NACIONAL DE EDUCACIÓN
Enrique Guzmán y Valle
Alma Máter del Magisterio Nacional
FACULTAD DE CIENCIAS
Escuela Profesional de Matemática e Informática

MONOGRAFÍA

SOFTWARE. Conceptos de Software, Importancia del Software, clasificación del Software, el Software Libre y su impacto en el mundo informático actual, producción de Software, aplicaciones.

Examen de Suficiencia Profesional Resolución N° 0756-2019-D-FAC

Presentada por:

Arando Llerena, Yover Max

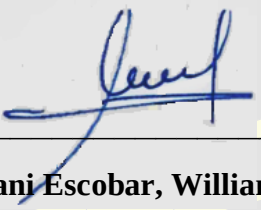
Para optar al Título Profesional de Licenciado en Educación
Especialidad: Informática

Lima, Perú
2019

MONOGRAFÍA

SOFTWARE. Conceptos de Software, Importancia del Software, clasificación del Software, el Software Libre y su impacto en el mundo informático actual, producción de Software, aplicaciones.

Designación de Jurado Resolución N° 0756-2019-D-FAC



Dr. Huamani Escobar, William Alberto

Presidente

Dr. Quispe Andia, Adrián

Secretario

Dr. Morales Romero, Guillermo Pastor

Vocal

Línea de investigación: Tecnología y soportes educativos

Dedicatoria

A todos los que me apoyaron en el logro de este título.

Índice de contenidos

Lista de figuras

Introducción

Un sistema informático consta de hardware y software, el hardware controla los dispositivos electrónicos que son capaces de calcular y manipular información y el software (conjunto de instrucciones) lleva a cabo tareas predefinidas para completar un trabajo determinado. Este sistema debe ser visto como un gran aliado para nuestras tareas del día a día, diseñado para mejorar nuestra calidad de vida. Vivimos en un mundo globalizado, donde la competencia es la única forma de triunfar, y que el éxito es el resultado de una ecuación que consta de las siguientes variables: calidad, rapidez y precio.

Cuando hablamos de Tratamiento de datos estamos hablando de una amplia variedad de actividades que se desarrollan tanto en las organizaciones industriales y comerciales, como en el día a día de cada uno de nosotros. Para intentar definir qué es el procesamiento de datos, tenemos que ver qué tienen en común todas estas actividades. Al analizar, podemos ver que en todos ellos se da cierta información inicial, a la que llamamos datos y que estos fueron sometidos a ciertas transformaciones, con las que se obtuvo la información. El procesamiento de datos siempre involucra tres fases esenciales: ingreso de datos, procesamiento y salida de información. Para transformar los datos utilizamos el software que es parte lógica del sistema de procesamiento de datos, hasta que lo almacenamos en hardware. Por lo que el software es una parte prioritaria para poder transformar nuestras vidas.

Capítulo I

El Software

1.1 Concepto de Software

El software de computadora es cualquier cosa que se pueda almacenar electrónicamente y es lo que ejecuta la computadora: programas. Un procesador de texto es un software, como un juego de computadora. Un programa consta de instrucciones o datos que la computadora debe usar.

Dicho de otra manera, un pianista es el hardware y su partitura musical es el software, si quita una nota y la coloca en otro lugar de la partitura, genera una canción diferente. El Sistema operativo es el software más importante en la computadora, se instala en un área especial dentro del disco duro y se carga (en la RAM) cada vez que se enciende la computadora. Solo después de que se active en el sistema operativo puede llamar a otro software o usar algún hardware. Ejemplos de sistemas operativos: Windows, Linux, Unix, OS, etc.

Por lo general, el usuario no puede acceder a él y generalmente contiene la programación de software básica del dispositivo. El firmware como el BIOS (sistema básico de entrada / salida) de una computadora personal generalmente contiene solo

funciones básicas elementales y permite que la computadora ejecute software más complicado.

"El software se refiere a los programas de computadora y los datos almacenados en la computadora" (Torres, 2014, p. 43). En la mayoría de las plataformas informáticas, el software se puede agrupar en dos categorías amplias:

El software del sistema es el software básico necesario para que funcione una computadora (sobre todo el sistema operativo), que incluye: DOS, Windows, UNIX, Linux, Mac OS, etc.

El software de aplicación es todo el software que utiliza el sistema informático para realizar un trabajo útil más allá del funcionamiento de la propia computadora. Ejemplos de aplicaciones comunes son: procesador de texto, hoja de cálculo, base de datos, navegador de Internet, correo electrónico, juegos, etc. Los datos son todos los documentos y archivos creados o manipulados por el software de la aplicación, incluidos documentos, hojas de cálculo, imágenes, películas, etc.

1.2 Tipos de Software

El software se puede dividir en dos categorías: sistemas operativos y software de aplicación. Los sistemas operativos administran el hardware y crean la interfaz con el hardware y el de aplicación permiten realizar algo útil para el usuario (Torres, 2014).

1.2.1 Sistemas operativos.

Sistema operativo (SO): un conjunto de rutinas que administran los recursos de hardware y brindan facilidades al usuario.

Funciones del sistema operativo:

Como una máquina virtual: presenta facilidad de operación, proporcionando una interfaz entre el usuario y el hardware, por ejemplo, escribiendo en un disquete y también presenta una extensión de las capacidades de la máquina, por ejemplo, múltiples usuarios y protección de acceso.

Como Administrador de Recursos: controla el uso de los recursos proporcionados por el hardware y su distribución entre programas con el fin de asegurar la correcta ejecución de los programas y una alta eficiencia en el uso de los recursos.

Multiprogramación: corresponde a varios programas diferentes que se ejecutan en el mismo procesador. Factores determinantes para la aparición de la multiprogramación: mientras hacía E / S, el procesador estaba inactivo, se vio la necesidad de la multiprogramación y la aparición de circuitos integrados hizo posible la multiprogramación.

Multiprocesamiento: corresponde a varios procesadores, dentro de un mismo sistema informático, ejecutando diferentes programas o cooperando en la ejecución del mismo programa.

Con la aparición de las microcomputadoras, aparece el denominado “user-friendly” para SO, que corresponde al desarrollo de sistemas operativos para ser utilizados por personas sin conocimientos de informática. Y otro desarrollo fue el sistema operativo para redes informáticas, dividido en:

a. Sistemas operativos de red: cada usuario conoce su propia computadora y puede acceder a los datos de otras computadoras.
b. Sistemas operativos distribuidos: el SO hace que todas las computadoras de la red formen una unidad, de modo que ningún usuario sepa cuántas computadoras hay en la red o en qué computadora se está ejecutando su programa.

Evolución de sistemas operativos: Los sistemas operativos han evolucionado a lo largo de los años.

La primera generación (1945-1955): válvulas y tapones: Después de muchos esfuerzos infructuosos para construir computadoras digitales antes de la Segunda Guerra Mundial, a mediados de la década de 1940 se lograron algunos éxitos en la construcción de máquinas calculadoras que utilizan válvulas y relés. Estas máquinas eran enormes y ocupaban habitaciones con bastidores que albergaban decenas de miles de válvulas (y consumían inmensas cantidades de energía). En aquel entonces, un pequeño grupo de personas diseñó, construyó, programó, operó y mantuvo cada máquina. Toda la programación se realizó absolutamente en lenguaje de máquina, a menudo interconectando enchufes para controlar las funciones básicas de la máquina. Se desconocían los lenguajes de programación; los sistemas operativos ídem. Alrededor de 1950 se introdujeron las tarjetas perforadas aumentando la facilidad de programación y la seguridad de los sistemas de información de una organización.

La segunda generación (1955-1965): transistores y procesamiento por lotes: La introducción del transistor cambió radicalmente el panorama. Las computadoras se volvieron confiables y generalizadas (con la fabricación en serie), y se utilizaron en múltiples actividades. Por primera vez, hubo una clara separación entre diseñadores, constructores, operadores, programadores y personal de mantenimiento. Sin embargo, dado su aún alto costo, solo las grandes corporaciones y universidades tenían los recursos y la infraestructura para usar las computadoras de esta generación. Estas máquinas fueron almacenadas en salas especiales con personal especializado para su funcionamiento. Para ejecutar un trabajo (programa), el programador produjo un conjunto de tarjetas perforadas (una tarjeta por comando de programa) y se las entregó al operador que ingresó el programa en la computadora. Cuando la computadora completó el trabajo, el operador

devolvió las tarjetas con la impresión de los resultados al programador. La mayoría de las computadoras de segunda generación se utilizaron para cálculos científicos y de ingeniería. Estos sistemas fueron programados en gran parte en FORTRAN y ASSEMBLY. Los sistemas operativos típicos eran FMS (Fortran Monitor Systems) e IBSYS (sistemas operativos de IBM).

La tercera generación (1965-1980): circuitos integrados y multiprogramación: A principios de la década de 1960, la mayoría de los fabricantes de computadoras mantenían dos líneas de productos distintas e incompatibles. Por un lado, estaban las computadoras científicas que se usaban para cálculos numéricos en ciencia e ingeniería. Por otro lado, estaban las computadoras comerciales que realizaban tareas como clasificar datos e imprimir informes, siendo utilizadas principalmente por instituciones financieras. IBM intentó resolver este problema introduciendo la serie System / 360. Esta serie constaba de máquinas con la misma arquitectura y conjunto de instrucciones. De esta forma, los programas escrito para una máquina de la serie realizada en todas las demás. La serie 360 está diseñada para cumplir con aplicaciones científicas y comerciales. IBM no pudo escribir un sistema operativo que cumpliera con todos los requisitos conflictivos de los usuarios. El resultado fue un sistema operativo enorme y complejo (OS / 360) en comparación con el FMS. A pesar de su tamaño y problemas, OS / 360 satisfizo relativamente bien las necesidades de los usuarios. También popularizó muchas técnicas que faltan en los sistemas operativos de segunda generación, como la multiprogramación. Otra característica introducida fue la capacidad de leer trabajos desde tarjetas perforadas a discos tan pronto como el programador los entregó. De esta forma, en cuanto finalizaba un trabajo, el ordenador iniciaba la ejecución del siguiente, que ya había sido leído y almacenado en disco. Esta técnica se denominó spool (operación periférica simultánea en línea) y también se utiliza para la salida de datos. El tiempo de espera para los resultados

del programa se ha reducido drásticamente con la tercera generación de sistemas. El deseo de respuestas rápidas allanó el camino para el tiempo compartido, una variación de la multiprogramación en la que cada usuario tiene una terminal en línea y todos comparten una sola CPU.

La cuarta generación (1980-actualidad): computadoras personales y estaciones de trabajo: Con el desarrollo de circuitos integrados a gran escala (LSI), chips que contienen miles de transistores en un centímetro cuadrado de silicio, surgió la era de las computadoras personales y las estaciones de trabajo. En términos de arquitectura, estos no difieren de los miniordenadores de la clase PDP-11, excepto por el elemento más importante: el precio. Mientras que las minicomputadoras servían a empresas y universidades, las computadoras personales y las estaciones de trabajo empezaron a prestar servicios a usuarios individuales. El aumento del potencial de estas máquinas ha creado un enorme mercado de software dirigido a ellas. Como requisito básico, estos productos (tanto las aplicaciones como el propio sistema operativo) debían ser fáciles de usar, dirigidos a usuarios sin un conocimiento profundo de los ordenadores y sin la intención de estudiar mucho para utilizarlos. Este fue sin duda el mayor cambio con respecto a OS / 360, que era tan oscuro que se han escrito varios libros al respecto. Dos sistemas operativos han dominado el mercado: MS-DOS (seguido de MS-Windows) para computadoras personales y UNIX (con sus diversas líneas) para estaciones de trabajo. El siguiente desarrollo en el campo de los sistemas operativos llegó con la tecnología de redes de computadoras: redes y sistemas operativos distribuidos. Los sistemas operativos de red se diferencian de los sistemas operativos por un solo procesador en su capacidad para manejar recursos distribuidos por procesadores de red. Por ejemplo, un usuario puede acceder a un archivo en un procesador, incluso si básicamente el archivo está en otro procesador. Los sistemas operativos de red proporcionan al usuario una interfaz

transparente para acceder a los recursos compartidos (aplicaciones, archivos, impresoras, etc.), ya sean estos recursos locales o remotos. Los sistemas operativos distribuidos son mucho más complejos. Estos sistemas permiten a los procesadores cooperar en servicios intrínsecos del sistema operativo, como la programación de tareas y la paginación. Por ejemplo, en un sistema operativo distribuido, una tarea puede migrar durante su ejecución de una computadora sobrecargada a otra que tiene una carga más liviana. A diferencia de los sistemas operativos de red que están ampliamente disponibles comercialmente, los sistemas operativos distribuidos todavía tienen un uso restringido.

1.2.2 Software de aplicación.

Este tipo de software son, básicamente, los programas que se utilizan para aplicaciones dentro del sistema operativo, que no están conectadas con el funcionamiento del mismo. Ejemplos: Word, Excel, Paint, Bloc de notas, Calculadora.

Figura 1. Logos de las principales aplicaciones. Fuente. Autoría propia

Pero también tenemos aplicaciones corporativas diseñado para clientes específicos es muy diferente al que se practica para la creación de software para el consumidor final.

Una aplicación corporativa es aquella que utiliza una empresa para satisfacer las demandas internas de gestión, producción y administración. Entre los tipos más conocidos de aplicaciones corporativas se encuentran los ERP (Enterprise Resource Planning) y los CRM (Customer Relationship Manager), pero existen varios otros tipos de software en esta área.

Las aplicaciones desarrolladas para empresas son software extremadamente personalizado y específico. Cuando un desarrollador decide crear una aplicación para el comercio minorista, primero debe comprender quién es exactamente su consumidor y, a partir de datos e investigaciones confiables, perfilar esta audiencia para comprender la demanda real del producto. En base a esto, será posible crear algo que sirva a estas personas, de una manera más amplia, como la ropa que se vende en una gran tienda.

Pero si la solicitud se hará para una empresa, el proceso de diseño es un poco diferente: el público y los objetivos ya están muy bien definidos, así como la existencia de una demanda. Aun así, depende de los desarrolladores profundizar en el área de sus clientes para comprender cuáles son los problemas que realmente enfrentan.

Una aplicación corporativa, entonces, es como un traje a medida: es fundamental que las necesidades de la empresa se midan correctamente y que el producto final sea capaz de satisfacerlas. Hoy en día, la mayoría de estas aplicaciones se entregan como un servicio, utilizando el modelo SaaS - Software as a Service. Esto significa que, en lugar de simplemente vender la aplicación, el proveedor de software es responsable de la estructura y las actualizaciones, cobrando una suscripción por ello. Este modelo de distribución y comercialización acaba resultando más ventajoso para el cliente, que necesita soportar menores gastos e inquietudes.

Importancia tiene una aplicación corporativa: La tecnología está transformando la forma de trabajar en todas las áreas y en todos los mercados. Con un buen CRM, una empresa puede conocer mejor a sus clientes y personalizar automáticamente la estrategia para atender a cada uno de ellos, maximizando las conversiones.

Un ERP, por su parte, centraliza los procesos e información de negocios útiles, permitiendo que todo se ejecute de manera organizada y brindando información calificada para el proceso de toma de decisiones.

Hoy en día, ya no tiene sentido que solo las computadoras puedan acceder a una aplicación corporativa. En un entorno cada vez más dinámico y exigente, las empresas necesitan soluciones móviles a las que se pueda acceder desde diferentes dispositivos. Cuando una plataforma está disponible a través del acceso web, como es el caso de muchos ERP y CRM, esto en teoría ya permite su uso en teléfonos inteligentes, pero en muchos casos vale la pena ir más allá y desarrollar una experiencia totalmente optimizada para dispositivos móviles.

En empresas que tienen grandes equipos de ventas, por ejemplo, una aplicación corporativa en los teléfonos celulares de los vendedores les permite organizar, informar y negociar con sus clientes mucho más rápidamente. No es necesario utilizar un portátil para esta tarea: el Smartphone será más práctico, rápido y portátil, evitando que este empleado tenga que salir con una mochila.

Con CRM Mobile, los vendedores pueden acceder a todo su historial de ventas, verificar información específica sobre cada cliente y comprender mejor el potencial de cada operación mientras están en el campo. Incluso es posible completar rápidamente informes verbalmente y luego escuchar la grabación en el momento adecuado.

La movilidad es algo que aporta beneficios a las aplicaciones corporativas, facilitando el trabajo de los empleados y ampliando las posibilidades de uso de este software.

Importancia de monitorear las aplicaciones corporativas: La seguridad es un punto de gran atención en el desarrollo de sistemas y software para dispositivos móviles y web. En el pasado, una gran parte de las aplicaciones corporativas se alojaban en servidores físicos de la empresa y, a menudo, habitaban solo en las redes internas de estas organizaciones, las llamadas intranets.

Hoy en día, con la popularización del software como servicio, las aplicaciones suelen estar alojadas en la nube y se puede acceder a ellas desde cualquier dispositivo que cuente con la autenticación correcta. Pero al contrario de lo que pueda parecer a un lego, la migración de estos servicios a la nube no los ha hecho más vulnerables: por el contrario, la estructura de defensa de las empresas especializadas que alojan los sistemas en la nube es incomparablemente más robusta y sofisticada. .que lo que se hizo localmente (y todavía lo es en algunos casos).

Es muy poco probable que un gran servicio de alojamiento en la nube sufra las consecuencias de ataques de diccionario burdos que explotan criptografía débil o malware masivo. Por otro lado, hay un factor que sigue reclamando atención: el elemento humano.

El problema es que estas credenciales pueden haber sido tomadas del usuario en un entorno fuera de la protección de la nube, como una estafa de phishing, que son esos correos electrónicos y mensajes fraudulentos que utilizan la ingenuidad y el exceso de confianza de las personas para obtener contraseñas y otros datos confidenciales.

Otro peligro para la seguridad digital puede surgir con empleados insatisfechos o incluso personas despedidas que, por alguna razón, aún tienen acceso al sistema.

Para protegerse de estas amenazas, existen dos estrategias que pueden abordarse simultáneamente. El primero y más simple es crear un modo de seguridad adicional para la autenticación, que requiere que se realice en dos pasos: además de la contraseña, los usuarios deben ingresar un código que se genera automáticamente y se envía a sus teléfonos celulares.

Otra forma de mejorar la seguridad digital es monitoreando constantemente la actividad de las aplicaciones. Si un usuario en particular verifica regularmente información sensible o intenta destruir datos, esto puede generar automáticamente una alerta para que la situación sea inspeccionada por el equipo técnico.

1.3 Evolución del Software

En el principio de procesamiento de datos, el hardware, que es toda la parte física que constituye una computadora, el equipo en sí, no contenía programas instalados y necesitaba ser cambiado físicamente según cada proyecto o propósito, a fin de satisfacer dicha demanda.

ENIAC se creó con el plan de almacenamiento de software inicial en su interior. Pero, debido al tiempo limitado para poner en marcha la máquina, esta idea fue abandonada. Por lo tanto, la ENIAC también tuvo que modificarse físicamente cada vez que se realizaba una tarea diferente. El código binario creado por Leibniz es un ingrediente central de todas las computadoras modernas y ha sido fundamental desde su creación. La EDVAC, una CPU creada por John Von Neumann en 1945 y también la Mark I, de la Universidad de Harvard, marcan el inicio de la era de las computadoras modernas capaces de almacenar programas.

Estos programas pasaron a llamarse Software unos años más tarde. El término inglés software se utilizó por primera vez en 1958 en un artículo escrito por el científico

estadounidense John Wilder Tukey. También fue responsable de introducir el término "bit" para designar "dígito binario".

Cronología de la evolución del software:

Década 40: cada programa se ejecutó por sí solo y tenía el control total de la computadora. Todo tenía que ser programado en detalle por el desarrollador, desde cargar el programa en la memoria, escanear los periféricos de entrada para buscar datos, el cálculo real y enviar los resultados a los periféricos de salida.

Aparecen los primeros sistemas operativos

- 50 Década - El concepto de sistema operativo apareció durante la segunda generación de computación moderna (1955-1965), desarrollado por GM Laboratories para la computadora IBM 701 a través de programación por lotes que usaba tarjetas perforadas y luego cintas magnéticas. Así, varios comandos ya podrían ejecutarse en secuencia a través de tarjetas perforadas, eliminando parte del trabajo del operador del terminal. Normalmente, un programa constaba de un conjunto de tarjetas insertadas por el usuario del sistema, en el orden correcto.
- 1961 - El grupo del investigador Fernando Corbató, del MIT, anuncia el desarrollo del CTSS - Compatible Time-Sharing System, el primer sistema operativo que tenía tiempo compartido.
- 1965 - IBM lanza OS / 360, un sistema operativo avanzado de tiempo compartido con excelente soporte de disco.
- 1965 - Un proyecto conjunto entre MIT, GE y Bell Labs define el sistema operativo Multics, cuyas ideas innovadoras influirán en nuevos sistemas durante décadas.
- 1969 - Los investigadores de Bell Labs, Ken Thompson y Dennis Ritchie, crean la primera versión de UNIX.

- 1981 - Microsoft lanza MS-DOS, un sistema operativo comprado a Seattle Computer Products en 1980.
- 1984 - Apple lanza el sistema operativo Macintosh OS 1.0, el primero en tener una interfaz gráfica totalmente incorporada al sistema.
- 1985 - Primer intento de Microsoft en el campo de los sistemas operativos con interfaz gráfica, a través de MS-Windows 1.0.
- 1987 - Andrew Tanenbaum, profesor de informática holandés, desarrolla un sistema operativo didáctico simplificado, pero respetando la API de UNIX, que recibió el nombre de Minix.
- 1987 - IBM y Microsoft presentan la primera versión de OS / 2, un sistema multitarea diseñado para reemplazar a MS-DOS y Windows. Más tarde, las dos empresas rompen la asociación; IBM permanece en OS / 2 y Microsoft invierte en el entorno de Windows.
- 1991 - Linus Torvalds, un estudiante de posgrado finlandés, comienza a desarrollar Linux, lanzando el kernel 0.01 a la red Usenet, pronto adoptado por cientos de programadores en todo el mundo.
- 1993: Microsoft lanza Windows NT, el primer sistema de 32 bits de la empresa.
- 1993 - Lanzamiento de UNIX FreeBSD y NetBSD de código abierto.
- 2001 - Apple lanza MacOS X, un sistema operativo derivado de la familia UNIX BSD.
- 2001: Windows XP.
- 2004 - kernel de Linux 2.6.c
- 2006 - Windows Vista
- 2009 - Windows 7
- 2012 - Windows 8

- 2015 – Windows 10

Sistemas operativos IBM: OS / 360, VM / SP, VM / ESA, VSE / SP, VSE / ESA, MVS, MVS / XA, OS390, zOS, zLINUX.

En la década de los 90 también tuvimos la aparición de los Sistemas Operativos de Red, cuyo mercado estuvo dominado por Novell con su NDS, durante algunos años. Novell llegó a tener una cuota de mercado del 90%, dejando el 1% restante a otros jugadores del mercado de redes, como Lantastic y Windows NT.

Microsoft, que poseía sólo el 5% del mercado, estaba rompiendo gradualmente la hegemonía de Novell. Hoy en día, las redes Microsoft Windows tienen casi el 100% de la cuota de mercado, con solo un pequeño porcentaje de la red Novell distribuida por todo el mundo.

Novell se convirtió en SuSe LINUX y sigue teniendo una pequeña representación en el mundo de los sistemas operativos, en comparación con la época dorada.

1.4 Software propietario y libre

No se puede decir cuál es el mejor modelo de desarrollo de software que puede adoptar una organización: código abierto o propietario.

Aunque muchas de las características de los paquetes de software de código abierto frente a los propietarios los distinguen claramente, también comparten varias características. Los proveedores de software han creado soluciones patentadas que luego lanzaron como código abierto. Del mismo modo, hay distribuidores de código abierto sin licencia.

Si ha obtenido una licencia de software, sin duda se ha encontrado con la licencia de software, lo que significa que probablemente también sepa lo confuso que puede ser.

Las operaciones de TI forman una parte importante de cómo se llevan a cabo los negocios en estos días.

Eso significa que cuando se trata de licencias de software, debe asegurarse de elegir el formato que mejor le permitirá atender a sus clientes. Hagamos un breve resumen de los diferentes tipos de modelos de licencias de software disponibles y cuáles son sus diversas ventajas y desventajas.

1.4.1 Software propietario.

"Es el software que tiene derechos de autor y límites de uso, distribución y modificación impuestos por su editor, proveedor o desarrollador, esta patentado y es propiedad de su propietario y es empleado por diversos usuarios con condiciones predefinidas" (González, 2016, p. 19).

Las licencias, también conocidas como licencias propietarias, llamadas software de código cerrado o software comercial, también son un formato de licencia de forma libre.

Ventajas: A diferencia del código abierto, las licencias propietarias incluyen soporte, corrección de errores y parches, además de otro soporte y soluciones cortesía del desarrollador. Esto puede ayudar a resolver problemas cuando hay interrupciones, ya que los tiempos prolongados de los tickets pueden costar dinero a su organización o, peor aún, a los clientes.

Desventajas: Debido a que este tipo de licencia es tan libre, no ofrece una supervisión real. Eso significa que cuando descargas un título de software con una licencia propietaria, es el desarrollador quien establece las reglas sobre lo que puedes y no puedes hacer con él. Además, debido a que este tipo de licencia no está reconocido por la ley, es posible que deba aceptar un extenso conjunto de Términos y condiciones.

1.4.2 Software libre.

"Es software que se puede usar, modificar y redistribuir libremente con una sola restricción: cualquier versión redistribuida del software debe distribuirse con los términos originales de uso, modificación y distribución gratuitos (conocido como copyleft)" (González, 2016, p. 21).

La licencia de software de código libre es un formato de licencia de derechos de autor aceptada para permitir a los desarrolladores modificar y compartir su código fuente.

Como ventajas tenemos que muchas organizaciones y grupos supervisan la concesión de licencias de código abierto, en parte porque existe desde hace mucho tiempo. Esto lo hace más viable para los desarrolladores de software que buscan proteger su trabajo. Dado que este formato ha existido durante tanto tiempo, más personas lo reconocen, lo que facilita la protección de su código fuente contra alguien que lo use de una manera que usted no autorizó.

Como desventajas, las licencias de código abierto no tienen garantías y no ofrecen soporte para solucionar problemas que puedan surgir. Además, dado que el software de código abierto a menudo es desarrollado y distribuido por varios desarrolladores, los problemas a veces se pasan por alto.

Licencia dual de código abierto es un modelo de negocio de código abierto bajo el cual los proveedores hacen que su software esté disponible a través de licencias de código abierto y bajo un modelo diferente en el que se aplica una tarifa.

Como ventajas, tenemos que este tipo de licencia de software puede ser ventajoso porque le brinda opciones. Puede optar por la versión gratuita, a menudo conocida como *freemium.*

Como desventaja, a menudo, las versiones gratuitas de software disponibles en este formato están limitadas en términos de capacidad. El software de licencia dual de código

abierto con licencia bajo un acuerdo freemium ofrece solo una versión básica de lo que ofrece la versión comercial basada en suscripción.

Cuando se trata de elegir el tipo de licencia adecuado, en última instancia, la pregunta que debe hacerse es: ¿cuál le permitirá atender mejor a sus clientes? En el panorama competitivo actual, atender las necesidades de TI de sus clientes significa no solo cumplir y superar sus expectativas, sino también mejorar todo el tiempo y anticipar sus necesidades de TI antes de que se den cuenta de ellas.

Capítulo II

Software educativo

2.1 Concepto de Software educativo

La Informática como ciencia del tratamiento automatizado de la información se viene introduciendo en las instituciones educativas progresiva e inevitablemente, así como también la computadora en el ámbito escolar tiene diversas posibilidades de aplicación.

En nuestro medio, el uso reciente de la informática dentro de las actividades educativas, está generando grandes cambios en la forma de enseñanza, porque no solo se emplea en el proceso educativo, sino también en la administración de los sistemas de información educativa.

Las computadoras interactúan con los estudiantes y mejoran su proceso de aprendizaje. En la actualidad se ha dado singular importancia a la aplicación de la informática como recurso didáctico en la enseñanza de diversos cursos del currículo secundario. "El Software Educativo llamado también programa educativo, programa didáctico o didácticas se designan genéricamente a programas hechos en la computadora con el objetivo de ser empleado como un medio didáctico y facilitar el proceso educativo" (Favaro, 2006, p. 29).

Cualquier aplicación o programa de computadora que tenga un propósito educativo es conocido como software educativo.

2.2 Características de los Software educativos

"Sus características son los componentes básicos que debe tener en cuenta. Al ajustarlos juntos en una colección o secuencia apropiada, un diseñador de software puede producir un paquete para alcanzar las metas educativas a través de estrategias didácticas establecidas" (Favaro, 2006, p .33). Presentamos los atributos más relevantes de software educativos:

- Organizador avanzado: permite dar preparación al estudiante mediante listas de temas, mapeos de recursos, etc.
- Recurso de aprendizaje: cualquier objeto de aprendizaje que pueda servir para el aprendizaje del estudiante.
- Pregunta de autoevaluación: una pregunta para que el alumno pruebe su comprensión, a menudo con un enlace a una respuesta modelo.
- Tarea marcada por computadora: una prueba o examen en un formulario, como opción múltiple o completar espacios en blanco, que puede marcar el programa informático.
- Simulación: permite representar en forma dinámica alguna acción.
- Modelo: El alumno puede cambiar el sistema subyacente que impulsa la simulación (por ejemplo, alterando la Ley de Ohm ecuación para que el circuito se comporte de forma irreal).
- Mapa conceptual: un mapa visual de ideas o temas y sus relaciones. Puede presentarse como mapa fijo (por ejemplo, para usar como organizador avanzado), o puede ser construido por los estudiantes como un medio de explorar sus conocimientos.
- Tutorial interactivo: puede generar una comunicación del computador con el estudiante. Creando convincentes tutoriales interactivos donde la computadora puede responder a

las consultas y conceptos erróneos de un alumno, es uno de los objetivos de la investigación sobre los sistemas de tutoría inteligentes.

- Tablón de anuncios: (o área de discusión) Un servicio en línea para que los alumnos publiquen y respondan mensajes y agrúpelos en hilos de discusión temática.
- Área de chat: una función en línea para que los alumnos se comuniquen a distancia, escribiendo y respondiendo a los mensajes.

2.3 Ventajas de los Software educativos en la enseñanza aprendizaje

Los Software Educativos están revolucionado todos los aspectos de la educación. La tecnología en la educación hoy en día ya no se limita a pizarras digitales, sistemas de gestión del aprendizaje y tabletas. Estas son las diversas formas en que la tecnología ha beneficiado a la educación: Los 7 principales beneficios en la educación

- Aprendizaje inmersivo: por ejemplo, tutoriales basados en realidad aumentada que pueden transferir estudiantes virtualmente.
- Gamificación: hace que toda la experiencia de aprendizaje sea mucho más agradable.
- Aprendizaje accesible a distancia: posibilidad de organizar su tiempo y aprender en un intervalo de tiempo adecuado de su elección.
- Experiencia educativa personalizada: debido al acceso a materiales digitales, complementa los temas de aprendizaje existentes en el aula.
- Preferido por los estudiantes: los estudiantes prefieren integrar la tecnología en su plan de estudios porque ya usan tabletas y teléfonos inteligentes en casa y quieren las mismas herramientas para la educación.
- Un entorno de aprendizaje mixto: permite a los estudiantes tener acceso ilimitado al material de aprendizaje.

- Mejor participación: mejore el enfoque de los estudiantes y haga que se entusiasmen por aprender más.

2.4 Estructura básica de los Software educativos

En la estructura básica del software educativo sus características pueden ser implementadas por una serie de combinación en los medios, como textos, imágenes, animaciones, sonidos, música y video. Se tienen que elegir los componentes y de los medios que permitan mejorar el aprendizaje de los estudiantes.

> Gardner en sus teorías de inteligencias múltiples manifiesta que cada persona tiene diferenciada su tipo de inteligencia y se puede dar en diversos grados. Propone siete formas primarias: lingüística, musical, lógico-matemática, espacial, corporal-cenestésica, intrapersonal (p. ej., comprensión, metacognición) e interpersonales (p. ej., habilidades sociales). Según Gardner, el aprendizaje y la enseñanza deben basarse en las inteligencias particulares de cada persona (Collazos y Guerrero, 2013, p. 92).

Los estudiantes pueden tener alguna preferencia de como organizan sus materiales didácticos. Algunos prefieren aprender de una manera holista, primero obteniendo una descripción general del tema y luego explorando cómo se compone a partir de conocimientos individuales.

Tamaño de la pantalla: El tamaño es generalmente menos crítico de un diseño comparando con la resolución, aunque una resolución demasiado alta en una pantalla pequeña hace que los pequeños detalles ilegible. Las resoluciones van desde la más pequeña a 800x600 hasta un máximo típico de 1200x1600.Al diseñar software, el estado de la pantalla es un recurso precioso y escaso, y es probable que desee utilizar tanto como sea posible. Pero si no se conoce la plataforma de entrega, es posible que se vea obligado a

diseñar dentro de un área de 800x600, para garantizar que todas las personas puedan ver la información. Existen muchos sitios web que están diseñados con este criterio. En pantallas de mayor resolución, suelen ser rodeado por un borde blanco, centrando la atención en el sistema en el centro. Sin embargo, la mayoría de las computadoras tienen resoluciones de al menos 1024x768 y, si conoce a sus usuarios y sus sistemas, en su lugar, es posible que pueda tomar la decisión de diseñar con este tamaño, intercambiando ocasionalmente usuarios fuera del público objetivo que no pueden utilizar el sistema en contra de la interactividad mejorada para la audiencia clave.

Convenciones del sistema: los menús deben encontrarse generalmente encima de las otras opciones dentro del monitor, representados por iconos. La información de complemento debe ser colocado en algún lugar periférico y la información clave en una parte principal. Los usuarios tienden a escanear pantallas con la misma manera en que leen (generalmente de arriba a la izquierda a abajo a la derecha), por lo que la información clave debe estar cerca de la parte superior.

Pantallas y desplazamiento: los diseñadores a menudo tienen que elegir cómo dividir el sistema en sus pantallas de componentes, en sí misma una decisión difícil, y esto se ve agravado al elegir si hacer que haya más información disponible en una pantalla de la que se puede mostrar a la vez. En general, es aconsejable tomar la ruta de desplazamiento considerando aspectos transcendentales como el nivel educativo a quienes está dirigido este software.

2.5 Clasificación de los Software educativos

Hay muchos software educativos disponibles y es imposible clasificarlos todos, consideramos a Dumas (2008), cómo ve la industria del software educativo:

2.5.1 Material didáctico.

Un acrónimo de cursos y software, el software educativo es, con mucho, el tipo más común de software educativo disponible. “El material didáctico ha existido desde que los laboratorios de computación reservan su uso para estudiantes. Incluyen cualquier software diseñado para instruir al alumno y puede ser en forma de un kit de profesor o tutorial” (Dumas, 2008, p. 37). Por lo tanto, tanto los programas de arte de jardín de infancia como el software complementario vinculado a un curso de laboratorio de física se consideran material de curso, según esta definición.

El material didáctico se ha mejorado y ampliado profundamente en solo un par de décadas, y algunos cursos pueden enseñar materias completas con excelente precisión. La idea de lo que podría ser el software educativo también se ha ampliado, ya que ahora incluye software entregado en línea.

Pero cuando las empresas de desarrollo de software se refieren a cursos, a menudo describen un solo curso (de cualquier tema) incluido con materiales de evaluación y lecciones complementarias.

Sin embargo, no importa la definición, el software educativo es la columna vertebral del paquete de software de cualquier educador.

2.5.2 Software de evaluación.

En un esfuerzo por reducir el desperdicio de papel y las pesadillas logísticas que vienen con la impresión de una gran cantidad de pruebas, el software de evaluación se ha convertido en un foco de atención para los distritos escolares. El software de evaluación es simple, ya que solo está diseñado para entregar pruebas o cuestionarios, registrar respuestas y calificar el resultado. El software de evaluación también es de uso

generalizado entre los maestros, ya que es una forma de bajo esfuerzo para rastrear las calificaciones de los estudiantes durante el transcurso de un año escolar.

Debido a que el software de evaluación es simple en principio, hay muchas opciones de código abierto y gratuitas disponibles para los educadores, por lo que esta no es un área donde generalmente se necesita una gran inversión.

2.5.3 Software de referencia.

El software de referencia se ha vuelto completamente en línea, y hay pocos ejemplos de ámbitos escolares que todavía usan piezas de software patentadas con fines de referencia. Esto marca un cambio extremadamente rápido, ya que las computadoras de las escuelas se cargaron con software de diccionarios y enciclopedias tan recientemente como a principios de la década de 2000. Ese ya no es el caso, ya que sitios como Wikipedia han hecho que dicho software sea obsoleto. Y además de los sitios de investigación general como Wikipedia, Google Scholar y Lexis, la mayoría de las revistas médicas y científicas ahora publican en línea. La web ofrece un compendio de conocimientos mucho mayor que el que podría igualar cualquier software de referencia.

2.5.4 Ayudas para el aula.

El mercado de la tecnología y el software de ayuda para el aula se han disparado en los últimos años y representa uno de los enfoques más prometedores para el aula del futuro.

> Las ayudas para el aula incluyen una impresionante tecnología visual y de audio, que combina lo mejor de A / V y material educativo para crear una suma mucho más atractiva. Las ayudas para el aula comenzaron con tecnología gastada y cada vez más arcaica, como el retroproyector, y pronto se convirtieron en pantallas

> digitales mejoradas y, finalmente, en proyectores y pizarrones interactivos (Alessi, y Trollip, 2001, p. 49).

Las pizarras digitales interactivas son el estándar actual en tecnología de ayuda para el aula y han disfrutado de tasas de adopción enormemente altas en Europa. Sin embargo, algunos países se están poniendo al día rápidamente y la tendencia solo se está acelerando. Existen algunos modelos de pizarra digital interactiva en el mercado, pero solo SMARTBoard y Clevertouch son ideales para fines educativos.

El Clevertouch es un buen ejemplo de lo que pueden hacer las pizarras interactivas en el aula. Cada uno viene con una interfaz intuitiva (llamada LUX) que se basa en la interfaz que se encuentra en los dispositivos Android. Los estudiantes, por lo tanto, son estudios rápidos cuando se trata de Clevertouch. Con Clevertouch, los profesores pueden crear lecciones utilizando recursos gráficos y multimedia, lo que reduce enormemente el tiempo de creación de lecciones. Los profesores también pueden modelar principios matemáticos y físicos utilizando Clevermaths y convertir el Clevertouch en un dispositivo para impartir lecciones en varias zonas con Snowflake. Esto, junto con las amplias capacidades de conectividad de dispositivos de Clevertouch, permite una verdadera colaboración multiusuario entre estudiantes.

Y la tecnología de ayuda para el aula viene con software patentado diseñado para mejorar la función de la tecnología. Una vez más, Clevertouch es un claro ejemplo de cómo esta fusión de hardware y software puede generar enormes dividendos para los educadores. A través de Clevertouch, los educadores pueden acceder a Cleverstore, que no es tanto una tienda como un enorme banco de aplicaciones educativas. Se han eliminado todas las compras en la aplicación y se ha desbloqueado todo el contenido, por lo que Clevertouch se puede aumentar de manera importante con el uso inteligente de Cleverstore.

Claramente, la tecnología juega un papel importante en el aula moderna, y la industria del software educativo se ha abierto tanto que existen opciones de software de bajo costo, incluso gratuitas, que pueden proporcionar una gran cantidad de potencia en las lecciones. Uno de los mejores lanzamientos de software en los últimos años es Duolingo, un sitio de enseñanza de idiomas en línea, con una amplia Gamificación para mantener a los estudiantes interesados y es gratis.

Por supuesto, el software como Duolingo suele ser la excepción, y si un ámbito escolar no tiene el tiempo o la gente interna para encontrar alternativas gratuitas al software pago de calidad, entonces pagar puede ser la única opción. Sin embargo, el uso de ayudas para el aula como Clevertouch puede cubrir muchas necesidades de software a la vez, lo que lo convierte en un término medio feliz y una solución altamente rentable.

Existe una solución tecnológica para cada institución educativa, desde escuelas primarias hasta universidades de primer nivel, cada institución educativa puede encontrar su solución ideal, ya sea de pago o gratuita.

2.6 Formulación de un Software educativo

Los docentes como desarrollo de su calidad profesional, deben de desarrollar software de carácter educativo o cultural, para que este nuevo material sea empleado como material educativo en el proceso de enseñanza aprendizaje.

La Direccion General de Tecnologías Educativas (DIGETE) del Ministerio de Educación en su calidad de responsable de integrar las tecnologías de información y comunicación en nuestro país, promueven el mejoramiento de los materiales educativos. Se da un proceso de acreditación, que evaluara y asegurara la calidad del software educativo o cultural producido por los docentes.

La formulación de un software educativo se da bajo los siguientes parámetros que establece la DIGETE hoy en día CRT (Centro de Recursos Tecnológicos).

Aspectos que se deben considerar: El aspecto pedagógico por el que el software educativo debe:

- Fomentar el aprendizaje significativo: Muestra información relevante a las capacidades que se propone desarrollar, y los contenidos tienen que ir con la realidad del estudiante y docente.
- Fomentar la autonomía del aprendiz: Promueve a que el estudiante elabore su propio producto a través de actividades que el software ofrece y a la vez evalué su meta cognición.
- Fomentar la construcción social de conocimientos: En algunos lugares del Perú no existe una correspondencia de uno a uno entre PC, ante esto el software debe plantear actividades de trabajo en equipo que les permita desarrollar proyectos de aprendizaje colaborativo.
- Fomentar la optimización del aprendizaje: Esta adecuado al nivel, grado y desarrollo psicológico del estudiante, presentando una estructura de actividades que van junto a las actividades que quiere desarrollar.
- El aspecto comunicativo por el que el software educativo se considera:
- Comunicación por medios digitales: Debe mostrar una calidad de contenidos, deberá de ser innovador y motivador.
- El aspecto tecnológico por el que el software educativo se considera:
- Interface usuario: Muestra fácil manejo de medios, presentando una adecuada estructura de contenidos que favorecen el aprendizaje.

2.7 Funciones del Software educativo

El uso del software como recurso didáctico es un camino irreversible, considerando la creciente versatilidad del software educativo, así como la capacidad de modelar y simular sistemas reales. “La eficiencia de estos recursos en la formación profesional depende de los criterios didácticos y cualitativos adoptados por los docentes, como la capacidad de simular y la capacidad de desarrollar la autonomía de los estudiantes” (Geisert y Futrell, 2000, p. 112).

La función principal de estas no es reemplazar la figura del docente, sino ayudarlo a mediar en el proceso de enseñanza-aprendizaje, tanto en materias específicas, como para incentivar a los estudiantes a interactuar con los recursos provenientes del avance de este mundo tecnológico y globalizado.

2.8 Modelo de desarrollo de Software educativos

Partiendo de la problemática del desarrollo de software educativo para la enseñanza, se buscaron investigaciones que abordaran este tema con el fin de observar qué métodos y procedimientos se utilizan para el software que tiene como objetivo ayudar a la enseñanza y el aprendizaje. Además, se consultaron las sugerencias, pautas y lo que aún no se ha investigado sobre el desarrollo de este software.

Costa y Costa (2013), con el objetivo de discutir una propuesta para el desarrollo de SE, presentan una propuesta de ingeniería basada en el diseño centrado en el usuario - DCU, combinado con la práctica de metodologías ágiles. Según los autores, el proceso desarrollado es extremadamente útil para asegurar la calidad de los paquetes de software educativo. Incluso al presentar una propuesta para el desarrollo de software, los autores no trabajaron con conocimientos específicos, la propuesta presentada es integral para los diversos conocimientos. Los autores señalan que los recursos tecnológicos contribuyen al

aprendizaje del conocimiento, pero cuando analizan la inserción de las nuevas tecnologías en los entornos de enseñanza y aprendizaje, cuestionan la calidad del software disponible, y creen que el proceso de ingeniería de SE es de fundamental importancia para la calidad del software educativo, según ellos, "el desarrollo de software educativo de calidad implica una evaluación formativa de los prototipos diseñados por los equipos durante el proceso de desarrollo" (Costa y Costa, 2013, p. 6).

Según los autores, los procedimientos de desarrollo de SE combinados con las teorías de aprendizaje colaboran con la calidad de los productos a desarrollar. Además, indican que los procesos no son capaces de resolver todos los problemas, pero presuponen la calidad de los productos.

En investigaciones anteriores, los autores creían que formar un equipo con profesionales técnicos (diseñadores y programadores) y profesores del área a la que estaba destinado el software sería suficiente para construir productos de calidad. Sin embargo, luego del desarrollo de dos sistemas de software educativo, de acuerdo a Costa y Costa (2013) se dieron cuenta de que existe cierta volatilidad en los requisitos educativos, esta situación se basa en la dificultad de obtener y cuantificar los requisitos, además de la necesidad de una comprensión completa de los conocimientos a ser ayudó con el software.

Para superar el problema de los requisitos educativos, se señala que el uso del diseño centrado en el usuario sirve “para describir los procesos de un proyecto en el que los usuarios finales influyen en la forma en que se lleva a cabo” (Costa y Costa, 2013, p. 15). Para ellos, los aportes del diseño centrado en el usuario están relacionados con la obtención directa de información por parte del usuario final.

Algunos métodos de diseño centrado en el usuario sondean a los usuarios sobre las necesidades que tienen en un área educativa en particular, involucrándolos en partes específicas del proceso de desarrollo. Por otro lado, existen métodos en los

> que los usuarios tienen una mayor presencia, integrando el equipo, es decir, se involucran como elementos en todo el proceso (Costa y Costa, 2013, p. 17).

Conviene realizar algunas observaciones para dirigir estudios sobre desarrollo de software y modelos de desarrollo. En relación al uso del diseño centrado en el usuario, nos dimos cuenta que la principal fuente de información para el análisis de los requerimientos se concentró en los usuarios, lo cual se configura como importante, pero creemos que para el concepto de ES la consulta de investigaciones teóricas sobre la enseñanza y el aprendizaje el conocimiento es indispensable.

Para un entorno de micromundo, los usuarios son investigadores, profesores y estudiantes. Los investigadores, en el sentido de desarrollo, se convierten en usuarios con el propósito de proponer y analizar situaciones de uso del entorno, entre otros aspectos. Los docentes también son usuarios en la medida en que configuran los diferentes entornos y situaciones para el uso del artefacto, insertándolo en alguna situación didáctica con el objetivo de enseñar y aprender algunos conocimientos específicos y los estudiantes, a su vez, utilizan el entorno desarrollado para aprender algunos conocimientos.

Con las discusiones mantenidas hasta el momento, inferimos que la perspectiva de desarrollar entornos micro-mundiales basados en estudiantes, en diferentes niveles, como usuarios, se vería perjudicada si los requerimientos del entorno se adquirieran solo con la opinión de los usuarios. Dada la complejidad de la configuración de los micromundos, no sería factible obtener los requisitos de los usuarios finales, ya que estos entornos necesitan situaciones didácticas para aprender de manera efectiva los conocimientos a desarrollar.

Aún sobre los requisitos, hay una rama específica de la ingeniería de software para obtenerlos, la ingeniería de requisitos, que se configura como un aporte útil para conocer los requisitos de un software e incluso de un sistema.

También desde la perspectiva del desarrollo de SE utilizando metodologías ágiles, presenta una experiencia de ingeniería de objetos de aprendizaje.

> Se considera importante utilizar metodologías de desarrollo con el fin de satisfacer necesidades organizativas y pedagógicas, pero consideran que las metodologías de desarrollo tradicionales no satisfacen las necesidades de las propuestas actuales, según ellos, el enfoque de desarrollo tradicional ya no satisfacer las necesidades de las propuestas educativas actuales, especialmente aquellas con un dominio complejo, ya que no siempre contribuyen al aprendizaje final. Por tanto, es necesario investigar enfoques para el desarrollo de recursos educativos más adecuados a la práctica pedagógica (Lapolli, 2009, p. 250).

Muchos autores consideran las primeras metodologías de desarrollo como tradicionales, pero estas formas de desarrollo de software continúan siendo eficientes para la creación de diversos productos. Lo que podemos ver es que el uso de metodologías más antiguas sirve como base teórica y práctica para los modelos más recientes. Los métodos recientes son desarrollados a partir de los antiguos, lo que caracteriza una evolución y no un reemplazo.

Lapolli (2009) manifiesta que utilizaron la metodología ágil, específicamente el BDD Behaviour Driven Development (desarrollo orientado al comportamiento), un proceso que abarca el análisis de requisitos, hasta el desarrollo del código. Según ellos, BDD es la "combinación de varias prácticas consideradas ágiles y útiles en el desarrollo de software, cuyo énfasis está en características de alto valor y reducción de costos de cambio al identificar lo que realmente se está probando" (Lapolli, 2009, p. 253).

En relación a la obtención de requisitos, la consulta sobre el aprendizaje de conocimientos técnicos se realizó a través de entrevistas. Observamos que casi no se

consultan documentos, teorías de aprendizaje, cognitivas u otras. Los autores definen su búsqueda de la siguiente manera:

> Para identificar los requisitos de funcionalidades que deberían abordarse y desarrollarse, realizamos entrevistas semiestructuradas con instructores, donde se investigan las dificultades para construir cognitivamente un modelo del momento en el que los conceptos abstractos influyen significativamente en los procedimientos operativos. En la elaboración de las actividades propuestas, se debe contar con la colaboración de profesionales de la materia con un abanico de conocimientos, habilidades, convicciones y conceptos adquiridos a lo largo de su actividad profesional (Lapolli, 2009, p. 257).

La identificación de requisitos a través de la consulta a los usuarios es una práctica posible gracias a metodologías ágiles. Sin embargo, para el desarrollo de software educativo, específicamente para entornos de micromundo, es necesario cumplir con requisitos de enseñanza y aprendizaje. Analizando al alumno como uno de los usuarios finales en este tipo de entornos, se puede ver la imposibilidad de obtener requisitos didácticos o cognitivos, por ejemplo, ya que el alumno no tiene acceso a teorías cognitivas o didácticas (considerando alumnos de primaria, por ejemplo).

El usuario identificado en las metodologías ágiles, para el desarrollo de software sin intención didáctica, es el cliente, quien debe estar satisfecho con el producto. La perspectiva del usuario es diferente en el caso de SE, ya que, como ya se señaló, la pluralidad de usuarios de un SE es bastante amplia. En este sentido, al considerar algunos de los principios ágiles para el proceso de desarrollo de SE desarrollados, consideramos la pluralidad de la naturaleza de los usuarios y la importancia de su participación en la construcción de software.

Incluso con la participación de los usuarios en el desarrollo de una SE, es de fundamental importancia asegurar que se identifiquen todos los requisitos necesarios, así, nos daremos cuenta de la necesidad de utilizar la ingeniería de requisitos para obtener características, funcionalidades, especificaciones, etc. de los entornos a desarrollar con el proceso discutido. Con este método, es posible identificar las necesidades de los usuarios y del sistema.

2.9 Tipos de Software educativo

Consideramos cuatro tipos principales de software educativos: tutoriales, simulacros, simulaciones y pruebas.

Tutoriales: Se refieren a los software "diseñados para reemplazar las enseñanzas en el aula, proporcionando una instrucción completa del material de aprendizaje" (Shlechter, 1991, p. 105). Suelen estar estructurados para presentar información en pequeños pasos jerárquicos que incorporan explicaciones teóricas como así como la práctica del nuevo material. "Todos los sistemas de tutoría incorporan en uno u otra forma un modelo de un experto (o ejecutante competente) para las habilidades que se enseñan"(Venezky y Osin, 1991, p. 47). La experiencia se pone a disposición del alumno en una serie de pantallas de ayuda a las que los alumnos pueden acceder a voluntad. Las pantallas de auxilio brindan ayuda para superar dificultades o resolver un problema siguiendo algunos ejemplos de expertos.

Simulacros: Los simulacros, o los programas de simulacros y prácticas, ayudan a los alumnos a refinar o mejorar el rendimiento. Normalmente complementan la instrucción en el aula reforzando las habilidades ya aprendidas.

Geisert y Futrell (2003) manifiestan que los procesos rutinarios son simples:

- Presentar al estudiante preguntas o problemas correspondientes al nivel del objetivo de su desempeño;
- El alumno responde escribiendo la respuesta;
- La computadora revisa la respuesta y si es posible realiza una retroalimentación sobre el valor exacto de la pregunta;
- Si la respuesta es correcta, se le presenta al alumno otra pregunta o problema, si la respuesta no es precisa, el alumno tiene la oportunidad de intentarlo de nuevo.

Los simulacros funcionan bien en un sistema educativo a su propio ritmo, dando la oportunidad para que los aprendices rápidos avancen rápidamente en los niveles jerárquicos de actuación. El criterio de dominio puede ser un número fijo de respuestas correctas sucesivas, digamos tres seguidas, o podría ser un cierto porcentaje correcto en la última serie de diez ejercicios

Simulaciones: Las simulaciones difieren tanto de los tutoriales como de los ejercicios y la práctica de programas en que las interacciones de los alumnos no son respuestas a preguntas, sino decisiones que toman en una situación de juego de roles. Es decir, las respuestas de la computadora dependen de las opciones que los estudiantes hacen, Alessi y Trollip (2005) dividen las simulaciones en cuatro formas: físico, procedimental, situacional y de proceso.

En un físico la simulación modela algún aspecto de la realidad física, como un avión, cabina, con la que el alumno debe interactuar. Las procedimentales presentan acciones que permitirán un procedimiento particular a ser aprendido. Las simulaciones de tipo situacional representan en forma directa las interacciones humanas con el medio ambiente u otras personas. Las simulaciones de procesos genera la experimentación en el alumno de situaciones que podría pasar en un entorno seguro.

Pruebas: Tutoriales, simulacros y simulaciones presentan información y estrategias para ayudar a los alumnos a lograr resultados de aprendizaje específicos. Las pruebas se utilizan para evaluar este logro. Por lo general, las pruebas se generan a partir de un banco de elementos de prueba. "El concepto de banco se refiere a una colección muy rica de preguntas, que proporciona una respuesta al problema de evitar la repetición de la misma pregunta al permitir la construcción de pruebas equivalentes en el mismo sujeto "(Venezky y Osin, 1991, p. 180). La puntuación se logra mediante el hacer coincidir la respuesta de los alumnos con la que se ha registrado para ese elemento en la clave de prueba.

Capítulo III

Principales Software educativo y modelos de aprendizaje

3.1. Software educativo libre para utilizar en el aula

Se ha seleccionado una lista de las mejores herramientas y software de enseñanza en línea gratuito.

Canva: Una brillante herramienta de diseño gráfico gratuita que puede utilizar para crear imágenes e infografías educativas para el aula. No podría ser más intuitivo y es solicitado tanto por principiantes como por expertos.

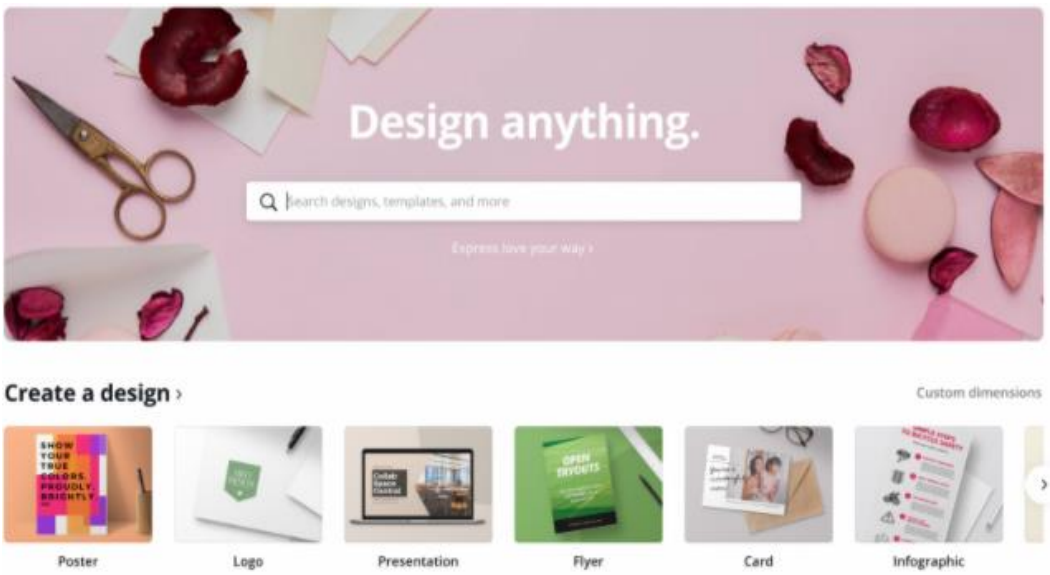
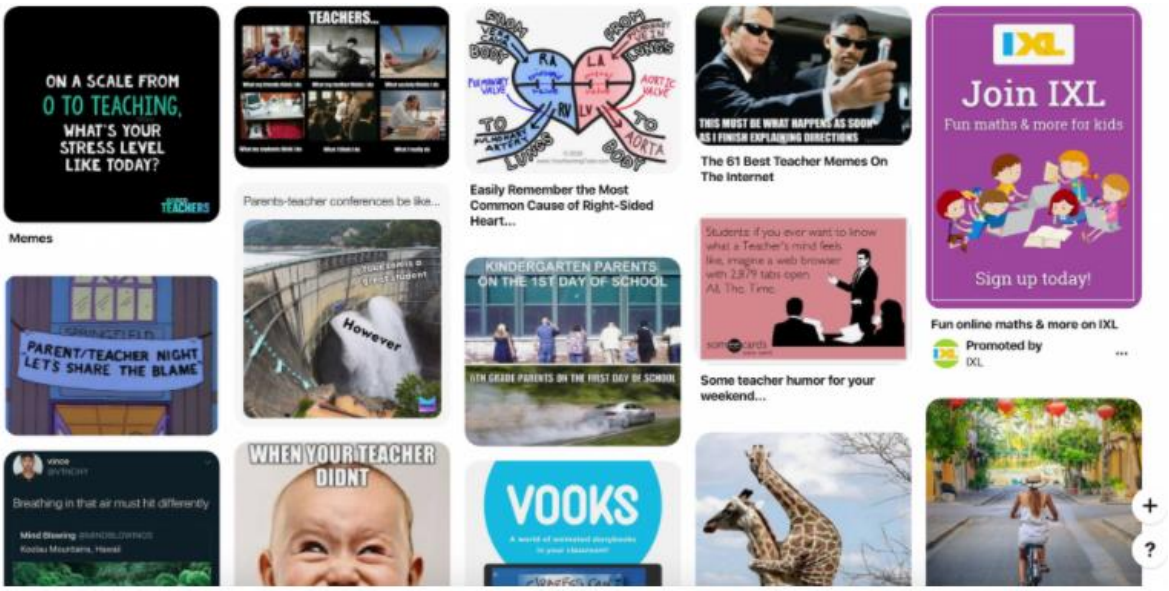


Figura 2. Presentación del Canva. Fuente: Recuperado de https://www.canva.com/es_419

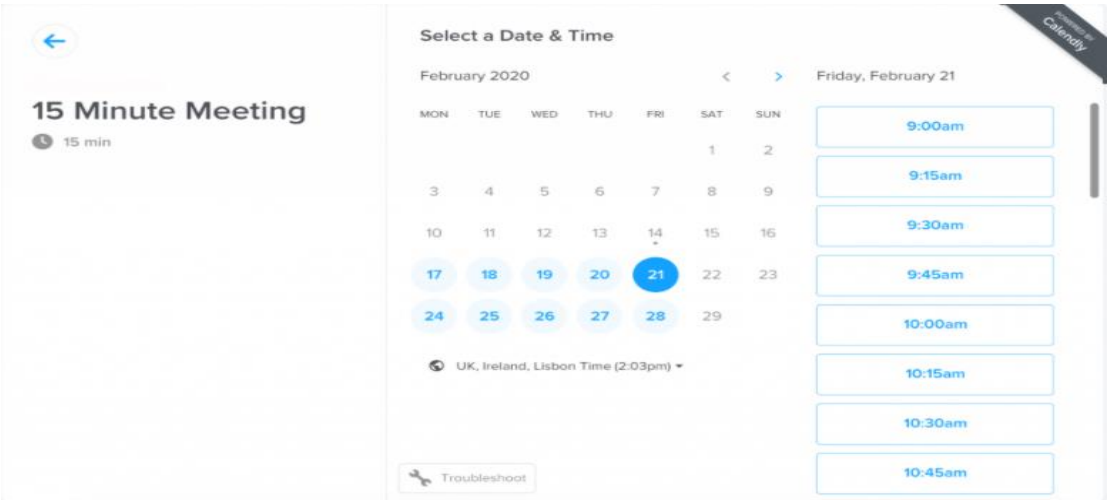
Pinterest: Crea una cuenta u utilizarla convenientemente. Desde planes de lecciones épicos hasta un diseño inteligente del aula, Pinterest es el lugar para estar. Crear sus propios tableros para recopilar todas las ideas en un solo lugar y compartir sus mejores



tableros con los alumnos.

Figura 3. Presentación de Pinterest. Fuente: Recuperado de https:// www.pinterest.com/

Calendly: Mantiene el control de con quién te encuentras y cuándo con Calendly, una aplicación de reserva de calendario gratuita. Calendly se integra con su calendario para que nunca tenga una reserva doble, cuando enseña y le permite ahorrar tiempo al reservar



sus citas de padres y maestros o reuniones con colegas.

Figura 4. Presentación de Calendly. Fuente: Recuperado de https:// calendly.com/

Edmodo: Red de educación global que conecta maestros y estudiantes. Colabora en grupos, administra y proporciona materiales educativos, mide el desempeño de los estudiantes y comunica con los padres para crear una experiencia de aprendizaje más personalizada y enriquecedora.

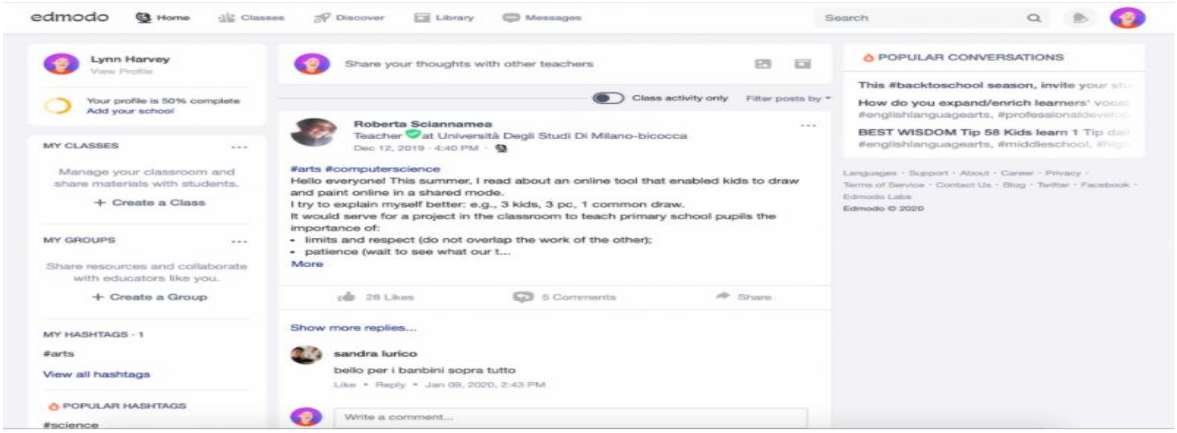


Figura 5. Presentación de Edmodo. Fuente: Recuperado de https://www.calendly.com/

Classdojo: Es una de las favoritas entre los maestros, es una plataforma de comunicación escolar gratuita que los maestros, estudiantes y padres pueden usar todos los días para construir comunidades unidas al compartir lo que se aprende en el aula a través de fotos, videos y mensajes en el hogar. Los padres pueden realizar un seguimiento del progreso de sus hijos y los niños pueden mostrar su aprendizaje. También hay un sistema gratuito de herramientas de gestión del comportamiento de la clase y muchas funciones adicionales para los profesores.

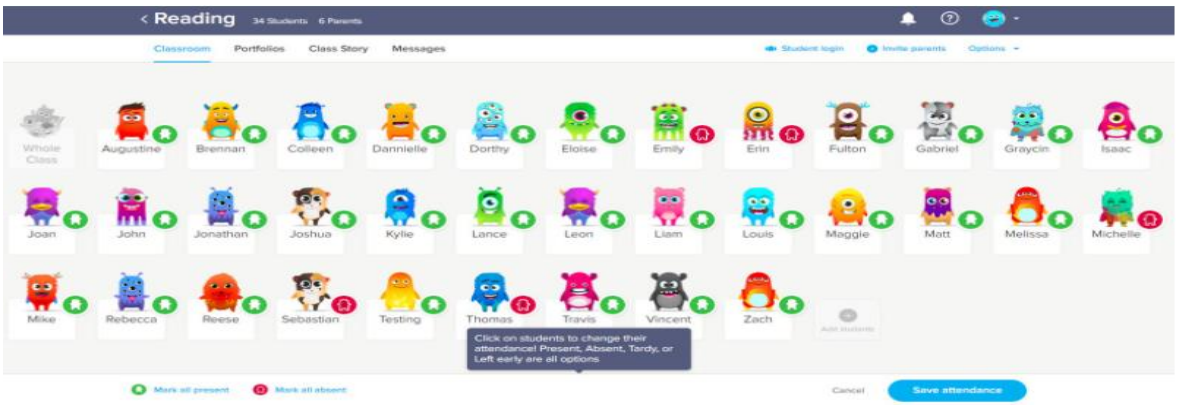


Figura 6. Presentación de Classdojo. Fuente: Recuperado de https://www.calendly.com/

Google Classroom: Ayuda a las clases a comunicarse, ahorrar tiempo y mantenerse organizadas. También hace que la enseñanza sea más productiva al permitirle optimizar las tareas, impulsar la colaboración y fomentar la comunicación. Puede crear clases, distribuir tareas, enviar comentarios y ver todo en un solo lugar. Google Classroom también se integra a la perfección con otras herramientas de Google como Google Docs y Drive, lo que te ayuda a mantenerte organizado y, lo que es más importante, a ahorrar tiempo.

Figura 7. Presentación de Google Classroom. Fuente: Recuperado de https://classroom.google.com/

Kahhot: Permite crear juegos de aprendizaje o un cuestionario de prueba sobre cualquier tema en cualquier idioma en minutos y hace que el aprendizaje sea divertido.

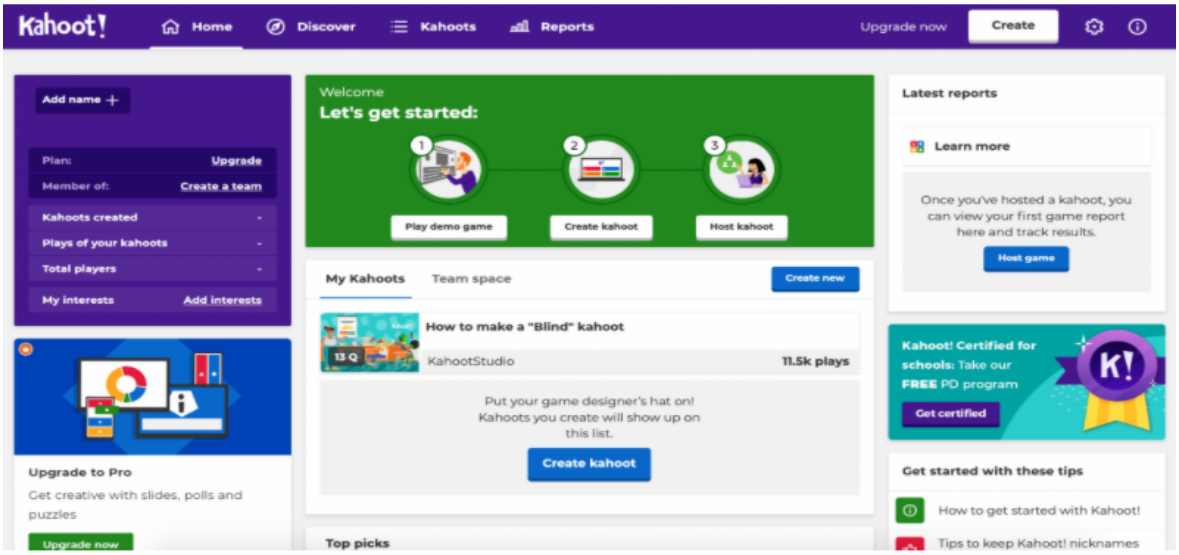
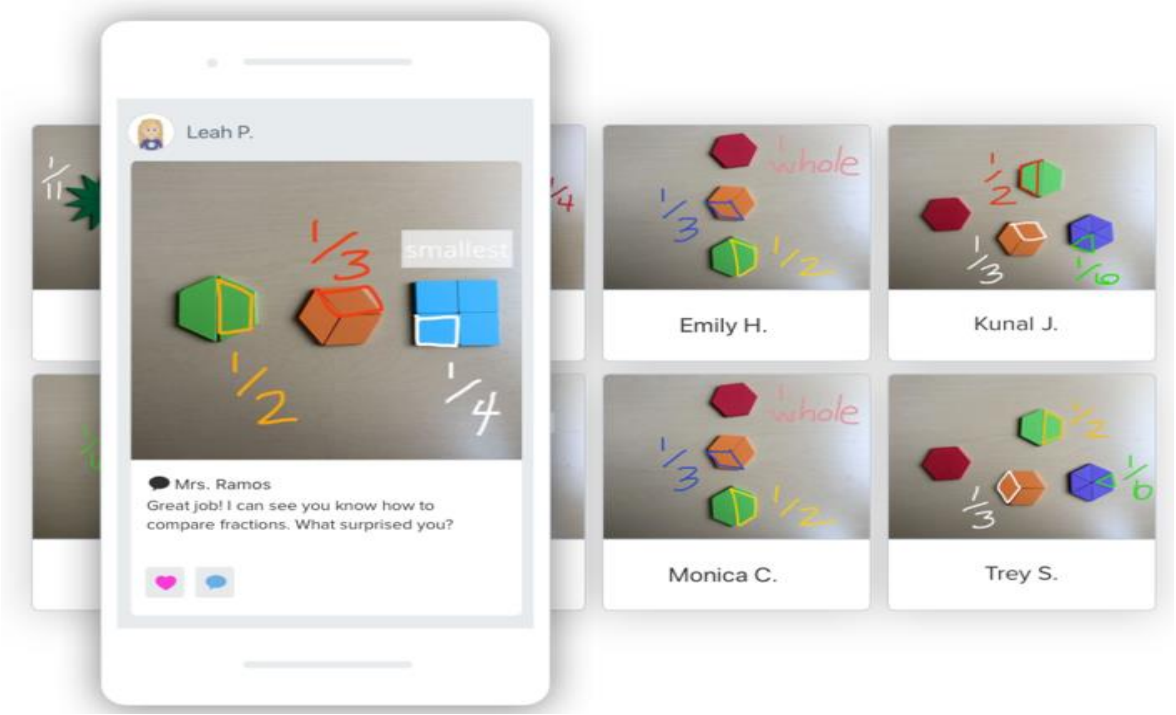


Figura 8. Presentación de Kahhot. Fuente: Recuperado de https://www.kahoot.com/

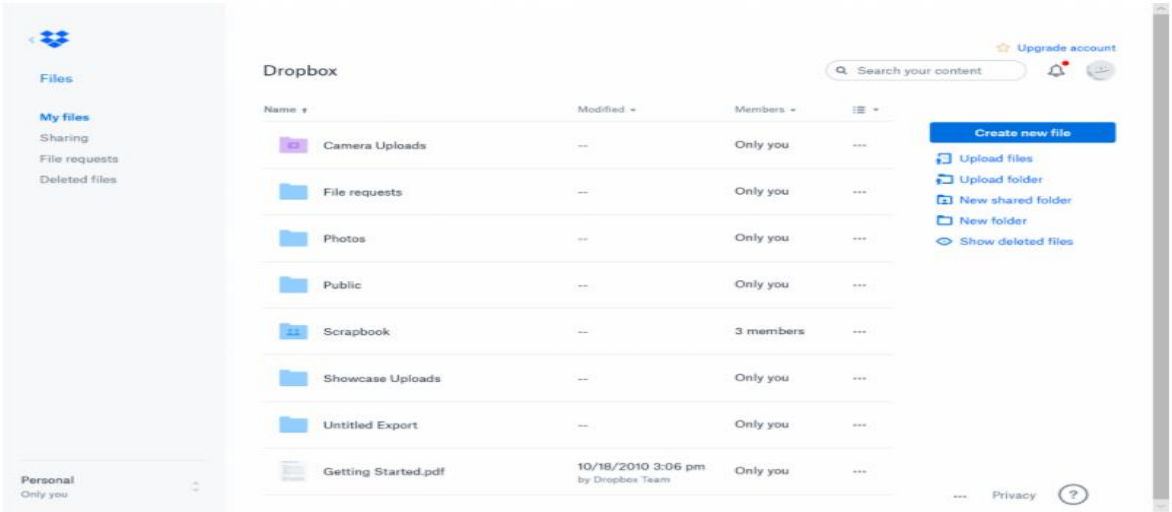
Seesaw: Un portafolio digital impulsado por los estudiantes y una herramienta sencilla para facilitar la comunicación con los padres. Seesaw es una herramienta excelente para que los profesores y los estudiantes compartan su trabajo sin problemas con sus



familias y es perfecto para mantener a todos informados.

Figura 9. Presentación de Seesaw. Fuente: Recuperado de https://web.seesaw.me/

Dropbox: Accede a sus archivos tanto dentro como fuera de la escuela, comparta su

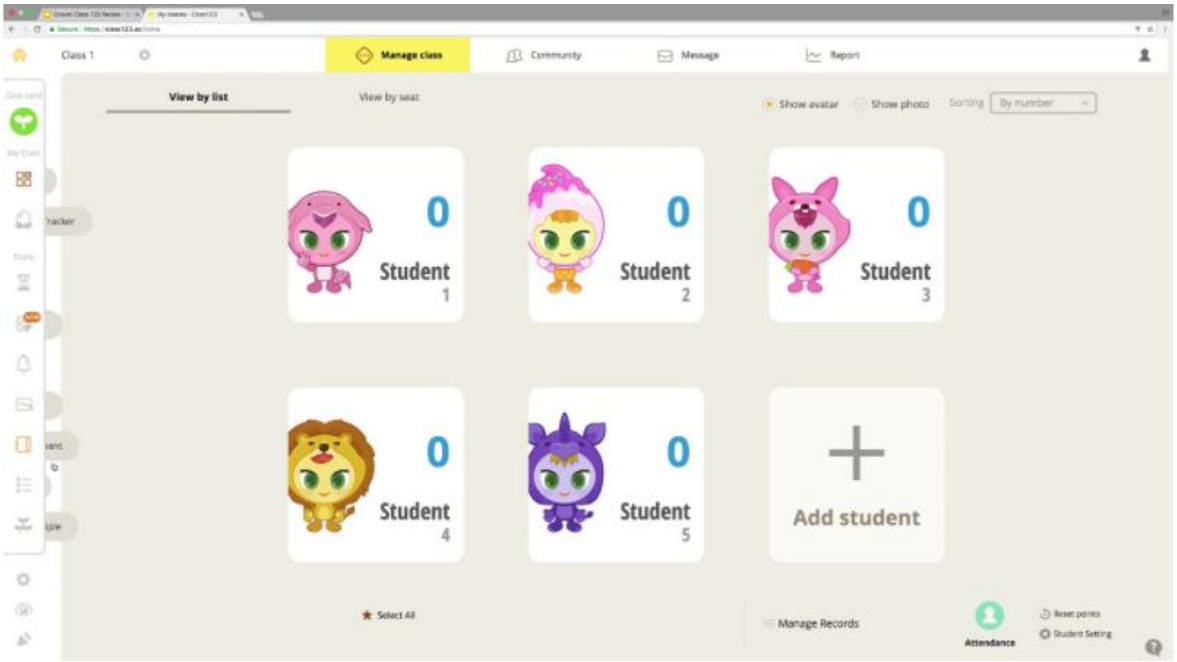


contenido y manténgase organizado.

Figura 10. Presentación de Dropbox. Fuente: Recuperado de https://https://www.dropbox.com/business/

Class123: Una herramienta gratuita de gestión del aula amada por millones.

Class123 tiene varios dispositivos útiles que incluyen un temporizador, una tabla de



asientos, un selector aleatorio, una pizarra, un rastreador de goles y más.

Figura 11. Presentación de Class123. Fuente: Recuperado de https://class123.ac/

3.2. Software y modelos de aprendizaje

Los software educativos están basados en modelos de aprendizaje y estos necesitan ser evaluados. “Los modelos orientados a la evaluación de software educativo, además de tener en cuenta los requisitos que demanda el software en general, también deben cumplir requisitos específicos para la enseñanza y el aprendizaje” (Rezende, 2013, p. 20).

Entonces, se puede entender que los modelos de evaluación de software educativo son un conjunto de requisitos requeridos en un software, que deben adoptar para su uso en el contexto educativo.

Existen varias metodologías para evaluar el software educativo. Se pueden destacar los siguientes: (a) la norma de productos de software ISO; (b) el modelo TUP (Tecnología, Usabilidad, Pedagogía), (c) el modelo Reeves y (d) el modelo de evaluación según Campos.

a. Producto de Software estándar ISO: La Organización Internacional de Normalización - ISO y la Comisión Electrotécnica - IEC9 desarrollaron un conjunto de estándares que tratan con la estandarización mundial actual para la calidad de productos de todo tipo de software, se dice que: “las diversas partes que componen la norma NBR ISO / IEC presente, en general, los instrumentos necesarios para realizar una evaluación de software, describiendo cómo medir cualitativa y cuantitativamente la calidad del producto” (Rezende, 2013, p. 33).

Según NBR ISO / IEC 9126-1 10, la calidad se refiere a "la totalidad de características de una entidad permite satisfacer necesidades explícitas e implícitas. Las explícitas cubren los objetivos, funciones y desempeño esperado, los implícitos son los principios básicos y obvios, necesarios para que el usuario cumpla su tarea” (Rezende, 2013, p. 30).

NBR ISO / IEC 2501011 sobre este estándar se dice que: “es un estándar que permite definir las características de calidad que todo software debe tener” (Rezende, 2013, p. 30). Este modelo tiene ocho características de calidad para ser abordadas por cualquier tipo de software. Son las siguientes:

- Funcionalidad: se revisa la integridad, corrección y adecuación funcional;
- Eficiencia: tiempo conductual, utilización de recursos, capacidad;
- Compatibilidad: convivencia, interoperable;
- Usabilidad: capacidad de aprendizaje, operatividad, protección contra errores del usuario, estética de la interfaz de usuario, accesibilidad;

- Fiabilidad: madurez, tolerancia a fallos, recuperabilidad;
- Seguridad: Confidencialidad, integridad, no repudio, responsabilidad, autenticidad;
- Modularidad: modularidad, reutilización, capacidad de análisis, capacidad de prueba;
- Portabilidad: adaptabilidad, inestabilidad, sustitución.

El uso de la norma ISO / IEC 25010 establece las características y subcaracterísticas de calidad para una evaluación de software, permitiendo verificar los atributos necesarios para cada tipo de software. Las características y subcaracterísticas, definidas por los modelos ISO / IEC, representan una referencia relevante para los estudios sobre evaluación de la calidad del software en general, independientemente de su aplicación.

b. Modelo TUP: El modelo TUP propone un modelo denominado TUP, a partir de las siglas en inglés Tecnología, Usabilidad y Pedagogía, surgido de la necesidad de crear un modelo de evaluación de software educativo, que permitió a los profesionales de la educación seleccionar software de acuerdo con un conjunto estructurado de criterios definidos. “Este modelo resalta la importancia de evaluar y seleccionar cuidadosamente el software educativo, trabajando de manera interdisciplinaria, abarcando aspectos de tecnología, usabilidad y pedagogía, integrados en un método de evaluación” (Rezende, 2013, p. 21).

En el acto de evaluar un software educativo es importante analizar los aspectos de la tecnología, así, Rezende (2013) señala aspectos importantes en el análisis de un software educativo relacionado con la tecnología, así como: disponibilidad, compatibilidad, accesibilidad, aspectos organizativos y fiabilidad.

La disponibilidad se refiere a que el software esté disponible durante el mayor tiempo posible para su uso, durante un período determinado, por ejemplo, el software debe estar disponible 24/7, es decir, debe estar disponible cuando el usuario lo necesite. Otro

aspecto importante es la compatibilidad, significa que el software debe funcionar en diferentes sistemas operativos.

El tercer atributo es la accesibilidad, según Rezende (2013) el software debe atender a diferentes usuarios con diferentes perfiles y necesidades, por lo que se puede entender que diferentes usuarios tienen diferentes necesidades y el software debe diseñarse de acuerdo a estas necesidades.

El siguiente atributo son los aspectos organizativos, que engloban las cuestiones de planificación, seguimiento e integración de las tareas soportadas por el software. Finalmente, la confiabilidad generalmente se define por la seguridad, confidencialidad y autenticidad del software.

En un análisis de software educativo, también se deben evaluar las características de usabilidad, que es la pregunta relacionada con qué tan bien los usuarios pueden utilizar la funcionalidad definida por el sistema de software. Según Rezende (2013), se asocia a cinco atributos para evaluar la usabilidad.

La primera es la facilidad, el software debe tener en cuenta que su

Los usuarios no necesitan tener conocimientos específicos sobre el software, por lo tanto, la necesidad de instrucciones y opciones claras y objetivas para aclarar dudas y ayudar en el uso del software, con la existencia de estos elementos, se traducirá en una facilidad para el usuario en el uso del software y llevar a cabo sus tareas, aprovechando al máximo el software.

El siguiente atributo es la interacción, se dice que está vinculado a "la forma en que el usuario interactúa con el software, comprende los comandos y realiza las tareas de interés" (Rezende, 2013, p. 25). Por lo tanto, el software siempre debe comunicarse con el usuario, cada elemento presente en la interfaz del software (iconos, botones, sonidos, palabras) tiene el potencial de comunicar algo.

El tercer atributo es la navegación, se refiere a los “caminos requeridos para explorar el sistema” (Rezende, 2013, p. 25), es decir, el software debe informar a los usuarios de su ubicación en el software e indicar cuáles son los caminos ya tomados y cuántos todavía están disponibles.

El siguiente atributo es la memorización, se refiere a la capacidad de realizar un seguimiento del punto donde se encontraba cada usuario en el momento de la interrupción del uso del software, asegurando así la posibilidad de continuar con el uso del software en otro momento.

El último atributo en relación a la usabilidad es la estética y el audio, el software debe tener una “visualización agradable y recursos de audio adecuados, que hagan satisfecho al usuario” (Rezende, 2013, p. 27), es decir, el software debe tener una estética diseño adecuado, haciendo uso de texto bien distribuido, imágenes y animaciones relevantes al contexto propuesto en el software, los efectos de sonido oportunos, sin embargo, no deben afectar la atención del usuario durante la realización de la actividad propuesta en el software.

El software educativo tiene características que lo diferencian del resto de software, ya que su énfasis está en el aprendizaje. La evaluación de la calidad de un software educativo debe tener en cuenta, principalmente, las características relacionadas con la calidad didáctico-pedagógica. En este aspecto, los atributos más importantes a considerar según Rezende (2013), son: contexto, tarea, herramienta, motivación y estructura pedagógica.

c. El modelo Reeves es generado por el profesor Thomas C. Reeves y ha desarrollado y evaluado numerosos programas de aprendizaje interactivo para la educación y la formación. “Sus intereses de investigación incluyen: evaluación de tecnología educativa, modelos mentales y multimedia interactivos, entornos de aprendizaje auténticos. Sus áreas

de enseñanza incluyen la evaluación de programas y el diseño instruccional" (Rezende, 2013, p. 34).

En 1994 se propuso el modelo Reeves, que "ha sido ampliamente utilizado en evaluaciones de software, en diferentes dominios, abordando principalmente software interactivo y con el uso de recursos multimedia" (Freitas y Kirner, 2013, p. 10).

El modelo define dos enfoques para evaluar el software educativo. El primer enfoque está relacionado con aspectos pedagógicos y se basa en catorce criterios. El segundo enfoque evalúa la calidad del software, refiriéndose a aspectos de la interacción humano-computadora, utilizando diez criterios

d. Modelo de evaluación de Campos (Frescki, 2008): Estas investigaciones están relacionadas con Educación, Ambientes Virtuales Interactivos, Evaluación, Educación a Distancia.

Este modelo (Frescki, 2008) trata de un manual de evaluación de la calidad del software educativo, con el objetivo de brindar pautas para desarrolladores y usuarios. Es un modelo de evaluación es importante para cualquier tipo de software educativo, como ejercicio y práctica, tutorial, simulación, investigación, juegos, entre otros.

El modelo está compuesto por factores, subfactores, criterios, procesos de evaluación. "La evaluación se realiza a través de una investigación de campo con los docentes, donde se tiene en cuenta el orden de importancia que los docentes atribuyen a los criterios más generales" (Frescki, 2008, p.18). Se puede observar que esta evaluación se acerca más a la mirada del docente, donde éste hace el juicio en relación al software.

Aplicación didáctica

Sesión de Aprendizaje

Conociendo el Software educativo Hot Potatoes

Datos Institucionales

1. Institución Educativa :
2. Área Curricular : Educación por el Trabajo – Computación
3. Componente : Formación Ocupacional Modular.
4. Ciclo : V
5. Grado y Sección : A
6. Tiempo : 2 horas pedagógicas.
7. Profesor (A) : Yover Arando
8. Tema Transversal : Educación para el trabajo.

I. Expectativa de Logro**:**

Finalizando la clase el estudiante deberá manejar y operar el software Hot Potatoes, de acuerdo al perfil de los cursos y de los estudiantes.

Contenidos	Aprendizajes Esperados	Valores / Actitudes
1. Anteriores ➢ Explicación sobre el software educativo. 2. Recientes ➢ Preparación e instalación del software Hot Potatoes ➢ Adecuación del Software Hot Potatoes ➢ Operación y resolución de problemas utilizando el software Hot Potatoes	Instalación, preparación y operación del software Hot Potatoes para utilízalo como material didáctico.	Disciplina Respeta a sus compañeros en las opiniones que viertan. Solidario Permite ser solidario en sus aprendizajes así como colaborar en los aprendizajes de sus compañeros. Responsable Es responsable con las actividades y tareas de clase.

II. Organización de los Aprendizajes

III. **Secuencia Didáctica:**

Situación de aprendizaje	**Estrategias didácticas**	**Recursos**	**Tiempo**	**Evaluación**		
				Criterio	**Indicador**	**Instrumento**
INICIO Situación problemática didáctica Recuperación de saberes previos Conflicto cognitivo	Saluda, prepara el aula en aspectos de orden y limpieza y comprueba la puntualidad y asistencia de los alumnos. Presenta las normas del trabajo. Organiza adecuadamente el modo de trabajo. Se pregunta ¿Cuál es la mejor forma de enseñar a sus alumnos? ¿Qué software educativo aplica y cual le parece mejor? Se decide con emplear el software educativo Hot Potatoes El profesor responde el software facilitara el aprendizaje	Palabras	02 min. 03 min. 02 min. 03 min.	Actitud frente el tema	Ayuda y coopera con sus compañeros Muestra interés en sus aprendizajes Instala el software educativo Hot Potatoes para poder ejecutarla. Configura el software educativo Hot Potatoes para facilitar su uso y su interfaz intuitiva.	Registro de los alumnos
PROCESO Acciones Formulación Argumentación Institucionalización	Se proporciona el tutorial del software Hot Potatoes para ser revisado por los alumnos. Se plantean respuestas. Se argumentan algunas respuestas y comunica a sus compañeros. Se conoce el software. Se instala, se configura y se utiliza el software educativo Hot Potatoes	Plumones Mota Computador Tutoriales	80 min.		Utiliza el software educativo Hot Potatoes para organizar tus tareas.	Lista de cotejo
SALIDA Extensión Metacognición	Se resuelven actividades aplicativas. Se toma una práctica sobre el tema tratado. Se deja ejercicios planteados por desarrollar como tarea. ¿Qué aprendí hoy? ¿Me sirve este conocimiento adquirido?	Plumones Mota Computador Tutoriales	10 min.			

V. Bibliografía

http://hotpot.uvic.ca Software educativo. Metodología y criterio para su elaboración y evaluación. Mg. Mirtha Ramos

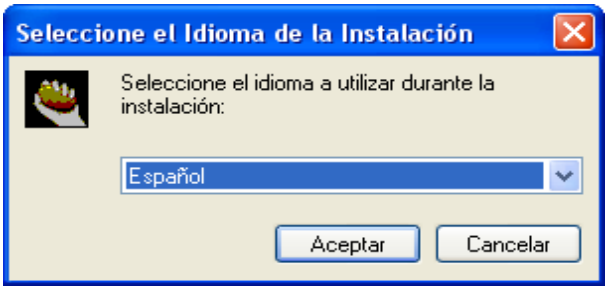
Guía de Aprendizaje

HOT POTATOES

Este software educativo permite construir herramientas en formato HTML sin programar. Obtenemos cinco tipos diferentes de ejercicios para que lo utilicen los alumnos.

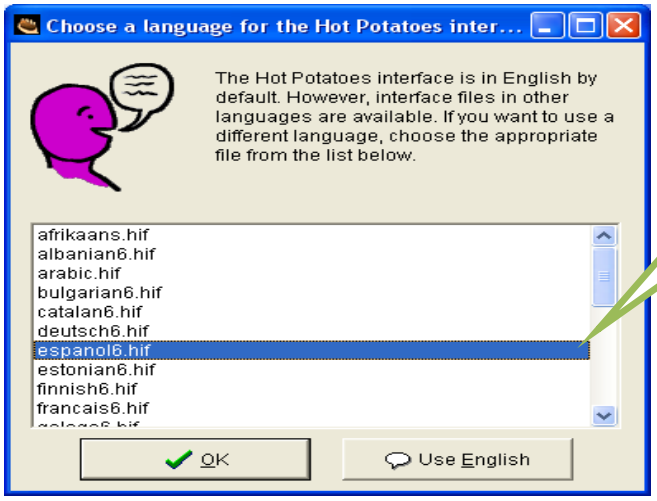
INSTALACIÓN

Su instalación es muy fácil:

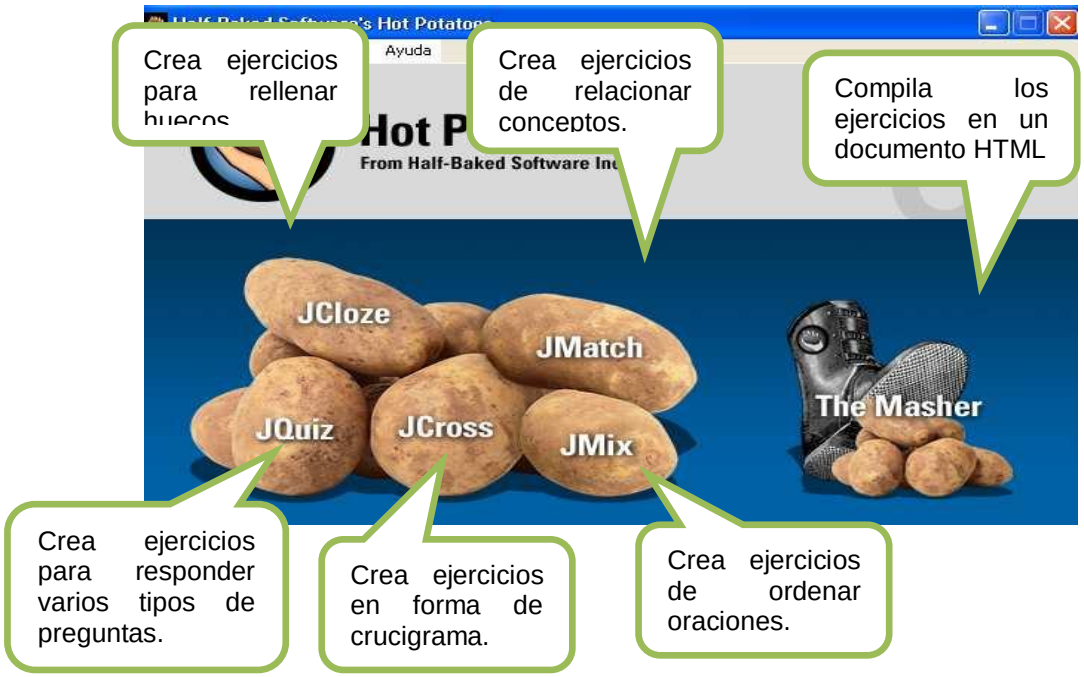


Luego aparece la ventana de bienvenida y nos pide hacer clic en siguiente para continuar.

INGRESANDO A HOT POTATOES



Al ingresar al programa lo primero que nos pide es seleccionar el idioma con el cual vamos a trabajar. Escogemos español y aceptamos.



Crea ejercicios para rellenar huecos

Crea ejercicios de relacionar conceptos.

Compila los ejercicios en un documento HTML

Crea ejercicios para responder varios tipos de preguntas.

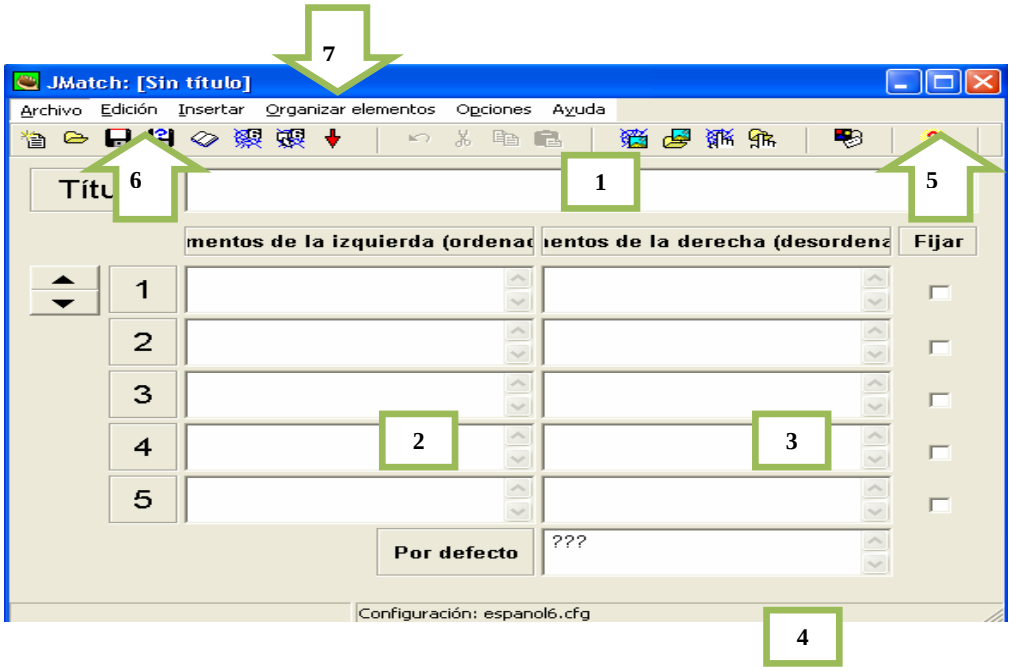
Crea ejercicios en forma de crucigrama.

Crea ejercicios de ordenar oraciones.

1. JMATCH EJERCICIOS PARA RELACIONAR

Se escoge el tema a tratar y relacionar. Buscamos y damos Clic en ARCHIVO / Guardar como.

Nos va a permitir relacionar conceptos con autores.



1.- Corresponde al título.

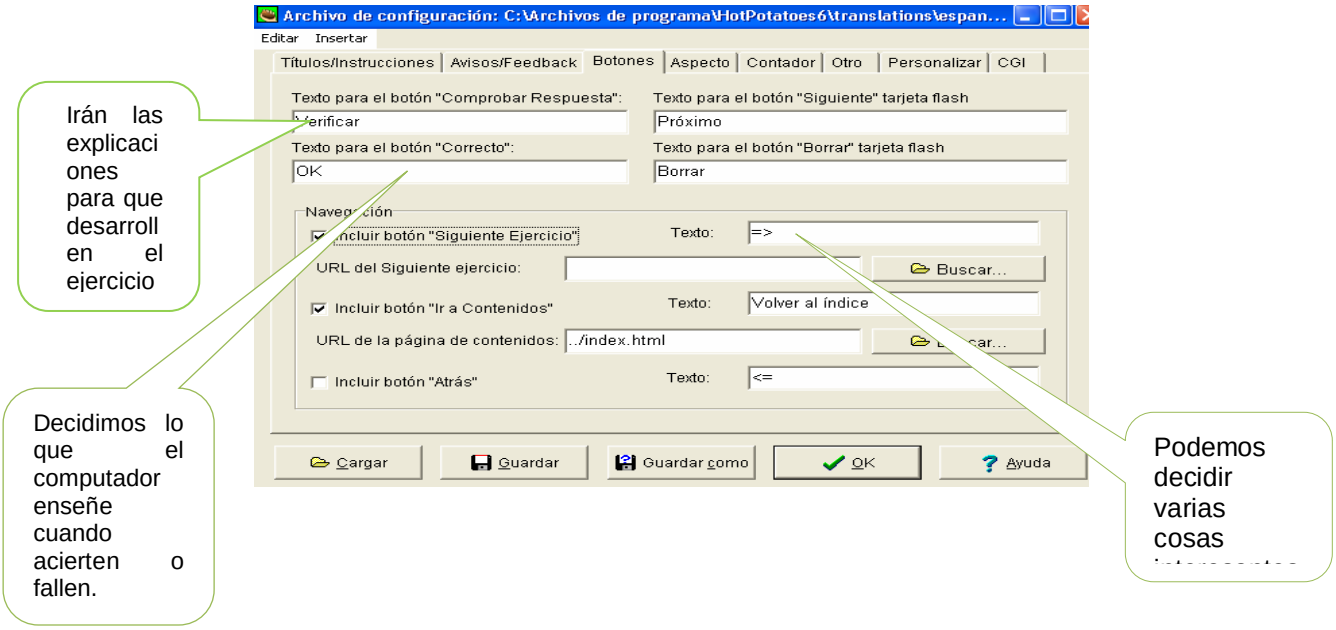
2 y 3.- Es la lista de palabras pueden ser palabras con palabras o con imágenes.

4.- No es necesario tocarlo.

5.- Se especifican los materiales educativos a realizar.

6.- Guarda los cambios realizados.

7.- Exporta el ejercicio a una página web;



Síntesis

Cualquier aplicación o programa de computadora que tenga un propósito educativo se incluye en la categoría de software educativo. Es un término amplio, por lo que es mejor explicarlo a través de su implementación práctica mostrando ejemplos de software educativo.

A mediados de la década de 1970, se empezaron a introducir algunos programas informáticos en el ámbito educativo. Este proceso se inició en las Universidades, donde en un seminario, impartido por E. Huggins, profesor de la Universidad de Dartmouth - Estados Unidos, se propuso el uso de computadoras en la enseñanza de la Física.

Un software educativo es un producto utilizado correctamente por la escuela, incluso si no ha sido producido con el propósito de ser utilizado en el sistema escolar. Muy independientemente de la finalidad para la que se creó el software, puede convertirse en software educativo, sin embargo, esto dependerá de cómo se utilizará en el contexto educativo y para qué fines.

Con el tiempo, los resultados obtenidos a través de la relación entre software y educación se han incrementado para quienes participan en los procesos de enseñanza y aprendizaje mediados por software educativo. Los jóvenes están acostumbrados a obtener información rápidamente y tienden a recurrir principalmente a fuentes digitales. Por estos comportamientos y actitudes Prensky los describe como Nativos Digitales, ya que 'hablan' el lenguaje digital desde que nacieron".

Apreciación crítica y sugerencias

Gran parte de la investigación educativa reciente se centra en la enseñanza y el aprendizaje dentro de las conversaciones en el aula. El Software Educativo proporciona recursos y, al mismo tiempo, enmarca y dirige las conversaciones de aprendizaje entre pequeños grupos de niños.

Básicamente, dos perspectivas educativas principales subyacen al software de autoevaluación: instruccionista y construccionista. El primero generalmente se centra en el maestro y el último está centrado en el alumno. Vale la pena señalar que no es estrictamente un compromiso de esto o lo otro como estas dos visiones teóricas no son necesariamente excluyentes entre sí en la educación.

Con el creciente uso de software educativo en las escuelas, los profesionales de la educación necesitan métodos e instrumentos que les permitan analizar la calidad del software que se utilizará en el proceso de enseñanza y aprendizaje. Así, este estudio tiene como objetivo sistematizar métodos e instrumentos para el análisis de software educativo, identificados en publicaciones, instituciones científicas nacionales.

Este trabajo se centra en los métodos e instrumentos de análisis del software educativo, con la intención de contribuir a la reflexión teórica y práctica educativa en el campo de la Pedagogía, colaborando con los docentes en formación y en la práctica profesional, que deseen adoptar en sus prácticas educativas software educativo y no sabe cómo seleccionar un software adecuado. En consecuencia, se espera colaborar para un aumento en el nivel de calidad del software educativo utilizado en las escuelas.

Referencias

Alessi, S. (2001). *Instrucción basada en computadora: métodos y desarrollo*. USA. Editorial Prentice-Hall.

Bork, A. (1985). *Computadoras personales para la educación*. España. Editorial Fila y Fila.

Burke, R. (1992). *Diseño de Software*. México. Editorial Prentice Hall.

Collazos C. y Guerrero L. (2013). *Diseño de Software Educativo*. México. Editorial McGraw Gill.

Costa, A. y Costa, E. (2013). *Contribuciones al desarrollo de software educativo basado en procesos centrados en el usuario*. EN WEB, Revista Iberoamericana de Educación Matemática y Tecnológica, v. 4, no. 2.

Croswell, E. (2005). *El diseño de la instrucción basada en computadora*. Panamá. Editorial Macmillan.

Dumas, J. (2008). *Diseño de interfaces de usuario para software*. México. Editorial Prentice-Hall.

Favaro, P. (2006). *Guía del educador sobre microcomputadoras y aprendizaje*. Argentina. Editorial La Nación.

Freitas, L. y Kierner, T. (2013). *Hacia el éxito en el uso de software educativo para la enseñanza y el aprendizaje de las ciencias*. Revista de informática aplicada, vol. 9, n° 1.

Freski, F. (2008). *Evaluación de la calidad de software educativo para la enseñanza de álgebra.* Monografía, Universidad Estatal del Oeste de Paraná – Cascavel, Brasil.

Geisert, P. y Futrell, M. (2000). *Maestros, computadoras y plan de estudios: Microcomputadoras en el aula*. Chile. Ed. Alva y Tocino.

González, E. (2016). *Estrategias para el diseño y desarrollo de software educativo*. México. Editorial SIGESA.

Hannafin, M. (2008). *El Diseño, Desarrollo y Evaluación de software instruccional.* USA. Editorial Gulf Publishing Company.

Keller, J. (2008). *Uso del modelo de motivación ARCS en Diseño de material didáctico*. Sudáfrica. Editorial Hills Dale.

Knowles, M. (1990). *El aprendiz adulto: una especie desatendida*. USA. Editorial Gulf Publishing Company.

Lapolli, F. (2009). *Modelo de desarrollo de objetos de aprendizaje basado en metodologías ágiles*. En: XX Simposio Brasileño de Información en Educación, Florianópolis. Porto Alegre: Sociedad Brasileña de Computación.

Larkin, J. y Chibay, R. (1992). *Instrucción asistida por computadora y Sistemas de tutoría inteligente: objetivos compartidos y enfoques complementarios*. México. Editorial Erlbaum.

Marqués, P. (2009), *Evaluación y selección de software educativo.* España. Universidad Autónoma de Barcelona

Rezende, C. (2013). *Modelo de evaluación de la calidad de software educativo para la enseñanza de las ciencias*. Disertación (Maestría en Enseñanza de las Ciencias) Universidad Federal de Itajubá. Brasil.

Richards, T. y Fukuzawa, J. (1989). *Una lista de verificación para la evaluación de Sistemas de creación de material didáctico*. Brasil. Publicaciones Tecnológicas en Educación. Recuperado de http:/www.educabras.gob.br/doc5478cde

Shlechter, T. (1991). *Problemas y promesas de la formación basada en computadora*. Norwood, USA. Nueva Jersey: Ablex Publishing Corporation.

Steinberg, E. (1991). *Instrucción asistida por computadora: una síntesis de la teoría, Práctica y Tecnología*. Argentina. Editorial Asociados, Inc.

Torres, G. (2014). *El Software Libre*. España. Editorial Paidós.

Venezky, R. y Osin, L. (1991). *El diseño inteligente asistido por computadora Instrucción*. Venezuela. Editorial Grupo Publicaciones La Mar.