



UNIVERSIDAD NACIONAL
AUTÓNOMA DE
MÉXICO

UNIVERSIDAD NACIONAL AUTÓNOMA
DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES CUAUTITLÁN

**DESARROLLO DE UNA APLICACIÓN WEB
INTERACTIVA PARA EL ESTUDIO Y EDICIÓN
DE GRAFOS**

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

LICENCIADO EN INFORMÁTICA

P R E S E N T A:

JAVIER ALEJANDRO OLGUÍN MENDOZA

ASESOR: L.C. CARLOS PINEDA MUÑOZ

CUAUTITLÁN IZCALLI, ESTADO DE MÉXICO 2018



Universidad Nacional
Autónoma de México

Dirección General de Bibliotecas de la UNAM

Biblioteca Central



UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.



UNIVERSIDAD NACIONAL
AUTÓNOMA DE
MÉXICO

**FACULTAD DE ESTUDIOS SUPERIORES CUAUTITLÁN
UNIDAD DE ADMINISTRACIÓN ESCOLAR
DEPARTAMENTO DE EXÁMENES PROFESIONALES**

U. N. A. M.
FACULTAD DE ESTUDIOS
SUPERIORES-CUAUTITLÁN

ASUNTO: VOTO APROBATORIO



**M. en C. JORGE ALFREDO CUÉLLAR ORDAZ
DIRECTOR DE LA FES CUAUTITLÁN
PRESENTE**

**ATN: I.A. LAURA MARGARITA CORTAZAR FIGUEROA
Jefa del Departamento de Exámenes Profesionales
de la FES Cuautitlán.**

Con base en el Reglamento General de Exámenes, y la Dirección de la Facultad, nos permitimos comunicar a usted que revisamos el: **Trabajo de Tesis**

"Desarrollo de una aplicación web interactiva para el estudio y edición de Grafos"

Que presenta el pasante: **JAVIER ALEJANDRO OLGUÍN MENDOZA**

Con número de cuenta: **30521608-9** para obtener el Título de la carrera: **Licenciatura en Informática**

Considerando que dicho trabajo reúne los requisitos necesarios para ser discutido en el **EXAMEN PROFESIONAL** correspondiente, otorgamos nuestro **VOTO APROBATORIO**.

ATENTAMENTE

"POR MI RAZA HABLARÁ EL ESPÍRITU"

Cuautitlán Izcalli, Méx. a 17 de noviembre de 2017.

PROFESORES QUE INTEGRAN EL JURADO

	NOMBRE	FIRMA
PRESIDENTE	L.C. Carlos Pineda Muñoz	
VOCAL	MCC. María Araceli Nivón Zaghi	
SECRETARIO	MCC. Bárbara Verónica Buendía Hernández	
1er. SUPLENTE	L.I. Mauricio Jaques Soto	
2do. SUPLENTE	L.I. Rosalba Nancy Rosas Fonseca	

NOTA: los sinodales suplentes están obligados a presentarse el día y hora del Examen Profesional (art. 127).

Contenido

Introducción	5
Capítulo 1 “Teoría de Grafos”	7
1.1 Conceptos Generales	8
1.2 Grafos simples dirigidos	13
1.3 Grafos simples no dirigidos	15
1.4 Subgrafos	17
1.5 Ejemplo de la aplicación de los grafos en la Informática	17
CAPÍTULO 2 “Software Libre”	20
2.1 Conceptos Generales	21
2.2 Libertades	23
2.3 Tecnologías que utilizan el software libre	25
CAPÍTULO 3 “Tecnologías Web”	27
3.1 Estándares	28
3.1.1 HTML	28
3.1.2 CSS.....	29
3.2 Lenguajes de programación web	31
3.2.1 JAVASCRIPT.....	31
3.3 Servidor	33
3.3.1 Servidor FTP	33
3.3.2 Servidor Web	34
3.3.3 Modelo Cliente - Servidor.....	35
3.4 Aplicaciones Web	36
3.4.1 Ejemplos de aplicaciones web	38
CAPÍTULO 4 “D3: Librería de JavaScript”	41
4.1 Introducción a D3	41
4.2 SVG	44
4.2 Javascript y patrones	45
4.2.1. Tipos de patrones	45
4.2.2. Utilidad de los patrones.....	46
4.2.3. Patrones en la aplicación web interactiva de grafos	47
4.3 Aplicaciones de D3	55
CAPÍTULO 5 “Desarrollo de la Aplicación”	59
5.1 Metodología	59

5.1.1. Formulación	61
5.1.1.2. Objetivo.....	61
5.1.1.3. Software a utilizar	62
5.1.2. Planificación.....	62
5.1.3. Análisis.....	63
5.1.4. Modelización.....	68
5.1.5. Generación de páginas	69
5.1.6. Test.....	70
5.1.7. Evaluación del cliente	70
5.2 Aplicación	71
5.2.1 Desarrollo de le sección “Edición Libre de Grafo Dirigido”	71
5.2.2. Desarrollo de le sección “Edición Libre de Grafo No Dirigido”	73
5.2.3. Desarrollo de la sección “Ejercicios”	76
Conclusiones	94
Referencias.....	96

Introducción

El presente trabajo se describe como una aplicación web interactiva para la enseñanza de la teoría de grafos, la propuesta está dirigida a nivel licenciatura concretamente a la Facultad de Estudios Superiores Cuautitlán. El autor enfocó la programación empleada al desarrollo web para responder a la tendencia actual: internet.

El software en general se enfoca cada vez más hacia la red de internet y las bondades que ofrece. Esta tecnología comienza a tener una penetración innegable en la sociedad, se está convirtiendo en un medio de comunicación primordial para muchas personas y es una herramienta más de trabajo durante la formación del estudiante.

Casi todos los usuarios, consciente o inconscientemente, han hecho uso de programas que estén ligados a internet, se encuentran familiarizados con el entorno y es algo que se aprovechó para la realización de la aplicación. Uno de los objetivos perseguidos durante el desarrollo del software fue que en lugar de instalar un programa en los equipos de los alumnos (con las respectivas restricciones que implica: sistemas operativos compatibles, capacidades técnicas de las máquinas, entre otros); el estudiante podrá acceder de manera remota desde distintos lugares y usando diferentes computadoras pues la aplicación no existe en el equipo desde el que se ingresa, sino que está en un servidor, o lo que muchos nombran como “la nube”.

Por su parte, la importancia del estudio de la teoría de grafos durante el desarrollo universitario es una parte clave para la resolución de problemas a los que se enfrentará el alumno en su integración al mercado laboral. Los grafos intentan resolver situaciones muy cotidianas como puede ser la búsqueda del camino más corto entre dos puntos; este problema, que aparentemente es tan simple, es una parte importante del cálculo de la ruta que hacen los navegadores GPS, por mencionar un ejemplo. Otro problema “famoso” es el del viajante de comercio: encontrar la ruta más corta que pasa por un conjunto de ciudades y regresa a la ciudad de origen; en este caso, la fama viene de la complejidad de calcular la ruta óptima.

Al ser la teoría de grafos un tema tan importante, se consideró necesario crear un apoyo para el profesor y los alumnos que permita fortalecer los conocimientos obtenidos. La propuesta presentada en la tesis busca convertirse en una manera de reforzar dichos conocimientos y permitir, tanto al alumno como al profesor, evitar dejar el tema a la deriva en la formación del estudiante. El trabajo se enfoca en grafos dirigidos y no dirigidos que se pueden construir a través de una matriz de adyacencia.

Cabe mencionar que ya existe software dedicado a la teoría de grafos, está por ejemplo “yEd”¹, un programa de uso libre escrito en lenguaje Java® que se ejecuta en Windows®, Linux® y MacOS®, “yEd” puede ser usado para dibujar diferentes tipos de grafos, además de diagramas de flujo, mapas mentales y diagramas de entidad-relación. También existe el programa “Grafos”², un software desarrollado en .Net® por la Universidad Politécnica de Valencia, utilizado como apoyo para sus estudiantes; entre sus principales características permite exportar imágenes del trabajo realizado.

Sin embargo, el software propuesto en la tesis busca llevar este tipo de programas a la computación en la nube, evitando así la limitante actual de sólo poder trabajar en los dispositivos que tengan instalados los programas de grafos. Esto implica mover parte del trabajo desde el equipo de cómputo a servidores remotos, estos servidores almacenan los datos que se necesitan para trabajar de manera que es posible acceder a ellos desde cualquier computadora conectada a internet en cualquier parte del mundo.

Para tal fin, se utilizaron herramientas de software libre y el lenguaje de programación JavaScript (específicamente las librerías de “D3”³) principalmente. Hacer mención también del uso de otros estándares web durante el desarrollo como HTML y CSS para el aspecto del diseño y las vistas de la página web.

1 Yworks. (2017). “yEd – Graph Editor”. Consultado el 6 marzo de 2017. Disponible en: <https://www.yworks.com/products/yed>

2 Rodríguez, A. (2013). “Grafos - software para la construcción, edición y análisis de grafos”. Consultado el 6 de marzo de 2017. Disponible en: <http://arodrigu.webs.upv.es/grafos/doku.php?id=software>

3 Bostock, M. (2015). “D3.js - Data-Driven Documents”. Consultado el 6 de marzo de 2017. Disponible en: <https://d3js.org/>

Capítulo 1 “Teoría de Grafos”

Durante la formación académica del estudiante de la licenciatura en Informática, en su cuarto semestre, el plan de estudios⁴ contempla la enseñanza de la materia “Matemáticas Computacionales” a los alumnos de la carrera; el propósito de esto es, citando, “*proporcionar al estudiante los elementos que le permitan obtener soluciones de modelos matemáticos, utilizando la computadora, con el objeto de la toma de decisiones, expeditas y confiables.*”, y dentro de dicha materia se aborda el tema de la Teoría de Grafos prácticamente al final de éste semestre.

Dado lo anterior, se encontró que muchas veces por la falta de tiempo, éste tema se trata muy poco y sólo de manera introductoria, por ello se determinó proponer una aplicación web interactiva que favoreciera la enseñanza de la teoría de grafos a los alumnos a la vez que le otorga al profesor una herramienta más para impartir el tema.

Es así que tomando como base los temas que se enseñan en la materia “Matemáticas Computacionales”, se estableció delimitar la información que comprendería la aplicación web en cuanto a la teoría de grafos, siendo lo abordado:

- Conceptos generales: grafo, nodo, aristas, lazo, matriz de adyacencia
- Grafos Simples Dirigidos
- Grafos Simples No Dirigidos
- Subgrafos

Además, el propio plan de estudios propone el uso de la computadora durante la impartición de la materia, por ello, ocupar una aplicación web encaja perfectamente con dicho objetivo.

4 Facultad de Estudios Superiores Cuautitlán (2017). “Licenciatura en Informática”. Recuperado de: http://www.cuautitlan.unam.mx/licenciaturas/informatica/plan_estudios.html

1.1 Conceptos Generales

La teoría de grafos tiene su origen en el año 1736 cuando Leonhard Euler⁵ publicó un trabajo que daba seguimiento al problema conocido como “Los puentes de Königsberg”. Dicho problema surgió en la ciudad de Prusia (en la actualidad Kaliningrado, Rusia), donde el río Pregel recorre el poblado y también dos islas (de nombre Kneiphof) que están en el centro, causando que la ciudad quede dividida en cuatro partes: dos regiones a cada lado del río y las islas de en medio; todas ellas unidas por siete puentes. La siguiente imagen⁶ ejemplifica las regiones del lugar:

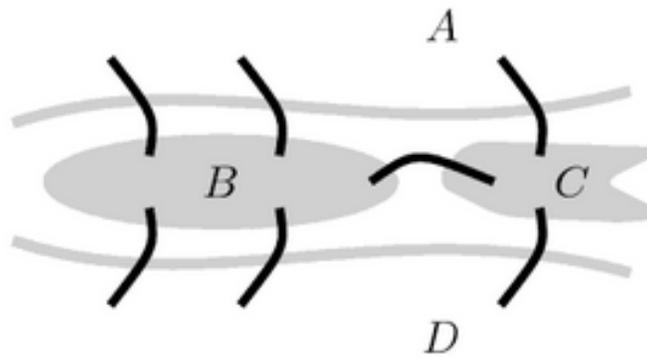


Figura 1. Ejemplo de los “puentes de Königsberg”.

Ante esta situación, muchas personas comenzaron a averiguar si era posible cruzar los siete puentes pasando solamente una vez por cada uno de ellos. Euler demostró que esto no era posible, y para probarlo elaboró un modelo con un grupo de puntos que representaban las áreas de tierra y líneas que simbolizaban los puentes; fue así como surgió el primer ejemplo de un grafo.

5 Leonhard Paul Euler (1707 – 1783) fue un matemático y físico suizo. Vivió en Rusia y Alemania la mayor parte de su vida y realizó importantes descubrimientos en áreas como el cálculo, análisis matemático y la teoría de grafos.

6 Imagen del libro: Vicites Rodríguez, A. (2014). Teoría de grafos. España: Paraninfo. Página 1 Sección 1.

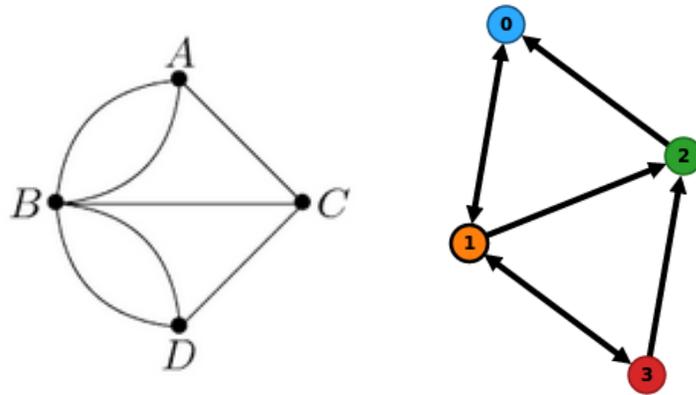


Figura 2. Imagen del grafo de Euler. A su lado está dicho grafo representado en la aplicación web.

Tomando como referencia la segunda imagen, las flechas indican el camino que se toma para recorrer los cuatro puntos y, como se puede notar, es necesario cruzar al menos dos veces por algunos enlaces para poder cumplir el ciclo.

A partir del trabajo de Euler, la teoría de grafos fue convirtiéndose en una útil herramienta para crear modelos en temas tan dispares como las matemáticas, la química, la física, las telecomunicaciones, internet, la economía, el estudio de probabilidades, temas de planificación, sistemas GPS, en redes neuronales, en programación, etc; propiamente en gran parte de la Informática aparecen los grafos, especialmente los grafos de árbol y los grafos dirigidos. Los diagramas de flujo, por ejemplo, son grafos dirigidos.

Citando a Héctor Cancela⁷, *“la eficacia de los grafos se basa en su gran poderío de abstracción y la muy clara representación de cualquier relación, lo que facilita enormemente tanto la fase de modelado como de resolución del problema”*. Por tal motivo, se han desarrollado gran variedad de algoritmos y métodos de resolución de problemas muy eficaces para la toma de decisiones.

7 Cancela, H. (2009). “Curso de Bioinformática”. Consultado el 8 de marzo de 2017. Disponible en: <https://www.fing.edu.uy/inco/grupos/bioinf/bioinfo1/index.html>

Enfocando la teoría de grafos a la informática, se puede elaborar un sencillo ejemplo⁸: Representar una red de comunicaciones donde la colocación ideal de los elementos evite interrupciones accidentales, en otras palabras, que no haya desconexión de la red.

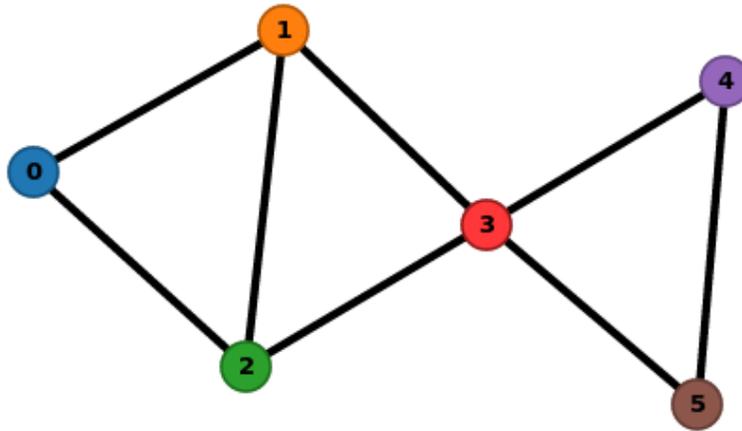


Figura 3. Grafo del ejemplo descrito.

El grafo propuesto expone que no existe ninguna línea que, al ser eliminada, desconecte la red.

Se puede definir al *grafo*⁹ como una representación abstracta de las relaciones (denominadas “aristas” o “arcos”) entre una serie de puntos (llamados “nodos” o “vértices”). Por su parte, la *teoría de grafos*¹⁰ es la disciplina que estudia los diferentes tipos de grafos, sus propiedades y los múltiples problemas relacionados a su análisis, manipulación y visualización.

Un *nodo*¹¹ es la unidad básica de la que están formados los grafos, generalmente se representan por un punto. Una *arista*¹² es la relación (unión) entre dos nodos de un grafo. Dos nodos son *adyacentes*¹³ si hay una arista que los conecte. Si un nodo es conecta consigo mismo se dice que existe un *lazo*¹⁴; y cuando

8 Cancela, H. (2009). “Material teórico sobre grafos”. Consultado el 8 de marzo de 2017. Disponible en: <https://www.fing.edu.uy/inco/grupos/bioinf/bioinfo1/teorico/grafos.pdf>

9 Vieites Rodríguez, A. (2014). Teoría de grafos. España: Paraninfo. Sección 1 Página 2.

10 Universidad de Pamplona. (2012). “Teoría de grafos”. Consultado el 8 de marzo de 2017. Disponible en: http://www.unipamplona.edu.co/unipamplona/portalIG/home_23/recursos/general/11072012/grafos3.pdf

11 Caicedo, A. (2010). Introducción a la Teoría de Grafos. Armenia. Ediciones Elizcom.

12 Caicedo, A. (2010). Introducción a la Teoría de Grafos. Armenia. Ediciones Elizcom.

13 Vieites Rodríguez, A. (2014). Teoría de grafos. España: Paraninfo.

14 Vieites Rodríguez, A. (2014). Teoría de grafos. España: Paraninfo.

un nodo no está conectado a ningún otro se denomina como *nodo aislado*¹⁵. También hay casos donde dos nodos están unidos por más de una arista, a esto se le conoce como *aristas múltiples*¹⁶ o *paralelas*.

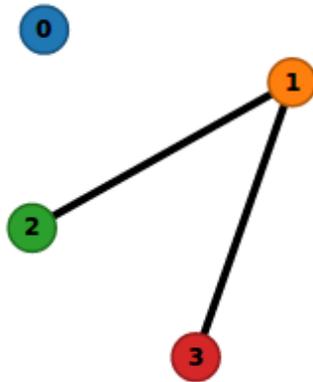


Figura 4. Ejemplo de nodos (representados por puntos) y aristas (líneas).

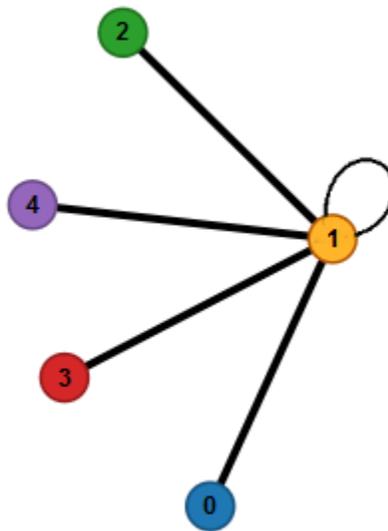


Figura 5. Ejemplo de conexión entre nodos y un lazo.

Existen los *grafos simples*¹⁷ y los *multigrafos*¹⁸, los grafos simples son cuando no hay ni lazos ni aristas paralelas; los multigrafos sí incluyen estas dos propiedades. La tesis se enfocará principalmente en los grafos simples.

15 Vieites Rodríguez, A. (2014). Teoría de grafos. España: Paraninfo.

16 Vieites Rodríguez, A. (2014). Teoría de grafos. España: Paraninfo.

17 Caicedo, A. (2010). Introducción a la Teoría de Grafos. Armenia. Ediciones Elizcom.

18 Caicedo, A. (2010). Introducción a la Teoría de Grafos. Armenia. Ediciones Elizcom.

Un grafo simple es completo cuando cada nodo es adyacente a todos los demás, es decir, que todos los nodos se conecten entre ellos, ejemplo:

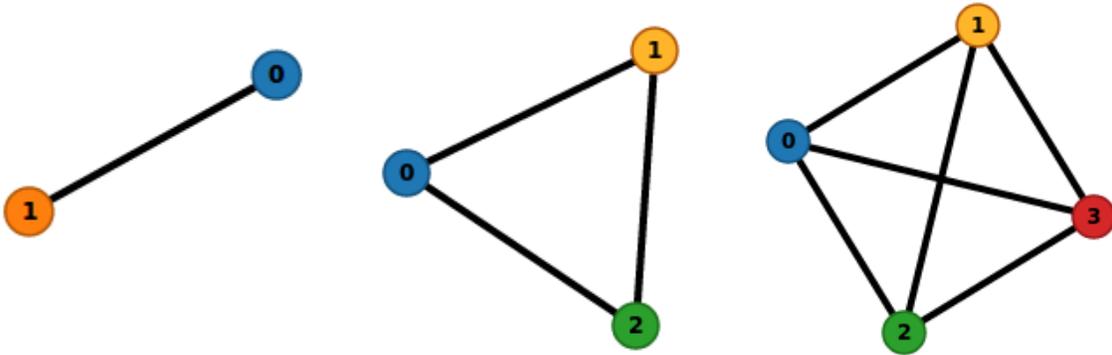


Figura 6. Los tres grafos tienen a sus respectivos nodos conectados entre sí.

Los grafos simples pueden ser representados por matrices, mismas a las que se les conoce como *matriz de adyacencia*¹⁹. Una matriz de adyacencia no es más que una matriz cuadrada de n filas por n columnas; n representa el número de nodos del grafo. Para construirlo, se agrega el valor 1 cuando dos nodos están conectados y el valor 0 cuando no es así:

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

¹⁹ Caicedo, A. (2010). Introducción a la Teoría de Grafos. *Armenia*. Ediciones Elizcom.

Un ejemplo de una matriz de adyacencia y su respectivo grafo es el siguiente:

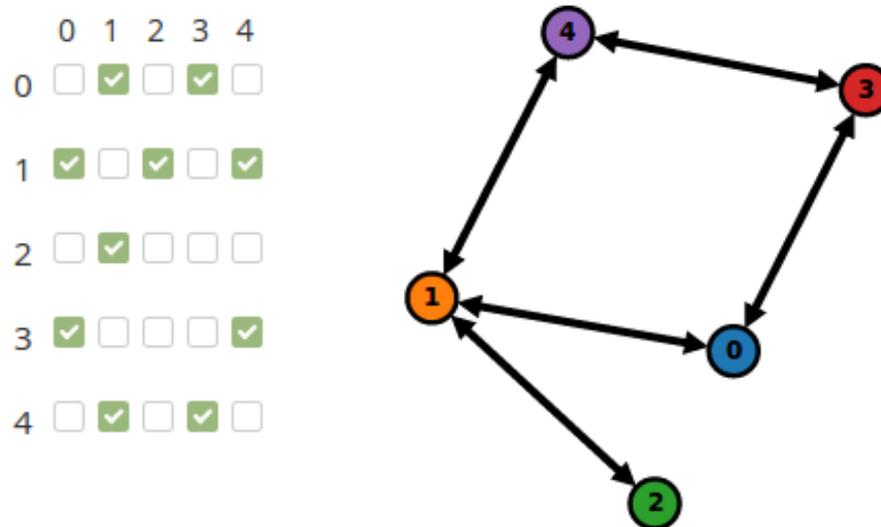


Figura 7. Imagen ejemplo generada en la aplicación web.

En la ilustración anterior, la matriz de adyacencia está del lado izquierdo y su grafo del lado derecho; se generaron una serie de checkbox²⁰ para representar los 1 y 0, si tiene una selección verde (“palomita”) equivale a 1 y en caso contrario equivale a 0.

1.2 Grafos simples dirigidos

Los *grafos simples dirigidos*²¹ son aquellos cuyas aristas están orientadas (generalmente representadas por flechas), es decir, los nodos tienen una dirección definida. También se les conoce como “dígrafos”.

20 Checkbox hace referencia a un elemento común dentro del lenguaje Javascript que permite definir cuando algo está seleccionado o no, en este caso es muy útil para trabajarlo con la representación de las matrices de adyacencia.

21 Vieites Rodríguez, A. (2014). Teoría de grafos. España: Paraninfo.

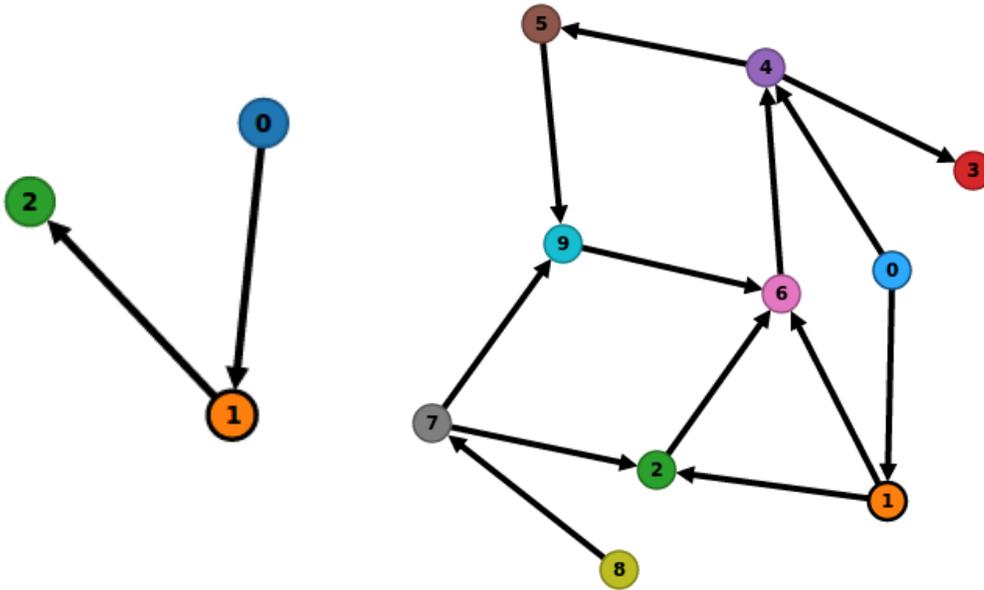


Figura 8. Ejemplos de grafos dirigidos.

Como se puede observar, en los grafos dirigidos los nodos se representan como puntos, por su parte las aristas se representan como flechas que parten de un vértice (el primero de cada par) y llegan a otro (el segundo de cada par).

Además, todo grafo dirigido puede representarse con una matriz de adyacencia. Si un nodo se dirige hacia otro se escribe 1, y cuando no es así se coloca 0.

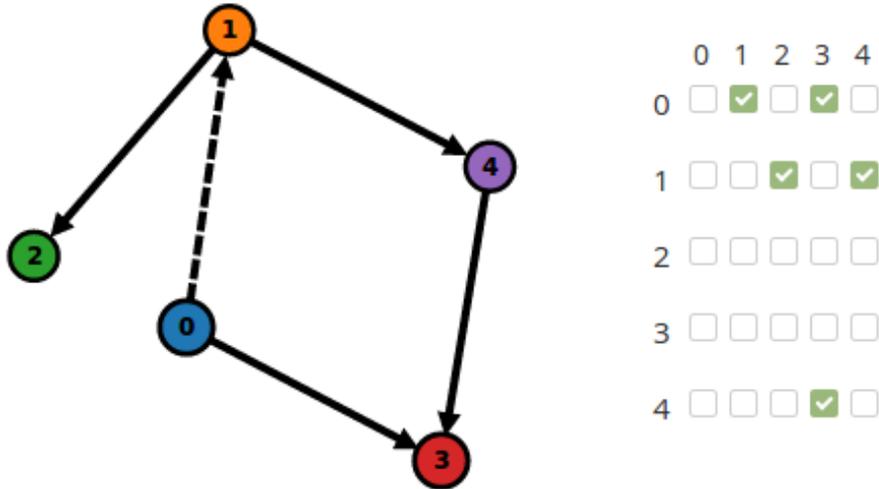


Figura 9. Ejemplo generado con la aplicación web de un grafo dirigido y su matriz de adyacencia.

1.3 Grafos simples no dirigidos

Por otro lado, los *grafos simples no dirigidos*²² no tienen la dirección de sus aristas definida, es decir, están sin orientar y no se puede determinar si hay una dirección concreta hacia un sentido, por ejemplo de A a B ($A \rightarrow B$) o sólo de B a A ($B \rightarrow A$). Es por ello que en lugar de ocupar una flecha sólo se marca el enlace con una línea recta entre los nodos.

²² Vieites Rodríguez, A. (2014). Teoría de grafos. España: Paraninfo.

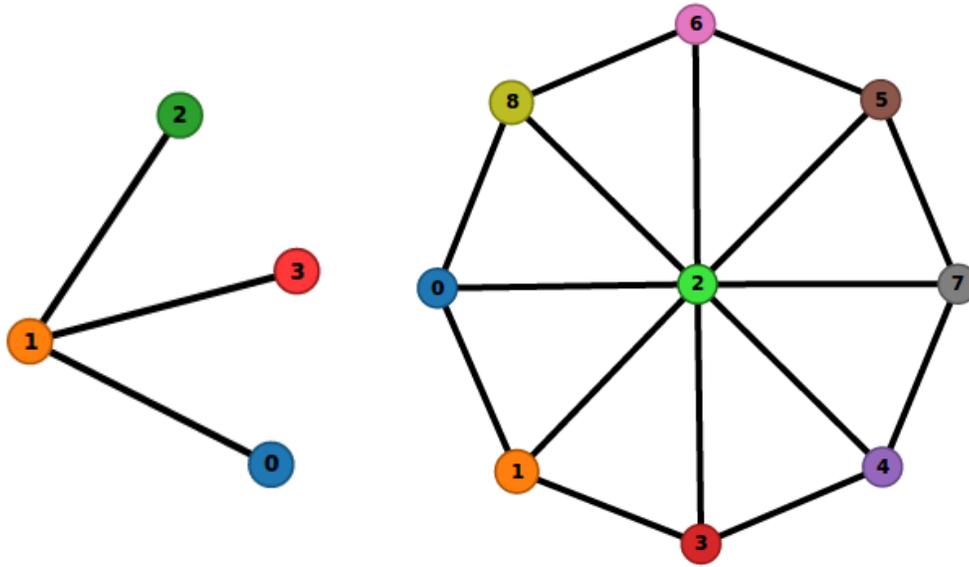


Figura 10. Ejemplos de grafo no dirigido.

Los grafos no dirigidos también pueden representarse en una matriz de adyacencia, el 1 se coloca cuando hay conexión y 0 cuando no es el caso.

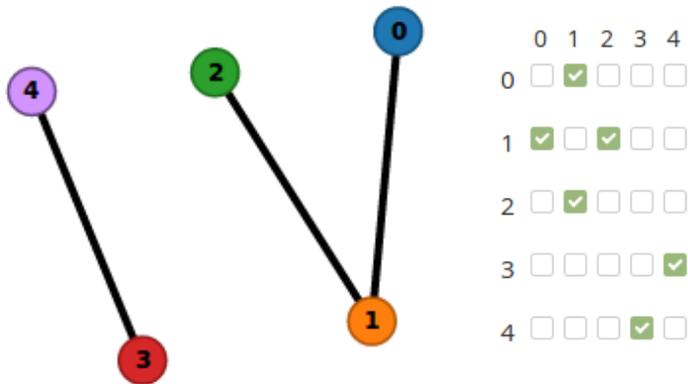


Figura 11. Ejemplo realizado con la aplicación web de un grafo no dirigido y su matriz de adyacencia.

1.4 Subgrafos

Un *subgrafo*²³ es un grafo cuya definición de nodos y aristas están basados en un subconjunto de un grafo “padre”.

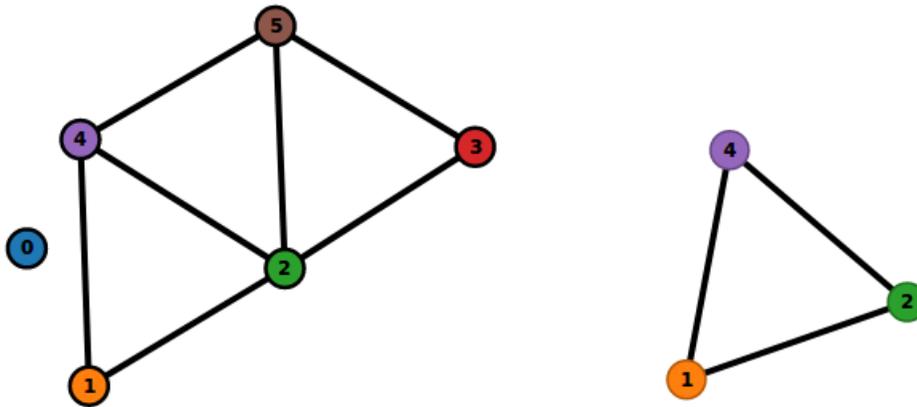


Figura 12. Del lado izquierdo hay un Grafo “Padre”, al lado derecho está el Subgrafo generado.

Todos los grafos pueden generar algún tipo de subgrafo, esto es porque de ellos se derivarían otros grafos más pequeños para ejemplificar situaciones. Tomando como base la Figura 12, podría decirse que el grafo “Padre” representa la conexión entre diversas computadoras de un laboratorio, y el subgrafo consecuente únicamente representa la unión entre las computadoras que están compartiendo algún archivo, a través de la red, en ese momento.

1.5 Ejemplo de la aplicación de los grafos en la Informática

Para representar de mejor manera cómo se pueden ocupar los grafos en la informática, se propone el siguiente ejemplo:

23 Veites Rodríguez, A. (2014). Teoría de grafos. España: Paraninfo.

El nuevo encargado de un laboratorio de computadoras está estudiando la relación existente en la red de máquinas compuesta por 5 equipos. Se sabe que las computadoras envían información cada cierto tiempo pero él desea saber qué equipos son los que interactúan entre sí y cuales no intercambian ningún tipo de dato, además, así podrá saber qué equipos tienen mayor carga de trabajo al tener más conexiones con otras computadoras pues serán a los que les dará prioridad en su mantenimiento. Mediante un grafo se consigue establecer qué maquina se conecta con cual, por ejemplo, $1 \rightarrow 2$ significa que la “computadora 1” envía información a la “computadora 2”.

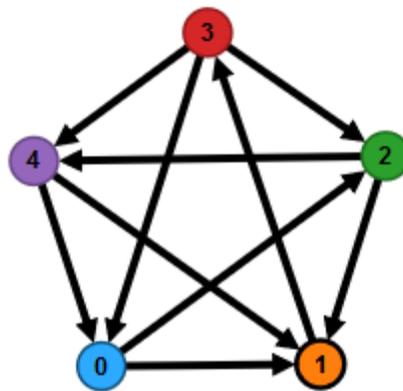


Figura 13. Grafo ejemplificado con el uso de la aplicación web.

El grafo anterior tendría una matriz de adyacencia de la siguiente manera:

	0	1	2	3	4
0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
2	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
3	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
4	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figura 14. Matriz de adyacencia ejemplificada con el uso de la aplicación web.

Para determinar cuál es la computadora que más conexiones tiene en la red, se suman las filas de la matriz de adyacencia, con esto se puede conocer a cuántas máquinas envía información a cada equipo. En este caso obtenemos:

- “Computadora 0” envía a 3 máquinas.
- “Computadora 1” envía a 1 máquina.
- “Computadora 2” envía a 2 máquinas.
- “Computadora 3” envía a 3 máquinas.
- “Computadora 4” envía a 2 máquinas.

Tanto la “Computadora 0” como la “Computadora 3” envían datos a 3 máquinas, es decir, son los equipos que mayor información comparten en la red y por consiguiente tienen una mayor carga de trabajo, son las computadoras a las que el encargado de laboratorio les brindará mayor cuidado pues una falla en ellas generaría problemas importantes en la red.

CAPÍTULO 2 “Software Libre”

Una buena práctica para el programador (y en general debería de serlo para cualquier usuario) es la utilización de software libre. Para la elaboración de la aplicación web interactiva, se utilizaron herramientas de software libre por las ventajas y bondades que ofrecen.

Durante la realización del proyecto se ocupó:

- Sistema operativo GNU/Linux: Debian²⁴ para el servidor que aloja la aplicación y Linux Mint²⁵ para el equipo de cómputo con el que se programó.
- Servidor HTTP Apache²⁶: Servidor web de código abierto que está diseñado para transferir datos de hipertexto, como la aplicación web desarrollada, con todos sus elementos (textos, imágenes, etc.) a cualquier navegador que realice alguna petición.
- D3²⁷: Librería de Javascript que permite representar gráficos interactivos en navegadores web. Otorga una licencia que faculta su uso tanto en programas de código abierto como en los que no lo son, es decir, no obliga al programador a compartir su código.
- Bootstrap²⁸: Se trata de un framework web, es decir, un conjunto de herramientas de código abierto para diseño de sitios y aplicaciones web.
- Navegador web Mozilla Firefox²⁹: Navegador web libre y de código abierto desarrollado para Linux y otros sistemas operativos. Propiedad de la Corporación Mozilla[®] y la Fundación Mozilla[®].

Además, la aplicación web interactiva propuesta en esta tesis se publica bajo la “*Licencia Creative Commons Atribución-NoComercial-CompartirIgual 4.0 Internacional*”, que autoriza compartir y

24 Debian. (2017). “Acerca de Debian”. Disponible en: <https://www.debian.org/intro/about>

25 Linux Mint. (2017). “About Us”. Disponible en: <https://linuxmint.com/about.php>

26 The Apache Software Foundation. (2017). “What is the ASF?”. Disponible en: <https://www.apache.org/foundation/>

27 D3 Data-Driven Documents. (2017). “D3 Data-Driven Documents Overview”. Disponible en: <https://d3js.org/>

28 Bootstrap. (2017). “Bootstrap Documentation”. Disponible en: <https://getbootstrap.com/docs/4.0/getting-started/introduction/>

29 Mozilla. (2017). “Conoce Mozilla”. Disponible en: <https://www.mozilla.org/es-MX/about/>

redistribuir el material además de adaptarlo y/o transformarlo según las necesidades; no obstante, lo anterior se permite siempre y cuando se otorguen los créditos al autor original, se utilice sin fines comerciales y no se añadan nuevos términos legales que interfieran con la licencia originalmente establecida.

2.1 Conceptos Generales

El Software Libre³⁰ trabaja bajo el principio de respetar la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software. Un programa es software libre si los usuarios tienen todas estas libertades, por tanto, como usuario se debe tener la libertad de distribuir copias, sea con o sin modificaciones, sea gratis o cobrando una cantidad por la distribución, a cualquiera y a cualquier lugar. El ser libre de hacer esto significa (entre otras cosas) que no se tiene que pedir o pagar permisos.

Con lo anterior, se busca que las personas controlen el programa y lo que este hace, son libertades que sus defensores argumentan todos tenemos derecho, pues un programa que no es libre (conocido como privativo) controla al usuario y lo limita; además, el desarrollador del programa pasa a tener control sobre los consumidores y puede verse tentado a realizar actividades poco éticas sacando provecho del poder que posee.

Cuando se habla de software libre es preferible evitar usar palabras como “gratis” o “regalado”, la razón es porque esos términos implican que lo importante es el precio y no la libertad que se le otorga al usuario, que es lo principal que persigue este movimiento. Un programa puede ser gratuito pero no libre, pues aunque no represente un costo económico adquirirlo, sí que les impide a las personas redistribuirlo o acceder al código fuente para adaptarlo a sus necesidades.

El fundador del movimiento por el software libre es Richard Stallman³¹, iniciándolo en 1983. Es creador del proyecto “GNU” y presidente de la “Free Software Foundation”, también trabajó en el concepto de

30 Arteaga, L. (2001) “¿Qué es el Software Libre”. Consultado el 16 de marzo de 2017. Disponible en: <https://www.gnu.org/philosophy/free-sw.es.html>

31 Richard Stallman (1953) es un programador estadounidense y figura del movimiento por el software libre. Participó en logros notables de la informática pero es más conocido por su referencia a la iniciativa del copyleft, el proyecto GNU y el software libre.

“copyleft” y la licencia “GNU GPL”. La ideología política y moral de Stallman lo ha convertido en una figura controvertida a la vez de importante en el mundo de la informática.

Muchos programadores están de acuerdo con el concepto de compartir el código, pero difieren con las posturas morales, filosofía, personalidad y el lenguaje que utiliza Stallman para describir sus posiciones, por ello, también surgió una alternativa al movimiento del Software Libre: El Movimiento del Código Abierto³².

GNU³³ proviene del acrónimo recursivo “GNU's Not Unix” (GNU no es Unix). Unix³⁴ era un sistema operativo muy popular en los años 80, así que Stallman diseñó GNU para ser muy compatible con Unix logrando que la migración a GNU fuera fácil. GNU aprendió del diseño técnico de Unix, pero también indica claramente en su propio nombre que no están relacionados. A diferencia de Unix, GNU es Software Libre.

GPL³⁵ es acrónimo de “General Public License” (Licencia Pública General, de GNU). Esta licencia está pensada para proteger las libertades del usuario final del software. Un programa sujeto a dicha licencia tiene que ofrecer el código para que el usuario pueda modificarlo si le es necesario, y en dicho caso lo único que se le pide es que redistribuya los cambios también con la licencia GPL. Además, el usuario puede utilizar el programa de la manera que le apetezca sin tener que pedir permiso al autor del software, también tiene el derecho a redistribuirlo a quien desee.

Destacar que, en 1992, el programador Linus Torvalds³⁶ utilizó las licencias GPL del sistema GNU para su más importante creación: el kernel Linux³⁷. A partir de entonces ambos proyectos empezaron a

32 Garzón, E. (2012). “Biografía de Richard Stallman”. Consultado el 22 de marzo de 2017. Disponible en: http://e-logicasoftware.com/tutoriales/curso-virtual-software-libre-alcaldia/1_introduccion/recursos/richardstallman.pdf

33 Free Software Foundation. (2016). “El sistema operativo GNU”. Consultado el 22 de marzo de 2017. Disponible en: <https://www.gnu.org/gnu/gnu.es.html>

34 Fundación del Software Libre en Europa (2017). “¿Qué es el proyecto GNU?”. Consultado el 22 de marzo de 2017. Disponible en: <https://fsfe.org/freesoftware/basics/gnuproject.es.html>

35 Free Software Foundation. (2016) “Licencias”. Consultado el 23 de marzo de 2017. Disponible en: <https://www.gnu.org/licenses/licenses.es.html#GPL>

36 Linus Torvalds (1969) es un programador de origen finlandés que en su juventud creó una propia versión del sistema UNIX por no poder pagar los costos de las licencias, tiempo después ese sistema se convertiría en el conocido “Linux”.

37 Yúbal, F. (2016). “Entrevista a Richard Stallman”. Consultado el 23 de marzo de 2017. Disponible en: <https://www.genbeta.com/especial/el-usuario-merece-que-sea-obligatorio-que-se-respete-su-libertad-entrevista-a-richard-stallman>

caminar de la mano creando así el sistema operativo “GNU/Linux”. En ocasiones, a dicho sistema se le llama simplemente “Linux”, lo cual es un error ya que Linux es sólo el núcleo del sistema; en un principio GNU tenía su propio kernel llamado “Hurd”, pero se acabó apostando por el de Torvalds creando así un sólo sistema (GNU/Linux).

Free Software Foundation³⁸ (Fundación del Software Libre, también referida simplemente como FSF). Es una organización creada por Richard Stallman, en 1985, junto con otros programadores defensores del software libre. En su propia página web, se describe como “*una organización sin fines de lucro con una misión mundial para promover la libertad de los usuarios de computadoras*”. La FSF está dedicada a eliminar las restricciones sobre la copia, redistribución y modificación del software; promueven el desarrollo y uso del software libre en todas las áreas de la computación, pero muy particularmente, ayudando a desarrollar el sistema operativo GNU, esta fundación se apoya de donaciones que recibe de personas, principalmente en Estados Unidos.

Copyleft³⁹ es una regla que indica que cuando se redistribuya un programa, no se pueden agregar restricciones para negar a otras personas su acceso o libertades. Esta regla protege la libertad del usuario de manipular y estudiar el programa. Su principal objetivo es permitir libremente a todos compartir software GNU.

2.2 Libertades

Referente a la libertad, se debe tener en cuenta que la palabra “libre” dentro de “Software Libre” no necesariamente hace referencia a que el programa no tenga un precio o sea libre de pago; más bien indica que el usuario tiene libertad sobre el uso del programa. Ahora bien, para responder sobre qué tipo de libertad es la que se obtiene, sencillamente se puede hacer mención a los siguientes puntos que todo

38 Free Software Foundation. (2017). “Free Software Foundation – Front Page”. Consultado el 23 de marzo de 2017. Disponible en: <http://www.fsf.org/>

39 Free Software Foundation. (2016). “¿Qué es el copyleft?”. Consultado el 23 de marzo de 2017. Disponible en: <https://www.gnu.org/licenses/copyleft.es.html>

software debe cumplir para considerarse como libre, también conocidas como las “4 libertades esenciales”⁴⁰:

0. Se cuenta con la libertad de ejecutar el programa cualquiera que sea el propósito.
1. Hay libertad para modificar el software según las necesidades del usuario. Es aquí un punto determinante pues obligatoriamente se debe contar con acceso al código fuente, caso contrario sería prácticamente imposible modificar el programa para adaptarlo.
2. Se tiene libertad para distribuir copias, sean gratuitas o de pago.
3. Libertad de distribución de copias del programa modificado para que otros usuarios tengan acceso a él y puedan aprovechar las bondades del software.

Cabe retomar el hecho de que aunque la palabra “libre” no impida la venta de copias del programa a otros usuarios, esto no significa una contradicción en cuánto a la libertad propuesta, de hecho resulta conveniente mencionar que las ventas de un disco que contenga diversos programas de software libre o los pagos/donaciones dentro de las páginas web de los autores, son una buena forma de recaudación de fondos que les permite continuar con su trabajo.

Para que estas libertades sean reales deben ser irrevocables mientras el usuario no haga nada incorrecto. Si el desarrollador del software tiene el poder de revocar la licencia aunque no se le hayan dado motivos, el software no es libre.

40 Arteaga, L. (2001). “¿Qué es el software libre?”. Consultado el 23 de marzo de 2017. Disponible en: <https://www.gnu.org/philosophy/free-sw.es.html>

2.3 Tecnologías que utilizan el software libre

El software libre ha tenido presencia a lo largo de la historia de la informática, principalmente por mostrarse como una alternativa confiable respecto a los programas privativos que existen en el mercado. Propiamente se puede hacer mención de algunas ventajas que tiene el software libre frente al de tipo privado:

- Fomenta la colaboración entre las personas.
- No hay costo por licencias o actualizaciones.
- No promueve la piratería, las licencias son de distribución libre.
- Se encuentra fácilmente soporte técnico gracias a las comunidades en Internet de software libre.
- Existen para diversas plataformas (Windows®, MacOS®, Linux®).
- Sin problema de virus: Principalmente en el caso de Windows®, hay gran cantidad de virus que atacan al software propio de Microsoft®; estos virus no van dirigidos hacia las herramientas de software libre.
- Las actualizaciones del programa generalmente son constantes y accesibles desde internet.
- Son adaptables y configurables a las necesidades del usuario.

Dicho esto, inevitablemente han surgido diversos proyectos y actividades que se han apoyado del software libre para alcanzar sus metas, por ejemplo:

- Desarrollo de plataformas informáticas enfocadas al apoyo de actividades y estudio dentro de universidades o grupos de investigación de cualquier tema. Se trata de comunidades virtuales que comparten información relevante para ellos.
- Sistemas de ofimática útiles para la creación de documentos, presentaciones, hojas de cálculo y demás actividades que se necesiten realizar como mapas mentales, diagramas, manipulación de imágenes, etc.
- Herramientas de comunicación tales como correo electrónico, mensajería instantánea, salas de conversaciones, foros de discusiones, entre otros.
- Buscadores que permitan el acceso eficiente de recursos en internet.

- Herramientas multimedia para la publicación de trabajos individuales o colaborativos, compendios y fuentes de información. Por ejemplo weblogs, sistemas wiki⁴¹, podcast, etc.

Es notoria la presencia del software libre en diversos tipos de programas que utilizamos cotidianamente, resulta conveniente entender que se cuenta con muy buenas alternativas al software privativo y que en muchos casos son mejores por las ventajas antes mencionadas.

41 “Wiki” es el nombre que el programador Ward Cunningham, escogió para su invento, en 1994: un sistema de creación, intercambio y revisión de información en la web, de forma fácil y automática. La finalidad de un Wiki es permitir que varios usuarios puedan crear páginas web sobre un mismo tema, de esta forma cada persona aporta un poco de su conocimiento para que la página web sea más completa.

CAPÍTULO 3 “Tecnologías Web”

Para entender correctamente las tecnologías web, primero hay que remitirse al concepto propio de tecnología⁴², y citando a Leandro Alegsa²⁷, ésta es “*un conjunto de técnicas, conocimientos y procesos, que sirven al diseño y construcción de objetos para satisfacer necesidades humanas; la palabra tecnología proviene del griego tekne (técnica, oficio) y logos (ciencia, conocimiento)*”.

El concepto de tecnología permite hacer mención tanto a objetos físicos (por ejemplo máquinas, utensilios, computadoras y hardware en general) y también propiamente a elementos intangibles como el software (sistemas informáticos, técnicas, etc.). La informática se enfoca mayormente en las computadoras, las cuales por medio de aplicaciones, permiten realizar tareas que antes se hacían de forma manual y ahora se realizan de manera más rápida y eficaz; se consigue así una automatización de procesos.

Una de las tecnologías que abarcan la informática es la Web⁴³, que es sinónimo de “World Wide Web” y cuya definición es “*la idea básica de fusionar las técnicas de conexión de computadoras mediante redes y el hipertexto en un sistema de información global potente y fácil de utilizar*”. Gracias a ella millones de personas tienen acceso inmediato a una cantidad extensa y diversa de información. Ciertamente, en la actualidad existe una gran preferencia hacia los entornos Web por ser cómodos, atractivos y ágiles.

Por su parte, un navegador⁴⁴ web (también conocido como explorador o “browser” en inglés) es la aplicación que permite acceder a la web y “navegar” en ella (de ahí el nombre); esencialmente es el programa que permite visualizar los archivos que hay disponibles y trabajar con ellos.

42 Alegsa, L. (2016). “Definición de Tecnología”. Consultado el 24 de marzo de 2017. Disponible en: <http://www.alegsa.com.ar/Dic/tecnologia.php>

43 Gago, F. (). “¿Qué es la World Wide Web?”. Consultado el 24 de marzo de 2017. Disponible en: http://www3.uah.es/farmamol/Public/Curso_Internet/CERN/quees.html

44 Alegsa, L. (2016). “Definición de Navegador”. Consultado el 24 de marzo de 2017. Disponible en: <http://www.alegsa.com.ar/Dic/navegador.php>

3.1 Estándares

3.1.1 HTML

Al hablar de las tecnologías web, no se puede dejar de lado a HTML⁴⁵, cuyo significado es el de un “Lenguaje de Marcado para Hipertextos” (HyperText Markup Language). Se trata del elemento de construcción básico del desarrollo web y se usa para crear una página web, determina el contenido pero no su funcionalidad (para ello existen los lenguajes propios de programación). Básicamente se trata de un conjunto de etiquetas que sirven para definir el texto y otros elementos que compondrán una página como imágenes, listas, vídeos, etc.

HTML sólo sirve para indicar como va ordenado el contenido de una página web, por tanto necesita apoyarse de otros lenguajes (PHP y Javascript, usados mayormente) para agregar las funciones que se deseen desempeñar en la plataforma web. Hiper Texto⁴⁶ se refiere a los enlaces que conectan una página con otra, ya sea dentro del mismo sitio o uno diferente.

En cuanto a las etiquetas⁴⁷, son la manera en que se estructura el documento que dará forma al archivo .html (o .htm, también es reconocido así). Estas etiquetas, al igual que muchos lenguajes, necesitan que se les indique donde inician y donde terminan, por ejemplo si se abre una etiqueta <body> (que es utilizada para escribir el cuerpo de la página web), necesariamente requerirá de una etiqueta de cierre </body> para que el navegador (que en este caso es el intérprete) sepa dónde acaba esa sección del documento. Existen varias etiquetas disponibles, por ejemplo para indicar un párrafo está <p>, para las negritas en el texto es , para interlineado
, para las imágenes , para los enlaces <a> y muchas más.

Por otro lado, en cuanto a las ventajas que posee HTML se encuentran las siguientes:

45 Flores, J. (2016). “¿Qué es HTML”. Recuperado el 27 de marzo de 2017. Disponible en: <https://codigofacilito.com/articulos/que-es-html>

46 Mozilla Foundation (2017). “HTML”. Recuperado el 27 de marzo de 2017. Disponible en: <https://developer.mozilla.org/es/docs/Web/HTML>

47 Álvarez, M. (2001). “Qué es HTML”. Recuperado el 27 de marzo de 2017. Disponible en: <https://desarrolloweb.com/articulos/que-es-html.html>

- Sencillo de utilizar.
- Se considera de fácil aprendizaje, quien desee conocerlo no necesita conocimientos profundos en programación.
- El texto está presentado de forma estructurada, lo que le da fácil interpretación.
- Lo admiten todos los navegadores web.
- Usualmente, los archivos .html son pequeños, no ocupan mucho espacio en memoria.

Sin embargo, también se deben tener en cuenta las desventajas, mismas que principalmente son:

- La interpretación que le dé cada navegador puede ser diferente.
- Las etiquetas son algo limitadas (afortunadamente la aparición de HTML5⁴⁸ ha solucionado en parte esta situación).

3.1.2 CSS

Por otro lado, está CSS⁴⁹, que es acrónimo de Cascading Style Sheets (hojas de estilo en cascada, en español), es el encargado de describir cómo será la presentación de los archivos .html, es decir, el aspecto del contenido en una página web (colores, tipo de texto, etc.).

CSS es un estándar de la W3C⁵⁰ que es ampliamente aceptado entre los navegadores web modernos. Gracias a él, un programador puede definir varios detalles de una página web como las fuentes, colores, diseño, posicionamiento, imágenes, tamaño de página, márgenes, etc., manteniendo así un estándar entre sus páginas y otorgando una fácil actualización y adaptación del diseño pues permite hacer modificaciones al aspecto rápidamente sin tocar el código HTML.

48 Pimentel, V. (2009). “Las novedades de HTML5”. Recuperado el 27 de marzo de 2017. Disponible en:

<http://www.nobbot.com/personas/las-novedades-de-html5-i/>

49 Mozilla Foundation (2017). “Tecnología web para desarrolladores”. Recuperado el 27 de marzo de 2017. Disponible en:

<https://developer.mozilla.org/es/docs/Web/>

50 W3C es el acrónimo de World Wide Web Consortium (Consortio de la Web o Telaraña Mundial), es un organismo que establece los estándares de la WWW. Está dirigida por Tim Berners-Lee (creador de la WWW).

Las reglas del diseño que se obtienen con el uso de CSS pueden aplicarse a uno o más tipos de etiquetas de HTML, pueden ejecutarse tanto a todas las de un sólo tipo (por ejemplo, a todas las etiquetas <p> de un archivo .html), o sólo a algunas en específico (en un caso podría ser que sólo las etiquetas <h1> tendrán un aspecto definido mientras que las demás <h2>, <h3>, <h4>, <h5> y <h6> continuaran con el estilo predefinido por el navegador).

Esas reglas de CSS respetan una jerarquía "en cascada", esto quiere decir que las reglas más generales se definen y heredan a las demás, pero pueden ser reemplazadas por otras reglas más específicas. Por ejemplo, si se aplica un estilo a todas las etiquetas <p> (párrafos) y luego se aplica otro estilo a una clase especial de párrafos, estos párrafos "especiales" heredarán el estilo general y el estilo especial; eso sí, mencionar que el especial tiene prioridad y reemplaza lo que se le indique del general.

Las ventajas de utilizar CSS son las siguientes:

- Los sitios que utilizan CSS suelen cargar más rápido, esta velocidad se obtiene porque cuando se abren varias páginas que comparten la misma hoja de estilo, el navegador ya cargó en el caché los elementos y no necesita volver a descargarlos para mostrarlos.
- Reduce el tiempo de mantenimiento pues al querer hacer una modificación (por ejemplo al tipo de fuente del texto) no se tiene que modificar uno por uno los archivos “.html”, bastará con actualizar el archivo “.css”.
- Incrementa las opciones de personalización y diseño en un sitio web.

Respecto a las desventajas:

- Puede darse el caso de que dos navegadores interpreten el mismo código CSS de diferente manera (situación que eventualmente se evitaría con un código bien estructurado y ordenado).
- Si hay problemas o límites de compatibilidad, el navegador aplicará el formato predeterminado; esto provocaría que el trabajo de asignar estilos y formatos termine siendo inútil.
- Si no se ocupan adecuadamente, algunas propiedades de las hojas de estilo pueden provocar que haya contenido de la página web que resulte inaccesible desde algunos navegadores.

3.2 Lenguajes de programación web

Evidentemente, no se podría entender la tecnología web sin sus propios lenguajes de programación, mismos que permiten dar indicaciones precisas a las computadoras para realizar las actividades que se les designen.

Dichos lenguajes fueron surgiendo para responder a las tendencias y necesidades de los desarrolladores y las plataformas web emergentes, entre ellos destacan PHP, Ruby, Python, etc; pero en esta ocasión se hablará del que se ocupó durante la elaboración de la aplicación web propuesta: Javascript.

3.2.1 JAVASCRIPT

JavaScript⁵¹ es un lenguaje de programación interpretado (no requiere de compilación) y orientado a objetos. Para dar resultados, este lenguaje necesita trabajar en conjunto con HTML y así crear sitios web interactivos y aplicaciones avanzadas. El estándar de JavaScript es ECMAScript⁵² (European Computer Manufacturers Association), éste es el que permite que los navegadores puedan soportarlo y desde el 2012 todos los exploradores modernos son compatibles con él.

JavaScript fue creado por Brendan Eich⁵³ en la empresa Netscape Communications, no debe ser confundido con el lenguaje de programación Java®, pues este último es una marca registrada de Oracle® en Estados Unidos; realmente Javascript se llamó así⁵⁴ porque Netscape (que estaba aliado con los creadores de Java en esa época), quiso aprovechar la popularidad de Java y creó un producto que tenía ciertas similitudes en la sintaxis del lenguaje y el nombre, pero haciendo énfasis en que su creación estaba

51 Mozilla Foundation (2017). “Tecnología web para desarrolladores”. Recuperado el 27 de marzo de 2017. Disponible en: <https://developer.mozilla.org/es/docs/Web/>

52 Mozilla Foundation (2017). “Javascript”. Recuperado el 27 de marzo de 2017. Disponible en: <https://developer.mozilla.org/es/docs/Web/Javascript>

53 Brendan Eich (1961) es un programador estadounidense de renombre pues fue quien creó el lenguaje JavaScript, también colaboró con el fundador de la Fundación Mozilla sirviendo como principal arquitecto.

54 Álvarez, M. (2009). “Diferencias entre Java y JavaScript”. Recuperado el 27 de marzo de 2017. Disponible en: <https://www.desarrolloweb.com/articulos/492.php>

orientada para páginas web, lo cierto es que también se provocó que muchas personas cayeran en el error de pensar que son lo mismo.

Las primeras páginas web eran estáticas, la única forma de ofrecer contenido dinámico era mediante una sucesión de páginas que fuera abriendo secuencialmente el usuario, además cada cambio significativo requería que se refrescara la página web entera. Sin embargo, cuando el navegador Netscape (1995) introdujo Javascript, permitió a los programadores agregar algunos elementos interactivos a la interfaz de usuario (del lado del cliente); fue entonces que en lugar de depender totalmente del servidor para generar una web completa, los scripts podían realizar ciertas tareas sin todo ese apego a su servidor, por ejemplo la validación de entradas, mostrar u ocultar secciones, etc.

Tiempo después, para el año de 2005, apareció el término AJAX⁵⁵ (Asynchronous Javascript and XML) que es una técnica que permite, mediante programas escritos en Javascript, que un servidor y un navegador intercambien información de forma asíncrona (significa que el emisor emite un mensaje y continúa con su trabajo, dado que no sabe -ni necesita saber- cuándo le llegará el mensaje al receptor.). Fue así que nacieron aplicaciones web sumamente interactivas (como Gmail® de Google®). AJAX permite que un script se contacte con el servidor para almacenar o recibir datos sin necesidad de descargar o actualizar la página web completa.

Dentro de las principales ventajas de Javascript, se encuentran:

- El código se ejecuta en el cliente, aligerando la carga de trabajo del servidor.
- Otorga dinamismo a la página web y funcionalidades interesantes.
- Compatibilidad con todos los navegadores modernos.

Por el lado de las desventajas están:

- Código visible por cualquier usuario.
- El código se debe descargar completamente para su ejecución.

55 Digital Learning. (2012). “¿Qué es AJAX?” Recuperado el 27 de marzo de 2017. Disponible en: <http://www.digitalllearning.es/blog/que-es-ajax/>

- Si bien es seguro, también hay vulnerabilidades conocidas como el problema llamado XSS⁵⁶ (Cross-Site Scripting,) donde hay una inyección de código por algún atacante informático con la intención de redirigir al usuario a otro sitio web con contenido que ponga en riesgo su seguridad.

3.3 Servidor

Un Servidor⁵⁷ es una computadora que está al servicio de otros equipos de cómputo para resolver sus peticiones mediante el uso programas informáticos. Precisamente, el nombre “servidor” se deriva del hecho de estar “sirviendo” a otras computadoras. Las máquinas que realizan las peticiones al servidor se denominan “clientes”.

Existen diversos tipos de servidores, por ejemplo están los de correo electrónico, que a su vez se subdividen en servidores POP3 (envío), IMAP (manejo de correos) y SMTP (recibir). Pero en este caso será oportuno enfocarse en los dos tipos de servidores utilizados para la aplicación web desarrollada: el servidor FTP y el servidor Web.

3.3.1 Servidor FTP

FTP⁵⁸ viene del acrónimo File Transfer Protocol (Protocolo de Transferencia de Archivos, en español), y se trata de un servidor conectado a una red (normalmente internet) que permite el intercambio de archivos entre diferentes computadoras (cliente/servidor). Los FTP y su software de gestión garantizan la seguridad de los archivos y control de las transferencias, esto los hace de gran utilidad y muy ocupados en el mundo.

56 Aguilar, A. (2015). “¿Qué es y cómo opera un ataque de Cross-Site Scripting”. Recuperado el 27 de marzo de 2017. Disponible en: <http://www.seguridad.unam.mx/documento/?id=35>

57 Área Tecnología (2017). “Qué es un servidor y tipo de servidores”. Recuperado el 28 de marzo de 2017. Disponible en: <http://www.areatecnologia.com/informatica/servidor-y-tipos.html>

58 Autor Desconocido (2016). “¿Qué es el servicio FTP – File Transfer Protocol?”. Recuperado el 28 de marzo de 2017. Disponible en: <http://www.internetya.co/que-es-el-servicio-ftp-file-transfer-protocol/>

Las aplicaciones más comunes que tienen los servidores FTP suelen ser el “hosting” (relacionado con el tema de Servidores Web que se tocará a profundidad más adelante), en el que los clientes lo utilizan para subir las páginas web y sus archivos relacionados; otra utilidad de este tipo de servidores es para hacer copias de seguridad (“backup”) de los archivos importantes que pueda tener una persona o empresa. Existen protocolos de comunicación usados por el servidor FTP para que los datos se transmitan cifrados, por ejemplo está el SFTP (Secure File Transfer Protocol).

Un buen ejemplo del software FTP ocupado en estos servidores es FileZilla®, utilizado durante el desarrollo de la aplicación de grafos y que permite conectarse al servidor web para enviar los documentos (“.html”, “.css”, “.js”, imágenes, etc.) necesarios para el correcto funcionamiento de la aplicación y así alojarlos en él.

3.3.2 Servidor Web

Los archivos propios de un sitio web se almacenan y se ejecutan en este tipo de servidor. El servidor envía el archivo “.html” al navegador del cliente para que lo pueda visualizar, toda esta comunicación sigue una norma llamada “protocolo HTTP”⁵⁹.

Estos servidores cuentan con un espacio para el alojamiento que se llama “Hosting”. El hosting⁶⁰ es una analogía del hospedaje o alojamiento en hoteles, esto porque se ocupa un lugar virtual específico designado por el servidor. Por lo general, los servidores tienen la capacidad de soportar un gran número de páginas web almacenadas.

Hay dos tipos principales de hosting:

59 “HyperText Transfer Protocol”, es el método de transferencia de páginas web desde un servidor a un cliente (a través del navegador).

60 Aprigiliano, D. (2016). “¿Qué es un hosting o servidor web? ¿para qué sirve?”. Recuperado el 29 de marzo de 2017. Disponible en: <http://www.dosd.com.ar/que-es-un-hosting-o-servidor-web-para-que-sirve/>

- Hosting compartido: en el servidor web hay varias páginas alojadas de distintas personas.
- Hosting dedicado: es un servidor para una sola persona o empresa, lógicamente son más caros pues el mantenimiento corre a cargo de ese individuo.

3.3.3 Modelo Cliente - Servidor

El modelo o arquitectura que siguen los servidores es el de “cliente-servidor”⁶¹, es decir, el cliente pide información o datos y el servidor proporciona los recursos o servicios.

Los usuarios utilizan una computadora que se convertirá en el cliente, y a través de una aplicación se construye una solicitud que se envía al servidor. Por su parte, el servidor utiliza software que recibe dicha petición, realiza el servicio requerido y devuelve los resultados en forma de respuesta. Generalmente un servidor puede tratar múltiples peticiones (de clientes diferentes) al mismo tiempo.

61 Área Tecnología (2017). “Qué es un servidor y tipo de servidores”. Recuperado el 29 de marzo de 2017. Disponible en: <http://www.areatecnologia.com/informatica/servidor-y-tipos.html>

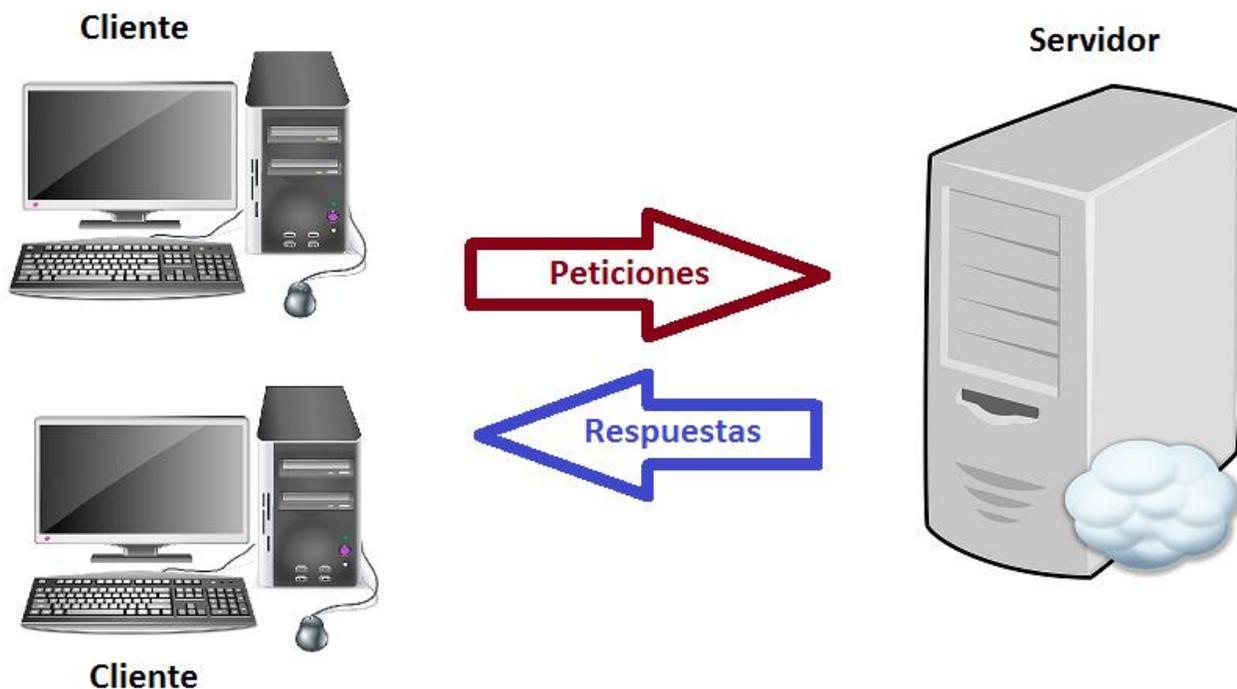


Figura 15. Representación de computadoras clientes y un servidor⁶².

3.4 Aplicaciones Web

Una aplicación web⁶³ es un programa informático al que se accede desde un navegador web, es decir, se le confía la ejecución de la aplicación a dicho explorador, en otras palabras, se trata de una aplicación como otra cualquiera, con la ventaja de que está almacenada en la red y que puede ser ejecutada por cualquier navegador. Este tipo de software se encuentra alojado en un servidor y se ingresa a ellos, generalmente, desde internet (aunque también podría ser desde una intranet).

Una de las ventajas de las aplicaciones web es la facilidad de mantener y actualizar los programas sin la necesidad de distribuir e instalar software en las computadoras de los usuarios, también tienen la

62 Imagen generada con recursos de tipo “CC0 Public Domain” de: OpenClipart-Vectors (2013). “Vectores gratis”. Recuperado el 29 de marzo de 2017. Disponible en: <https://pixabay.com/es/nube-servidor-equipo-alojamiento-2023901/> y en <https://pixabay.com/es/equipo-escritorio-158675/>

63 Alegs, L. (2016). “Definición de Aplicación Web”. Consultado el 29 de marzo de 2017. Disponible en: http://www.alegsa.com.ar/Dic/aplicacion_web.php

posibilidad de ser ejecutados en múltiples sistemas operativos por la gran portabilidad que otorgan los navegadores web.

Usualmente tienen funcionalidades similares a las de un programa de escritorio (o una aplicación móvil), prácticamente no hay limitaciones, las aplicaciones web pueden hacer a casi todo lo que está disponible para aplicaciones tradicionales: acceder al mouse y al teclado, ejecutar audio o video, mostrar animaciones, etc.

Entre las principales características de las aplicaciones web se encuentra la posibilidad de acceder a ellas desde cualquier parte del mundo siempre y cuando se cuente con una conexión a internet, en otros términos, no se está apegado a usar un sólo equipo de cómputo que tenga instalado el software como sucede con los programas que funcionan localmente. Otra cualidad interesante es que pueden existir miles de usuarios para una única aplicación web instalada en un servidor, lo que permite que sólo se deba actualizar y mantener un solo programa y todos sus usuarios se verán beneficiados por los resultados.

Las aplicaciones web se desarrollan en diversos lenguajes de programación, cada uno ofreciendo distintas posibilidades donde el programador es el que elige cuál es el que adapta mejor a sus necesidades, se puede mencionar Javascript, PHP, Ruby on Rails y Python por ser los más utilizados, aunque no se debe olvidar la necesaria utilización de HTML y CSS para la interfaz del usuario. Respecto a la aplicación web interactiva presentada en este trabajo, se ocupó Javascript, HTML y CSS.

Las ventajas que ofrecen las aplicaciones web son:

- Generalmente, requieren poco espacio de almacenamiento en disco.
- No se necesita que el usuario las actualice constantemente, este proceso se implementa del lado del servidor.
- Hay gran compatibilidad y portabilidad entre plataformas, esto se debe a que los navegadores web están presentes en la mayoría de los sistemas operativos actuales.

Y en cuanto a las desventajas se puede mencionar:

- Las aplicaciones web requieren navegadores totalmente compatibles, es decir, se necesita que el explorador cumpla los requisitos previos para su correcto funcionamiento.
- Algunas aplicaciones necesitan que se instale, en el equipo del usuario, complementos o extensiones para su ejecución, sin ellas la aplicación presentará problemas.
- Muchas veces se necesita de conexión continua a internet, por lo tanto si ésta se interrumpe, la aplicación dejará de funcionar.
- Varias de estas aplicaciones no son de código abierto, perdiendo flexibilidad.
- La aplicación web desaparece si así lo decide el desarrollador (al eliminarla del servidor). Con el software tradicional que está instalado en el equipo del usuario, esto no sucede pues aunque se termine el soporte al programa, las personas pueden seguir ocupándolo localmente.
- En general, el usuario no tiene la libertad de elegir qué versión del software es la que va ocupar, utiliza únicamente la que esté disponible en el servidor (sea la más reciente o una versión antigua y obsoleta).
- Técnicamente, el desarrollador web puede rastrear toda actividad de los usuarios, esto implica problemas sensibles de privacidad.

3.4.1 Ejemplos de aplicaciones web

En la actualidad, existen varios tipos de aplicaciones web presentes en todas las actividades informáticas que las personas realizan⁶⁴, principalmente se pueden mencionar:

- Portales web:
Se trata de un tipo de aplicación en el que la página principal permite el acceso a diversos apartados, categorías o secciones. Hay de varios tipos, por ejemplo foros, chats, buscadores, elaboración de registros, etc. Entre las principales ventajas se encuentran la visibilidad 24/7, una

64 Mocholí, A. (2015). “6 Tipos de desarrollo de aplicaciones web más relevantes”. Recuperado el 30 de marzo de 2017. Disponible en: <https://www.yeeply.com/blog/6-tipos-desarrollo-de-aplicaciones-web/>

mayor exposición que permite llegar a un público numeroso (básicamente, todo aquel que tenga internet), reducción de costos en cuanto a difusión de información (por ejemplo, para dar a conocer alguna noticia no es necesario imprimir una gran cantidad de folletos y/o ilustraciones, basta con incluirlo en la página web), entre otros.

- **Tienda virtual (comercio electrónico):**

Es una aplicación web que tiene el enfoque de ser una tienda y ofrecer comercio digital, también se les conoce como “e-commerce”. Este tipo de software en la web es más elaborado y debe enfocarse también en la seguridad de los consumidores pues debe permitir pagos electrónicos a través de tarjeta de crédito, PayPal®, u otro medio. Entre los principales ejemplos de tiendas línea se puede mencionar a Amazon®, Mercado Libre®, Ebay®, entre otros. Sus ventajas más llamativas están la exposición mundial de los productos vendidos, fácil modificación de la información (tan solo se actualiza la página web con nuevos productos o eliminando otros) y un mejor control de la satisfacción del cliente (creando encuestas, recibiendo comentarios, etc).

- **Gestores de contenidos:**

Cuando se trabaja bajo la premisa de ir actualizando continuamente noticias e información, se deberá utilizar un tipo de aplicación web conocida como “Gestor de contenidos” (o CMS, por las siglas en inglés de “Content Management System”) a través del cual se pueden ir realizando los cambios y actualizaciones de información. Estos gestores son muy intuitivos y tienen la gran ventaja de que no es necesario que un experto en desarrollo web sea el encargado de agregar el nuevo contenido. Se trata de una aplicación muy común para crear blogs personales, blogs corporativos, blogs profesionales, páginas de noticias, de artículos, medios de comunicación, etc. En cuanto a las ventajas (y principal diferenciador respecto a otras aplicaciones) es que, dado que no es necesario un experto en informática para manipular la información, cualquier persona sin conocimientos técnicos puede editar el contenido del portal, ya que justamente el gestor busca automatizar los procesos para mantener una página web actualizada, no obstante, esa también es su principal desventaja, ya que el personal encargado de la edición de contenido no puede modificar más allá de lo que las plantillas predefinidas le permitan.

- **Páginas enfocadas al correo electrónico:**

Es una de las aplicaciones web más utilizadas, básicamente porque en general todos los usuarios de internet cuentan con un correo electrónico para poder desempeñar sus actividades. Aquí la aplicación es muy robusta, es decir, cada vez permiten realizar más funciones como otorgar la posibilidad de consultar la información recibida, visualizar imágenes, crear contactos, adjuntar archivos, etc. Sin duda este es un buen ejemplo de lo completas que pueden llegar a ser las aplicaciones web. Su principal ventaja es la rapidez con la que la información llega al destinatario, además de que la información estará disponible en todo momento siempre y cuando esté resguardada en la denominada bandeja de entrada; incluso se puede destacar una ventaja ecológica, pues desde su utilización ya no se imprime tanta documentación para ser enviada como antes.

CAPÍTULO 4 “D3: Librería de JavaScript”

D3⁶⁵, también conocido como “d3.js”, es una biblioteca de Javascript para crear visualizaciones de datos. El nombre se origina de la abreviatura “Data-Driven Documents” (en español “Documentos Basados en Datos”): lo referente a “datos” son los proporcionados por el usuario, y los “documentos” son archivos basados en el desarrollo web, es decir, tipos de archivos con los que trabajan los navegadores: en este caso HTML y SVG⁶⁶ (siglas de Scalable Vector Graphics: un lenguaje abierto, que forma parte de HTML5, usado para realizar dibujos vectoriales en 2D). D3 es el encargado de hacer la conducción, esto significa que es quien conecta los datos a los documentos para así lograr las representaciones gráficas.

Por otro lado, resaltar la importancia de que D3 funcione con tecnologías web pues esto permite que el trabajo realizado sea visto y experimentado por cualquiera que utilice un navegador web reciente, independientemente del sistema operativo (Windows®, Mac®, Linux®). Además, D3 evita el uso de software privativo y “plug-ins” para asegurarse de que los proyectos elaborados con ésta herramienta sean accesibles para el mayor número de usuarios posible.

4.1 Introducción a D3

D3 realiza principalmente 4 procedimientos:

- Cargar datos en la memoria del navegador.
- Vincula esos datos a elementos dentro del documento, también crea nuevos elementos según vaya siendo necesario.
- Transforma esos elementos, mediante su interpretación, a propiedades visuales (vistas).

65 Murray, S. (2010). An introduction to designing with D3: Interactive Data Visualization. Estados Unidos: O’Reilly.

66 Aznar, J.(2014). “Qué es SVG”. Recuperado el 5 de abril de 2017. Disponible en:
<https://desarrolloweb.com/articulos/que-es-svg.html>

- Realiza modificaciones en las vistas en respuesta a las entradas del usuario.

El tercer paso, el de transformación, es muy importante ya que aquí es donde ocurre el “mapeo”⁶⁷, es decir, D3 proporciona una estructura para aplicar las interpretaciones visuales y el programador define las reglas. Por ejemplo, se proporciona información sobre qué tan altas deberán ser las barras en una gráfica, qué tan gruesos los círculos, si se debe clasificar por X o por Y, la paleta de colores que se ocupara, en fin, todas las decisiones de diseño visual recaen en el programador, pues D3 se limita a presentarlas y ejecutarlas.

Por otro lado, para poder usar D3 basta con dirigirse a la dirección oficial del proyecto (<https://github.com/d3/d3/releases/latest>) y descargar los archivos más recientes para agregarlos al proyecto. Existen 2 tipos de versiones disponibles de D3, la conocida como “versión normal” y otra como la “versión minificada”⁶⁸, ésta última es a la que se le han quitado los espacios en blanco (saltos de línea, sangrías, etc.) para reducir el tamaño de los archivos y lograr que sean más pequeños consiguiendo así tiempos de ejecución y descarga más rápidos; la funcionalidad es la misma, pero normalmente se utiliza la versión normal mientras se trabaja en un proyecto (para un análisis cómodo) y se cambia a la versión minificada una vez que se ha lanzado la aplicación al público y se puedan obtener tiempos de carga óptimos.

67 Rivera, C. (2013). “Un poco de acerca del mapeo de la información”. Consultado el 18 de abril de 2017. Disponible en <http://www.infotecarios.com/mapeo-de-la-informacion/>

68 Google Developers. (2014). “Minificar recursos (HTML, CSS y Javascript)”. Consultado el 20 de abril de 2017. Disponible en <https://developers.google.com/speed/docs/insights/MinifyResources?hl=es-419>

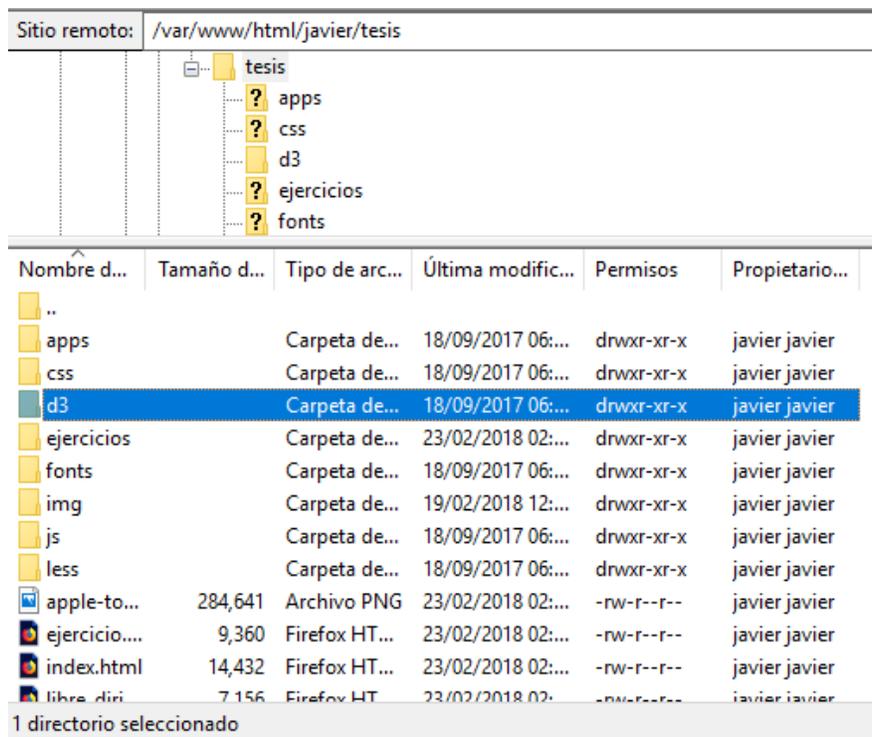


Figura 16. La ubicación de la carpeta que contiene la librería D3 debe ser en la raíz del proyecto para su invocación.

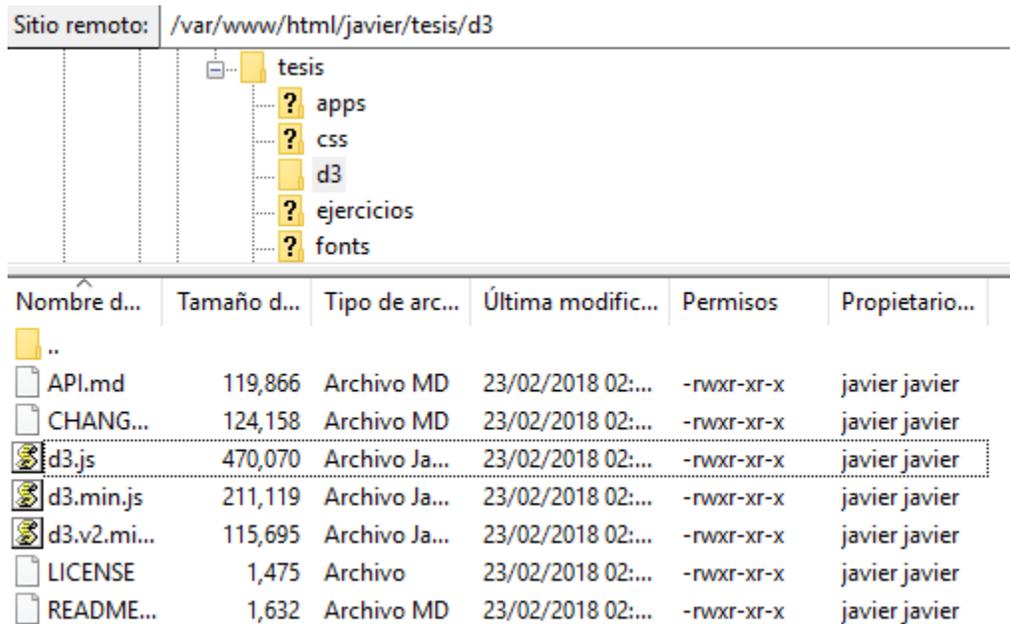


Figura 17. Contenido de la carpeta con la librería D3, aquí se encuentra la versión convencional y la minificada.

También se debe considerar que literalmente a D3 no le interesa la compatibilidad con navegadores muy antiguos, el motivo es porque persigue la filosofía de mantener un código limpio, libre de fallas en la

seguridad por dar soporte a versiones viejas y vulnerables de exploradores (Internet Explorer®, por ejemplo). Por ello, si se dice actualizar a la versión de D3 más reciente para realizar un proyecto, se deben considerar ese factor, en todo caso la solución será pedirle al usuario que actualice su navegador preferido y exhortarlo a que no continúe ocupando, por ejemplo, un versión Mozilla Firefox® de hace más de 1 año.

4.2 SVG

SVG es, básicamente, una imagen hecha a base de texto, esto es porque cada imagen SVG se define utilizando código de marcado similar a HTML (pues está basado en XML) y el código se puede incluir directamente en cualquier documento con extensión “.html”. Todos los navegadores web admiten y reconocen SVG a excepción de Internet Explorer 8 y versiones anteriores.

Para empezar a usar SVG (tanto en D3 como simplemente en un documento HTML) se debe definir el espacio a utilizar. Para entender mejor este principio se ocupa mucho la analogía de interpretar a SVG como un “lienzo” que contendrá la imagen que se va a crear y por ello hay que especificar como mínimo el ancho y alto que tendrá, además, la medición predeterminada para ambos son los “px” (píxeles) por lo que al sólo escribir un número (por ejemplo: `<svg width="500" height="50"></svg>`) no es necesario especificar que se refiere a píxeles pues ya está predefinido así, sin embargo, SVG también admite diferentes tipos de unidades de medida comúnmente utilizadas como son: “in” (inches o pulgadas en español), “cm” (centímetros), “mm” (milímetros) o “pt” (puntos) por mencionar algunos.

Otro aspecto importante a resaltar, que influye directamente al trabajar con D3, es el sistema de coordenadas. Aquí se parte del principio en que la coordenada principal “0,0” (“X, Y”) está ubicado en la parte superior izquierda del espacio destinado al gráfico. Al incrementar los valores “X”, estos se mueven hacia la derecha, mientras que el incrementar los valores “Y”, se mueven hacia abajo.



Figura 18. Representación de las coordenadas dentro de SVG⁶⁹

SVG permite generar tipos predefinidos de figuras para lo que bastará declararlos dentro de las etiquetas `<svg></svg>`; están los círculos (`circle`), elipses (`ellipse`), rectángulos (`rect`), líneas (`line`) y texto simple (`text`), además de permitir modificar su color, bordes, transparencia, tamaño, etc; sólo es necesario escribirlo directamente en el código o incluso utilizando CSS para definir estos aspectos.

4.2 Javascript y patrones

Un patrón⁷⁰, en su definición más general, se describe como “*a theme of recurring events or objects it can be a template or model which can be used to generate things*” y que a su vez traducido puede entenderse como “*un conjunto de sucesos o situaciones recurrentes; puede ser una plantilla o un modelo que se utilice para generar cosas*”. En Javascript (y en el desarrollo de software en general, por supuesto incluido el web), un patrón es una solución a un problema común, y no es necesariamente un trozo de código listo para copiar y pegar, sino más bien una mejor práctica, un razonamiento apropiado, un algoritmo útil que resuelve diversos tipos de problemas.

4.2.1. Tipos de patrones

Existen 3 tipos de patrones:

- Patrones de diseño

69 Imagen tomada de: Murray, S. (2010). An introduction to designing with D3: Interactive Data Visualization. Estados Unidos: O'Reilly.

70 Stefanov, S. JavaScript Patterns. Estados Unidos: O'Reilly & Yahoo! Press.

- Patrones de codificación
- “Antipatterns” (anti-patrones)

Los patrones de diseño⁷¹ son soluciones para problemas típicos y recurrentes que nos podemos encontrar a la hora de desarrollar una aplicación. Resulta lógico pensar que alguien se haya enfrentado a un reto similar al nuestro y por ello se puede modelar una solución similar a la que se ocupó, es decir, la solución al problema puede estar ya hecha, no hay que “reinventar la rueda”, se puede aprovechar una solución común a un problema que otra gente ya ha resuelto antes.

Los patrones de codificación⁷² son los que sirven y aplican específicamente para un lenguaje de programación.

Los anti-patrones⁷³ (Antipatterns) son lo contrario (lógicamente) a un patrón, ya que si se ha descrito con anterioridad que el patrón es “una buena práctica”, el antipatrón es una “una mal camino para llegar a una buena resolución”, o en otras palabras, son situaciones que representan una mala solución a un problema pero que sí resuelven el problemática original, no obstante, es muy probable que traigan consigo nuevos obstáculos o dificultades.

4.2.2. Utilidad de los patrones

1. Ahorran tiempo

Buscar siempre una nueva solución a los mismos problemas reduce la eficacia del programador porque se está perdiendo tiempo en el proceso de resolver algo que, paradójicamente, ya está resuelto. Es simple: en muchas ocasiones habrá reglas comunes para solucionar problemas comunes.

71 Fernández, R. (2014). “Patrones de diseño: qué son y por qué debes usarlos”. Consultado el 10 de junio de 2017. Disponible en <https://www.genbetadev.com/metodologias-de-programacion/patrones-de-diseno-que-son-y-por-que-debes-usarlos>

72 Garzás, J. (2014). “¿Qué tipos de patrones hay?”. Consultado el 10 de junio de 2017. Disponible en <http://www.javiergarzas.com/2014/08/tipos-patrones-software.html>

73 Garzás, J. (2014). “¿Qué tipos de patrones hay?”. Consultado el 10 de junio de 2017. Disponible en <http://www.javiergarzas.com/2014/08/tipos-patrones-software.html>

2. Otorgan confiabilidad al código

Los patrones de diseño son estructuras probadas por varios programadores a lo largo de muchos años, por lo que si se elige el patrón adecuado para modelar el problema, se puede estar seguro de que va a ser la solución más válida que se puede encontrar.

3. Establecen un lenguaje común

Modelar el código mediante patrones ayudará a que otras personas puedan entender cómo se ha solucionado el problema. Los otros programadores van a comprender lo que se ha implementado, cómo y por qué, además ayuda a descubrir rápidamente si esa era la mejor solución o no. Los patrones de diseño ayudarán a todo un equipo de trabajo a avanzar mucho más rápido con un código fácil de entender para todos.

4.2.3. Patrones en la aplicación web interactiva de grafos

Dentro de la aplicación web interactiva presentada en este trabajo de tesis, se ocupan patrones (específicamente patrones de diseño) gracias a la utilización del framework Bootstrap. Es así que, en otras palabras, Bootstrap está formado por un grupo de archivos .html, .css y .js (JavaScript) que contienen patrones de diseño con los cuales se puede desarrollar un sitio web óptimo evitando perder tiempo reescribiendo código repetitivo.

La finalidad de ocupar Bootstrap es para crear una interfaz de usuario limpia y totalmente adaptable a cualquier tipo de dispositivo y tamaño de pantalla.

```
bootstrap.js index.html x
1 <!DOCTYPE html>
2 <html class="no-js">
3   <head>
4     <meta charset="utf-8">
5     <meta http-equiv="X-UA-Compatible" content="IE=edge">
6     <title>Aplicación web interactiva para el estudio y edición de grafos</title>
7
8     <link rel="icon" type="image/png" href="img/intro.png" />
9
10    <link rel="apple-touch-icon" href="apple-touch-icon.png">
11
12    <meta name="description" content="Aplicación Web Interactiva para la FES Cuautitlán, UNAM.">
13    <meta name="keywords" content="teoria de grafos, FES, unam">
14    <meta name="author" content="Javier Alejandro Olguin Mendoza">
15    <meta name="robots" content="INDEX,FOLLOW">
16
17    <meta name="viewport" content="width=device-width, initial-scale=1">
18
19    <!-- CSS -->
20    <link rel="stylesheet" href="css/owl.carousel.css">
21    <link rel="stylesheet" href="css/bootstrap.min.css">
22    <link rel="stylesheet" href="css/font-awesome.min.css">
23    <link rel="stylesheet" href="css/style.css">
24    <link rel="stylesheet" href="css/ionicons.min.css">
25    <link rel="stylesheet" href="css/animate.css">
26    <link rel="stylesheet" href="css/responsive.css">
27
28    <!-- Js -->
29    <script src="js/vendor/modernizr-2.6.2.min.js"></script>
30    <script src="//ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"></script>
31    <script>window.jQuery || document.write('<script src="js/vendor/jquery-1.10.2.min.js"></script>')</script>
32    <script src="js/bootstrap.min.js"></script>
33    <script src="js/owl.carousel.min.js"></script>
34    <script src="js/plugins.js"></script>
35    <script src="js/min/waypoints.min.js"></script>
36    <script src="js/jquery.counterup.js"></script>
```

Figura 19. Invocación de los archivos propios de Bootstrap en la aplicación web (CSS y JS).

Una vez que se invocaron los archivos pertinentes se pueden empezar a usar las herramientas que otorga el framework, y específicamente en el caso de los patrones se utilizarán las de la clase “container”.

“Container” (o en español contenedor), permite dividir el área de trabajo en doce columnas que se pueden combinar de diversas maneras para adaptarse a diferentes resoluciones de pantalla, esto permite una gran flexibilidad al momento de crear la interfaz de usuario.

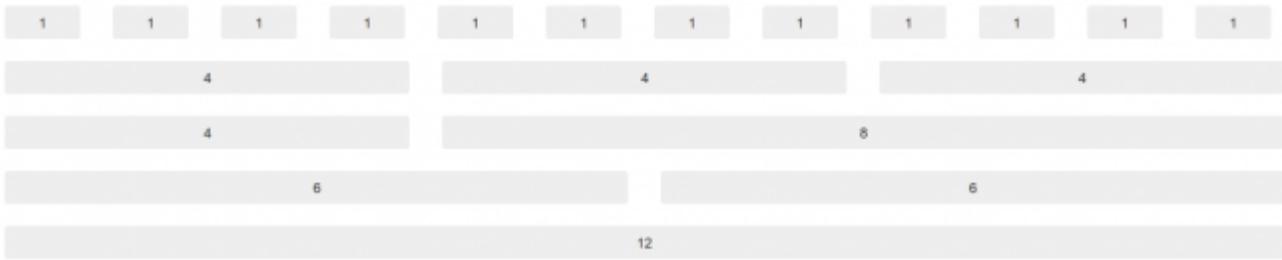


Figura 20. Gráfico de las 12 columnas que se generan usando “container”⁷⁴

Se puede ocupar la columna “12” para agregar un elemento que ocupe toda la página (por ejemplo un menú o un pie de página), o bien una combinación de números que en conjunto den el mismo número 12, por ejemplo 3 columnas “4” (izquierda, central y derecha) o una combinación de columna “8” con columna “4”; el requisito es que sea un número que no rebase el 12 para que “container” sepa dónde ubicarlo.

Por ejemplo, para crear la barra de menú de la aplicación web interactiva, se especifica que se utilizará “col-md-12”, es decir, la columna “12” (que es la que ocupa toda el área de trabajo, según la figura 20), y con la palabra “md” se contempla que el diseño está dirigido a dispositivos con resoluciones medianas (una anchura máxima de 940px, la cual sería para una computadora convencional); también existen las definiciones “xs” (para los teléfonos celulares con una anchura máxima generada automáticamente, pero usualmente menor a los 728px), “sm” (para una anchura máxima de 728px, perteneciente a las tabletas) o “lg” (para una anchura máxima de 1200px, propia de monitores más grandes).



Figura 21. Apariencia de la barra de menú

74 Imagen tomada de: Borillo, R. (2016). “Diseñando tu nuevo proyecto con Bootstrap”. Disponible en: <https://www.genbetadev.com/desarrollo-web/disenando-tu-nuevo-proyecto-web-con-bootstrap-2-0>

```

52 <header>
53 <div class="container">
54 <div class="row">
55 <div class="col-md-12">
56 <nav class="navbar navbar-default">
57 <div class="container-fluid">
58 <!-- Brand and toggle get grouped for better mobile display -->
59 <div class="navbar-header">
60 <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target="#bs-example-navbar-collapse-1">
61 <span class="sr-only">Toggle navigation</span>
62 <span class="icon-bar"></span>
63 <span class="icon-bar"></span>
64 <span class="icon-bar"></span>
65 </button>
66 <a class="navbar-brand" href="index.html">
67 
68 </a>
69 </div>
70 <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
71 <ul class="nav navbar-nav navbar-right">
72 <li><a href="index.html">Inicio</a></li>
73 <li><a href="teoria.html">Teoría de Grafos</a></li>
74 <li><a href="ejercicio.html">Ejercicios</a></li>
75 <li><a href="libre_dirigido.html">Edición Grafo Dirigido</a></li>
76 <li><a href="libre_nodirigido.html">Edición Grafo No Dirigido</a></li>
77 </li>
78 </ul>
79 </div><!-- /.navbar-collapse -->
80 </div><!-- /.container-fluid -->
81 </nav>
82 </div>
83 </div>
84 </div>
85 </header><!-- header close -->

```

Figura 22. Código con la sección “container”.

Tomando como referencia la figura 22, mencionar también que los elementos pertenecientes al menú se definirán con una lista de HTML (y), en este caso: inicio, teoría de grafos, ejercicios, edición de grafo dirigido y edición de grafo no dirigido.

Una de las principales ventajas de Bootstrap es que fácilmente se adapta a dispositivos móviles, precisamente haciendo uso de patrones de diseño para conseguir tal objetivo, por ello es que en el espacio donde se define la barra de menús también se debe escribir cómo reaccionará la aplicación al visualizarse en un teléfono celular. Ver figura 23.

```

52 <header>
53 <div class="container">
54 <div class="row">
55 <div class="col-md-12">
56 <nav class="navbar navbar-default">
57 <div class="container-fluid">
58 <!-- Brand and toggle get grouped for better mobile display -->
59 <div class="navbar-header">
60 <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target="#bs-example-navbar-collapse-1">
61 <span class="sr-only">Toggle navigation</span>
62 <span class="icon-bar"></span>
63 <span class="icon-bar"></span>
64 <span class="icon-bar"></span>
65 </button>
66 <a class="navbar-brand" href="index.html">
67 
68 </a>
69 </div>
70 <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
71 <ul class="nav navbar-nav navbar-right">
72 <li><a href="index.html">Inicio</a></li>
73 <li><a href="teoria.html">Teoría de Grafos</a></li>
74 <li><a href="ejercicio.html">Ejercicios</a></li>
75 <li><a href="libre_dirigido.html">Edición Grafo Dirigido</a></li>
76 <li><a href="libre_nodirigido.html">Edición Grafo No Dirigido</a></li>
77 </li>
78 </ul>
79 </div><!-- /.navbar-collapse -->
80 </div><!-- /.container-fluid -->
81 </nav>
82 </div>
83 </div>
84 </div>
85 </header><!-- header close -->

```

Figura 23. Código con la sección Container especificando el tratamiento para dispositivos móviles

Se sigue tomando como referencia “container” y “col-md-12” para definir el espacio que ocupará en pantalla, sin embargo, después de “nav-header” (que es la etiqueta que define la barra de navegación) se menciona que cuando haya un colapso (o sea que la pantalla sea más chica de lo que “md” indica), se genere un botón en la columna “1” (retomando la figura 20, indica una columna mucho más pequeña que la “12”) y ese botón creado a su vez tendrá otras características de diseño.



Figura 24. Visualización del menú de la aplicación web en dispositivos móviles.

La utilización de la clase “container” representa un ahorro significativo de tiempo y escritura de código para el diseño de interfaces pues, al dividir el área de trabajo en columnas, se le facilita al programador ubicar los elementos donde lo considere necesario.

Otro de los patrones que significan un apoyo importante para el programador es el que se define en la clase “img-responsive”. En la realización de un proyecto se ocupan imágenes de diversos tamaños y, al ser de diferentes resoluciones, trabajar con ellas podría generar el caso de que los navegadores no permitan visualizarlas correctamente en la aplicación; no obstante, cuando se vuelven “responsivas”, se indica que su tamaño dependerá del espacio que brinde la pantalla del dispositivo ayudando a solventar ésta problemática.

```
101
102 <!-- Textos de teoria de grafos -->
103 <section id="teoria-de-grafos">
104   <div class="container">
105     <div class="row">
106
107       <strong>Definición.</strong>
108       <br><br>
109
110       La teoría de grafos tiene su origen en el año 1736 cuando Leonhard Euler publicó un trabajo que daba seguim:
111       conocido como "Los puentes de Königsberg". Dicho problema surgió en la ciudad de Prusia (en la actualiad Ka:
112       donde el río Pregel recorre el poblado y también dos islas (de nombre Kneiphof) que están en el centro, cau:
113       quede dividida en cuatro partes: dos regiones a cada lado del río y las islas de en medio; todas ellas unid:
114       puentes. La siguiente imagen ejemplifica las regiones del lugar:
115       <br><br>
116       <center></center>
117       <br><br>
118
119       Ante esta situación, muchas personas comenzaron a averiguar si era posible cruzar los siete puentes pasando
120       por cada uno de ellos. Euler demostró que esto no era posible, y para probarlo elaboró un modelo con un grup
121       representaban las áreas de tierra y líneas que simbolizaban los puentes; fue así como surgió el primer ejem:
122       <br><br>
123       <center></center>
124       <br><br>
125
126       Tomando como referencia la segunda imagen, las flechas indican el camino que se toma para recorrer los cuat:
127       se puede notar, es necesario cruzar al menos dos veces por algunos enlaces para poder cumplir el ciclo.
128       <br><br>
129
130
```

Figura 25. Clase "img-responsive" dentro de la aplicación web

Su inclusión es muy sencilla, dentro de la etiqueta se debe invocar a la clase "img-responsive" para poder hacer uso de las ventajas anteriormente mencionadas. Por su parte, dentro de la documentación de Bootstrap⁷⁵, se especifica que la clase "img-responsive" aplica otras clases más de CSS como son "display:block" (para estirar la imagen tan lejos como se pueda), "max-width: 100%" (para indicar que la imagen puede ser tan ancha como su tamaño predefinido y no más, básicamente para no estirar demasiado el recurso y que no se vea "píxeleado") y "height:auto" (para que la altura la establezca el navegador en función al tamaño de la ventana).

75 W3Schools (2017). "Bootstrap Images". Disponible en: https://www.w3schools.com/bootstrap/bootstrap_images.asp

Ante esta situación, muchas personas comenzaron a averiguar si era posible cruzar los siete puentes pasando solamente una vez por cada uno de ellos. Euler demostró que esto no era posible, y para probarlo elaboró un modelo con un grupo de puntos que representaban las áreas de tierra y líneas que simbolizaban los puentes; fue así como surgió el primer ejemplo de un grafo.

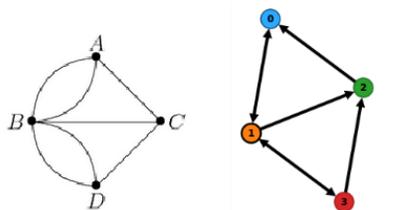


Figura 2. Imagen del grafo de Euler. A su lado está dicho grafo representado en la aplicación web.

Tomando como referencia la segunda imagen, las flechas indican el camino que se toma para recorrer los cuatro puntos y, como se puede notar, es necesario cruzar al menos dos veces por algunos enlaces para poder cumplir el ciclo.

A partir del trabajo de Euler, la teoría de grafos fue convirtiéndose en una útil herramienta para crear modelos en temas tan dispares como las matemáticas, la química, la física, las telecomunicaciones, internet, la economía, el estudio de probabilidades, temas de planificación, sistemas GPS, en redes neuronales, en programación, etc; propiamente en gran parte de la Informática aparecen los grafos, especialmente los grafos de árbol y los grafos dirigidos. Los diagramas de flujo, por ejemplo, son grafos dirigidos.

Enfocando la teoría de grafos a la informática, se puede elaborar un sencillo ejemplo: Representar una red de comunicaciones donde la colocación ideal de los elementos evite interrupciones accidentales, en otras palabras, que no haya desconexión de la red. El siguiente grafo expone que no existe ninguna línea que, al ser eliminada, desconecte la red.

Figura 26. Imagen presente en la aplicación web vista en computadora y en su tamaño predefinido.

Ante esta situación, muchas personas comenzaron a averiguar si era posible cruzar los siete puentes pasando solamente una vez por cada uno de ellos. Euler demostró que esto no era posible, y para probarlo elaboró un modelo con un grupo de puntos que representaban las áreas de tierra y líneas que simbolizaban los puentes; fue así como surgió el primer ejemplo de un grafo.

Figura 2. Imagen del grafo de Euler. A su lado está dicho grafo representado en la aplicación web.

Tomando como referencia la segunda imagen, las flechas indican el camino que se toma para recorrer los cuatro puntos y, como se puede notar, es necesario cruzar al menos dos veces por algunos enlaces para poder cumplir el ciclo.

A partir del trabajo de Euler, la teoría de grafos fue convirtiéndose en una útil herramienta para crear modelos en temas tan dispares como las matemáticas, la química, la física, las telecomunicaciones, internet, la economía, el estudio de probabilidades, temas de planificación, sistemas GPS, en redes neuronales, en

Figura 27. Imagen presente en la aplicación web siendo re-escalada para acoplarse a un tamaño menor (dispositivo móvil).

Por último, y como dato importante, además de definirse los patrones dentro de la aplicación, Bootstrap necesita que se declare al inicio la sentencia siguiente:

```
12 <meta name="description" content="Aplicación Web Interactiva para la FES Cuautitlán, UNAM.">
13 <meta name="keywords" content="teoria de grafos, FES, unam">
14 <meta name="author" content="Javier Alejandro Olguín Mendoza">
15 <meta name="robots" content="INDEX,FOLLOW">
16
17 <meta name="viewport" content="width=device-width, initial-scale=1">
18
```

Figura 28. Definición de etiqueta <meta> con “viewport”, indicadora del tamaño de la ventana.

“Viewport” se refiere al tamaño de la ventana donde se visualizará la aplicación y en “content” se está especificando que el tamaño de la misma será la que el ancho del dispositivo pueda otorgar. El objetivo de esto es contemplar desde el principio que muchas de las funciones trabajan regidas por el ancho que brinda el dispositivo donde se ejecuta la aplicación.

4.3 Aplicaciones de D3

D3 actualmente esté presente en muchas aplicaciones en el mundo y son de lo más variadas y útiles; ejemplos existen muchos y aunque son diversos, todos comparten una idea principal: La representación gráfica de los datos.

Por mencionar algunas aplicaciones, se puede destacar:

4. World Bank Global Development Sprint⁷⁹

Se trata de una aplicación que permite ver los tratados bancarios que realiza un país con otro a través de una animación: Aparece una línea que conecta al emisor del contrato con el receptor. Utiliza datos anuales del Banco Mundial en fechas que van desde 2013, 2012, 2011, etc. Sirve como guía visual para ver la distribución de los principales compromisos contractuales entre los países miembros del Banco.

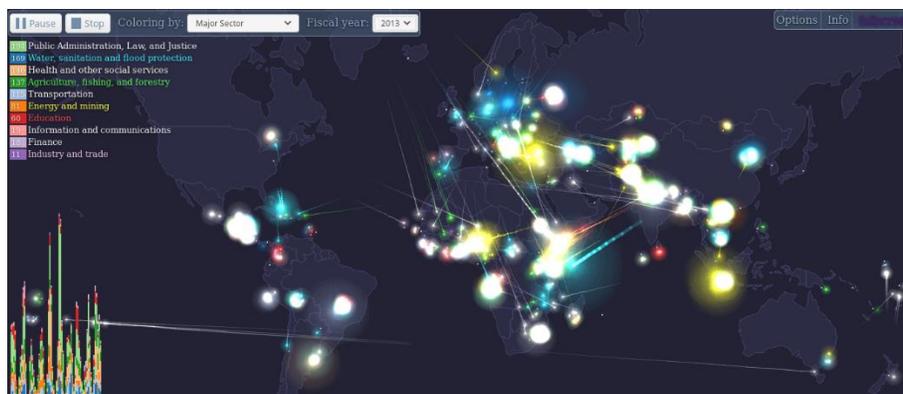


Figura 32. Aplicación D3 “World Bank Global Development Sprint”

Mencionar estas aplicaciones tiene como finalidad destacar que D3 está presente en numerosos proyectos e incluir sus capacidades a la aplicación web desarrollada en esta tesis significa estar a la vanguardia en cuanto a desarrollo web se refiere.

Además, al contar con gran apoyo por parte de su comunidad, D3 continuamente se está actualizando y ofreciendo nuevas funciones que se pueden aprovechar en los proyectos que se estén realizando.

La aplicación web interactiva para el estudio y edición de Grafos propuesta utiliza software moderno y competitivo con el objetivo de acercar a los alumnos de la Facultad de Estudios Superiores Cuautitlán a las nuevas tendencias informáticas y motivarlos a ocuparlas.

79 Zubkov (2013). “World Bank Global Development Sprint”. Consultado el 22 de junio de 2017. Disponible en: <http://d3.artzub.com/wbca/>

CAPÍTULO 5 “Desarrollo de la Aplicación”

La aplicación web interactiva se desarrolló buscando cubrir los objetivos trazados desde el protocolo de la tesis y siguiendo una metodología que se acopla al desarrollo web. A lo largo de éste capítulo se describe cómo se elaboró el software y la influencia de la metodología para lograr la meta trazada.

5.1 Metodología

La metodología empleada está basada en las ideas de 2 autores, por un lado Ana Nieves del Valle Rodríguez (2009) con su trabajo “Metodologías de Diseño usadas en Ingeniería Web, su vinculación con las Ntics” quien a su vez toma referencias de Roger S. Pressman (2009) de su libro “Web engineering: a practitioner's approach”. Se enumeran siete actividades que forman parte del proceso del desarrollo en internet y encajan en cualquier aplicación web independientemente de su tamaño o complejidad.

Sobre los beneficios que la metodología brinda, principalmente está que va dirigida específicamente al desarrollo web, no se trata de adaptar alguna otra ya existe al desarrollo en internet sino que ésta misma incluye secciones como “generación de páginas web” que hace referencia al momento en que es adecuado desarrollar ése importante elemento, además de la dirección a seguir.

Otro punto que determinó la elección de ésta metodología va en relación a que, efectivamente, se notó que es aplicable a diversos tipos de aplicaciones web sin importar la magnitud del proyecto, alcances y complejidad; los 7 puntos que abarca permiten encaminar un trabajo profesional exitoso. De hecho, esta metodología también se ocupa usualmente en los trabajos que realiza el tesista en su empleo.

Las actividades que conforman la metodología son:

1. **Formulación:** Se necesitan identificar los objetivos a perseguir y establecer los límites respecto hasta donde se quiere llegar con el proyecto.

2. **Planificación:** En un proyecto convencional aquí es donde se definen los costos, sin embargo, en el caso de la tesis, éstos son mínimos pues económicamente hablando se ocupó Software Libre además de que el desarrollador lógicamente no cobró honorarios. También se determinan los riesgos que se pueden presentar además de definir las fechas de los entregables y proyecto final.
3. **Análisis:** Sección dedicada a especificar los requerimientos que va a cubrir la aplicación web, es decir, qué llevará dentro de sí el proyecto desarrollado. También se debe considerar, de ser el caso, si se necesita ocupar un servidor en particular, algún tipo de computadora, un espacio de trabajo específico, etc.
4. **Modelización:** Se compone de las tareas de diseño y producción del contenido dentro de la aplicación, también del diseño de la arquitectura, navegación e interfaz de usuario. Es importante destacar la importancia del diseño de la interfaz pues, independientemente del contenido, una buena interfaz mejora la percepción que el usuario tiene de éste.
5. **Generación de páginas:** Tomando lo hecho en el punto anterior, se integra todo el contenido y la arquitectura de la navegación del usuario, para crear (estática o dinámicamente, responsivo o no responsivo) el punto central y más visible de la aplicación: las páginas web.
6. **Test:** Tiene la finalidad de encontrar posibles errores en el contenido, navegación y rendimiento. De encontrarse alguna falla, aquí es donde se realizan las correcciones pertinentes. En la metodología se hace mención a que existen muchas fallas relacionadas no sólo en cuanto a la programación, también influyen factores como la conexión a internet, el tipo de servidor, sistema operativo del servidor, etc.
7. **Evaluación del cliente:** El resultado final es sometido a la evaluación del cliente. En este caso es la exposición de la tesis ante los sinodales.

En paralelo a los 7 puntos, la metodología también hace mención a la calidad, un factor que debe estar presente en todo momento del desarrollo. Para conseguir dicha calidad, se tienen que cuidar aspectos como:

- Usabilidad: Se consigue cuando la página web ofrece una lectura sencilla del texto presente, carga rápida de imágenes o recursos y un menú claro y visible.
- Funcionalidad: Significa asegurar que la aplicación funciona tal y como se espera.

- Finalidad: El software se desarrolló persiguiendo un objetivo, es decir, solucionar alguna problemática. La finalidad es asegurar que dicha meta se consiguió.
- Seguridad: Se debe proteger el software manteniendo la integridad del mismo evitando que los usuarios puedan alterar los resultados que se arrojan y, cuando es el caso, cuidar la privacidad de los datos que las personas almacenan en las aplicaciones.
- Mantenimiento: Después de que se entregó la aplicación web al cliente, la idea es ofrecer la posibilidad de actualizar el software (o hacer modificaciones) de manera sencilla.

5.1.1. Formulación

La formulación va de la mano del objetivo que tiene la tesis y que se planteó desde el protocolo de la misma, por ende, será abordada en el subtema 5.1.1.2 titulado “Objetivo”. Además, también se debe especificar el software a ocupar, que en éste caso es software libre y propiamente será detallado en el subtema 5.1.1.3 “Software a utilizar”.

5.1.1.2. Objetivo

El objetivo es: Desarrollar una aplicación web que permita a los alumnos de la Facultad de Estudios Superiores Cuautitlán contar con una alternativa para el estudio de la “Teoría de Grafos”, proporcionándoles una opción dinámica que responda a la inclinación actual del uso de internet y la “nube” como medio efectivo de apoyo al aprendizaje.

La formulación retoma este planteamiento como lo principal a perseguir. La intención de que dicha opción de estudio esté disponible en internet es dada la portabilidad que otorga esa tecnología, los estudiantes podrán acceder a la plataforma desde cualquier parte (su casa, su trabajo, etc.) con solo ingresar la dirección web correspondiente (y destacando la necesidad de contar con una conexión activa a internet).

5.1.1.3. Software a utilizar

Respecto a la aplicación web como tal, se decidió ocupar la tecnología D3 por las amplias posibilidades que ofrece tanto para trabajar visualizaciones como el manejo de datos, además se trata de Software Libre incluyendo las bondades para el programador que ello significa. Hablando específicamente de lenguajes programación, se ocupó JavaScript como principal motor así como también el uso de los estándares HTML y CSS para las vistas de la página web.

También se ocupó el sistema operativo GNU/Linux en sus distribuciones “Debian” y “Linux Mint” para almacenar la aplicación en un servidor (por el lado de “Debian”) y el trabajo propio del desarrollo que ejecutó el programador fue en “Linux Mint”.

Por otro lado, para la generación de las páginas web (y principal elemento visual que percibe el usuario), se utilizó Bootstrap. Éste framework permite responder a la tendencia actual de programar aplicaciones web que, además de ser visibles en computadoras de escritorio o laptops, también puedan ejecutarse de manera responsiva en dispositivos móviles como celulares o tabletas.

Además de lo anterior y aprovechando las bondades de JavaScript, se utilizó “AlertifyJS”⁸⁰, un framework de código abierto dedicado a la generación de ventanas de diálogo y notificaciones dentro de la aplicación, se generan animaciones que resultan visualmente atractivas para el usuario y permiten personalizar las notificaciones que arroja el navegador web (para no utilizar las que ya vienen predefinidas).

5.1.2. Planificación

La planificación es el decir: ¿hasta dónde llegará el proyecto? ¿Qué riesgos hay y cómo enfrentarlos? También se debe hablar sobre el calendario de desarrollo, las fechas de entrega y reuniones con el cliente, aunque éste último punto aplica solamente para los proyectos que desde el inicio establecen fechas de

80 AlertifyJS (2017). “AlertifyJS”. Disponible en: <http://alertifyjs.com/>

reunión para ir mostrando avances, hay algunos proyectos en los que sólo interesa mostrar el producto final al ser concluido.

Respecto a definir hasta dónde llegará el proyecto, la intención que se tiene es que el software sea aplicado primeramente en los alumnos de la Licenciatura en Informática a partir de su 4to semestre, cuando en la materia de “Matemáticas Computacionales” se aborde el tema de “Teoría de Grafos”, es ahí donde la aplicación permitirá al profesor reforzar los conocimientos de los estudiantes mediante la aplicación de ejercicios disponibles en la plataforma web. No obstante, la aplicación podría escalar a otras carreras dentro de la facultad pues el tema de Teoría de Grafos es amplio e involucra varias disciplinas, en el futuro podrían agregarse más ejercicios (enfocados a cada una de las carreras dentro de la FES) y así apoyar a los alumnos en su formación.

Hablando de los riesgos durante el desarrollo de la aplicación, el principal es el propio de todas las tecnologías web que dependen de un servidor, pues si éste se “cae” (refiriéndose a que el equipo deje de funcionar por problemas en la fuente de alimentación eléctrica, conexión a internet, etc.), la aplicación no será visible para los usuarios hasta que el problema sea solucionado. Otro riesgo importante a considerar es que el estudiante depende de una conexión a internet desde el lugar donde acceda, ya que por razones propias de las tecnologías en la nube si no se tiene acceso a la red sencillamente no se puede trabajar.

Aplicando el tema del calendario de desarrollo a ésta aplicación, serían las fechas que el tesista propone junto con su asesor para ir avanzando en el proyecto; los dos llegan a un acuerdo sobre en qué fecha debe de ir avanzando el programa y hacer las pruebas pertinentes de que el software va logrando la meta planeada.

5.1.3. Análisis

Respecto al análisis, se identificaron los principales requerimientos:

1. Aplicación web interactiva.

2. Disponibilidad de información relevante (conceptos) sobre la teoría de grafos dentro de una sección en la página.
3. Posibilidad de acceder al software en cualquier momento las 24 horas del día (lo cual se obtiene gracias a la utilización de Internet).
4. Diversos ejercicios de la teoría de grafos.
5. Permitir la posibilidad de que, en el futuro, el software pueda escalar a otras carreras dentro de la Facultad aparte de la Licenciatura en Informática.

Además, se definieron los siguientes escenarios que se espera sucedan durante la utilización de la aplicación web. En primer lugar está el referente a cuando un alumno entre a la aplicación para resolver alguno de los ejercicios propuestos.

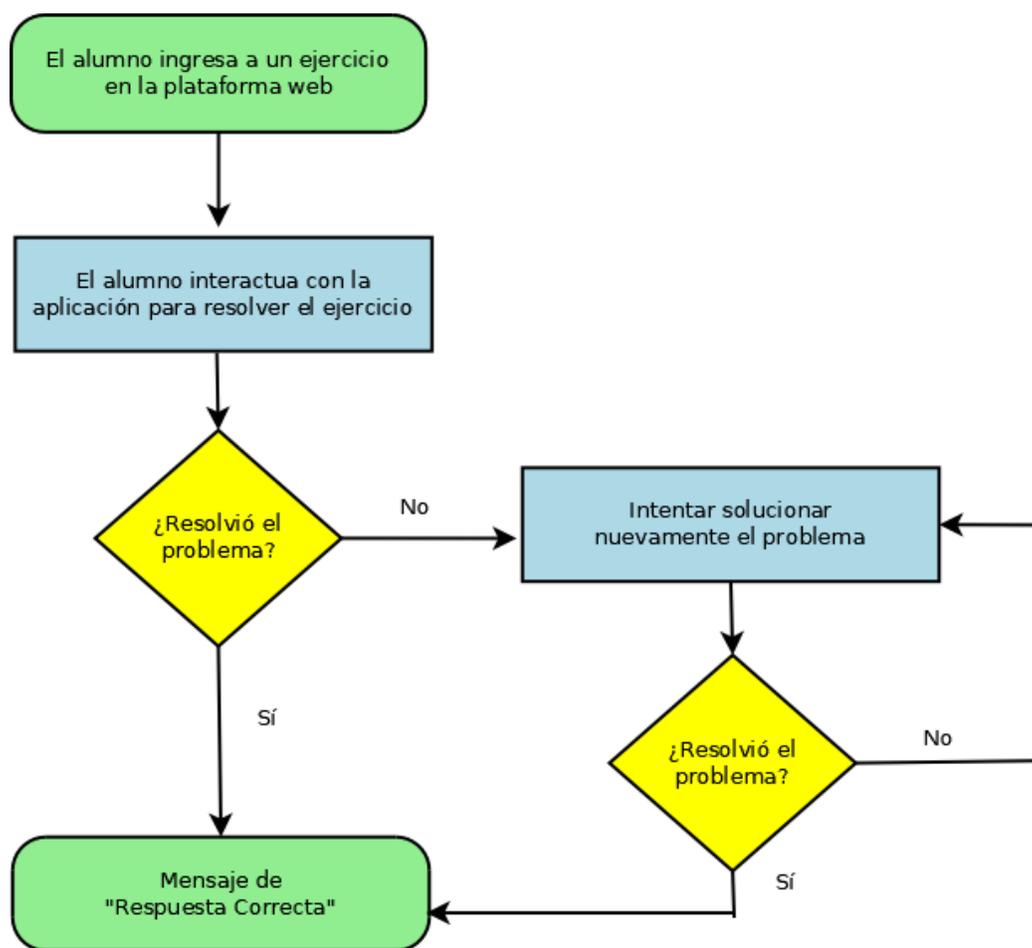


Figura 33. Diagrama de flujo cuando el alumno resuelve un ejercicio.

También se considera el escenario donde el profesor explique un tema al grupo y se apoye en la aplicación web para exponer su clase.

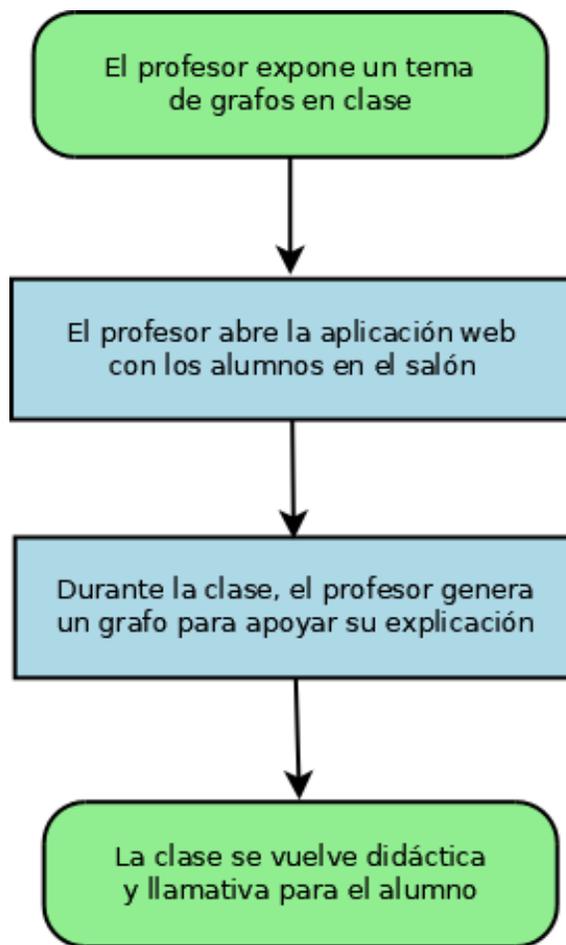


Figura 34. Diagrama de flujo cuando el profesor expone un tema generando el grafo en clase.

Por otro lado, también surge el escenario donde un alumno participa en clase haciendo uso de la aplicación web interactiva.

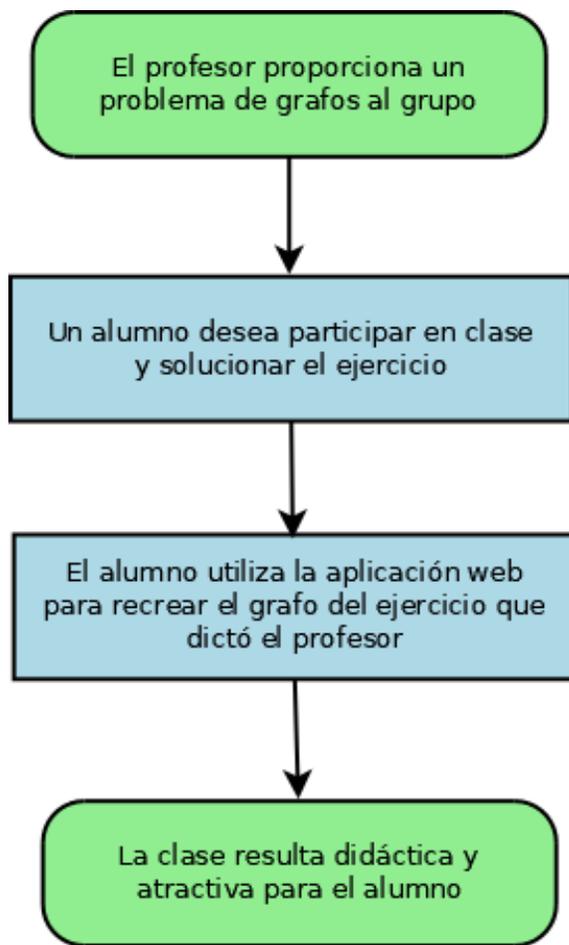


Figura 35. Diagrama de flujo cuando un alumno participa en clase.

Ya sea porque el profesor dejó tarea en clase para el alumno, o bien éste último desea poner en práctica sus habilidades, se considera el escenario donde los alumnos ingresen a la aplicación libremente para reforzar sus conocimientos.

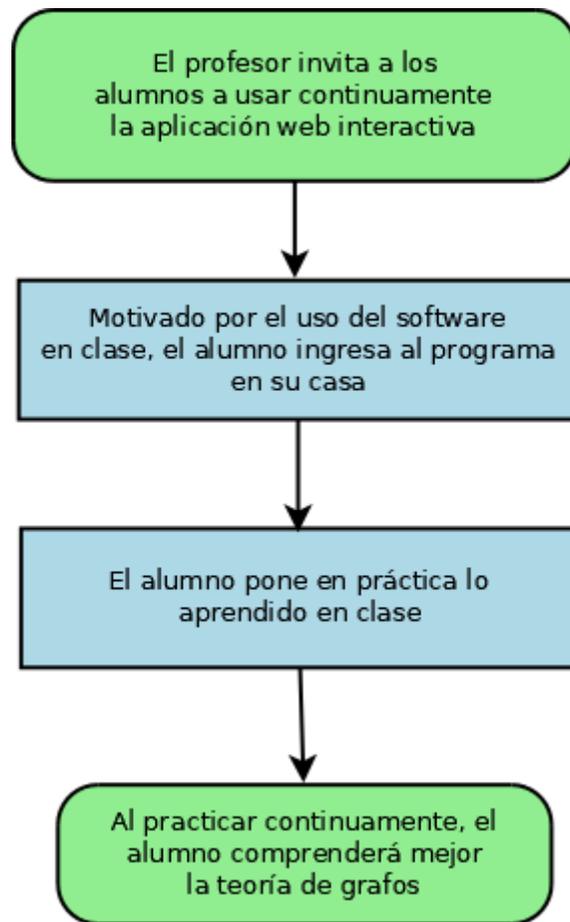


Figura 36. Diagrama de flujo cuando el alumno practica en casa.

El análisis también hace mención a la pregunta ¿de dónde se obtendrá el contenido para la aplicación? En este caso, las definiciones de la teoría de grafos y los ejercicios que están incluidos en el software, se eligieron principalmente de:

- a) Jordan, C. (1996). "Introducción a la teoría de grafos y sus algoritmos". España: Universidad Politécnica de Valencia Servicio de Publicaciones.
- b) Vieites Rodríguez, A. (2014). "Teoría de grafos". Madrid: Paraninfo.

Mientras que los recursos de D3.js, que posteriormente se modificaron conforme las necesidades de la aplicación, se obtuvieron de:

- a) Kirsling, R. (2017). “Directed Graph Editor”. Disponible en:
<http://bl.ocks.org/rkirsling/5001347>

5.1.4. Modelización

El diseño y la producción del contenido de la aplicación se elaboró tomando como referencia lo planteado en el análisis del proyecto y para cubrir lo establecido en ese punto, se generó un menú principal para la página web. Dicho menú está compuesto por: Inicio, Teoría de Grafos, Ejercicios, Edición Grafo Dirigido y Edición Grafo No Dirigido.

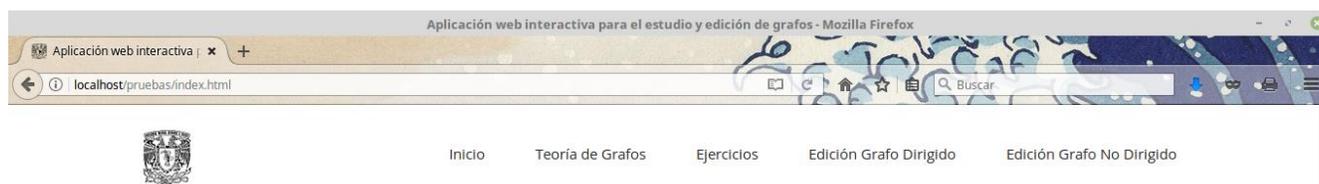


Figura 37. Menú principal de la aplicación web.

Inicio. Es la parte principal de la aplicación web, aquí se muestra un breve resumen de la teoría de grafos, ejemplos de su utilización y los 4 puntos clave que trata la aplicación: Teoría (texto), ejercicios y la edición libre tanto de grafos dirigidos como los no dirigidos.

Teoría de Grafos. Sección dedicada a los conceptos de la teoría de grafos dentro de la página web, se buscaron definiciones de varios autores que permitieran reforzar los conocimientos de los alumnos. Además, el objetivo de esta sección es servir como un recordatorio de los puntos principales que corresponden a la teoría de grafos.

Ejercicios. Es el área dedicada a la presencia de problemas y ejercicios, en ésta sección el alumno ingresará para responder algún conjunto de ejercicios que el profesor le asigne. También podría entrar por iniciativa propia a resolver ejercicios durante sus ratos de estudios y actividades autodidactas.

Edición Grafo Dirigido. Además de los ejercicios, el alumno encontrará un área en la que podrá poner en práctica sus conocimientos desarrollando algún grafo dirigido de manera libre.

Edición Grafo No Dirigido. Por su parte, el alumno también podrá crear un grafo no dirigido para poner en acción sus conocimientos.

Tomando como base el menú principal ya definido, se comenzó a desarrollar la aplicación web buscando cumplir con lo establecido ahí, es decir, fue la guía para ir llenando los campos que requería poseer el software.

En cuanto a la interfaz gráfica, se buscó un apartado visual cómodo y atractivo para el usuario aunque no por ello perdiendo la formalidad que una aplicación académica debe poseer. Además, se responde a la tendencia actual de visualizar las páginas web en cualquier dispositivo ya sea móvil o de escritorio.

5.1.5. Generación de páginas

D3 no es totalmente compatible con dispositivos móviles, si bien sí se visualiza en celulares y tabletas, esta librería tiene 2 principios que no encajan con el desarrollo de aplicaciones móviles:

1. D3 necesita, en la mayoría de las veces, de un mouse para ejecutar los eventos que el desarrollador programará en la aplicación, por ejemplo: hay una acción para el doble click, otra para dar un click y a la vez oprimir una tecla específica del teclado, otra para dar un click sostenido, etc., y no todas estas funciones pueden ser emuladas en una pantalla táctil.
2. El SVG que ocupa D3 necesita definir los pixeles en los que aparecerá el gráfico, el problema es que si se define un área de trabajo de 720 x 480 pixeles y el celular del usuario tiene una resolución de 640 x 480 pixeles, la lógica nos indica que no se podrá visualizar adecuadamente el gráfico, el usuario tendrá que estar haciendo zoom constantemente y mover el área de trabajo para poder ver correctamente las gráficas. Esto va en contra del diseño responsivo, el cual busca

que en cuanto cargue la aplicación la misma se adapte en automático a la pantalla que tiene el usuario.

A pesar de que D3 no es totalmente compatible, de momento, con los celulares o equipos móviles, al momento de generar las páginas web se optó por elegir el Framework Bootstrap⁸¹ con la finalidad de contar con un diseño dinámico que, si bien pudiera parecer contradictorio trabajar en un diseño responsivo para una aplicación que en primera instancia no lo ocupará, el objetivo es que la aplicación web propuesta en esta tesis pueda ser actualizada en el futuro y si aparece una nueva versión de D3 que esté pensada ya para integrar el uso de dispositivos móviles, entonces ejecutar una migración lo más sencilla y rápida posible, sin tener que volver a programar todo desde el principio.

5.1.6. Test

El test se realizó con el asesor de tesis. Continuamente se realizaban pruebas para verificar que la aplicación funcionara correctamente y no presentara fallas. Cuando alguna problema surgía, se procedía a tomar nota del fallo y evaluar en conjunto cuál sería la causante del error, generalmente los errores estaban en alguna parte del código y se debían modificar algunas sentencias para que el error no volviera a ocurrir.

5.1.7. Evaluación del cliente

La evaluación del cliente, específicamente para éste trabajo de tesis, se llevaría a cabo en dos partes, la primera es la evaluación final que se realiza con el asesor de tesis donde verificará que todos los objetivos se hayan cumplido satisfactoriamente. Por otro lado, el día del examen profesional y exposición de la aplicación, el jurado también evaluará que la aplicación cumpla con los puntos establecidos tanto en el protocolo como en la tesis como tal.

81 Otto, M. (2017). "Bootstrap. The most popular HTML, CSS, and JS framework in the world". Disponible en: <https://v4-alpha.getbootstrap.com/>

5.2 Aplicación

Como se mencionó anteriormente, el código fuente que se tomó para la realización de ésta aplicación web fue de Ross Kirsling⁸² y se procedió a editar para adaptarlo a las necesidades propias del programa de grafos. En primer lugar se explicará lo elaborado con las dos secciones de edición libre de grafos (dirigido y no dirigido) y después se hablará de cómo se elaboraron los 30 ejercicios y de dónde se obtuvieron (bibliografía).

5.2.1 Desarrollo de la sección “Edición Libre de Grafo Dirigido”

Esta fue la sección que se desarrolló en primer lugar, la razón de ésta decisión se basó en que es el área a la que menos cambios se necesitaban realizar al código fuente y por ello permitió comprender mejor las capacidades del código, funciones, entender asignación de variables y elementos, además de las áreas en las que se podían hacer cambios para adaptarlos a las necesidades del proyecto.

En cuanto a la vista hacia el usuario, al abrir esta sección se le muestra una lista con las instrucciones para operar la animación del grafo, se le detallan qué teclas o botones del mouse reaccionan a eventos dentro del propio software:

- a) Se da click en un área libre para agregar un nuevo nodo.
- b) Para unir 2 nodos: Click en el primer nodo y arrastrar el mouse hasta el segundo, soltar para crear el enlace entre ellos.
- c) Para mover de lugar el grafo completo: Oprimir "CTRL + Click en algún nodo" y arrastrar el mouse hasta la posición deseada.
- d) Cuando se selecciona un nodo, al apretar "R" en el teclado, se cambia el grueso de su contorno.
- e) Cuando se selecciona un enlace entre 2 nodos, al apretar "L" (cambia la dirección de la flecha a la izquierda), "R" (cambia la dirección a la derecha) y "B" (crea una flecha con ambas direcciones).
- f) Para borrar: Elegir el nodo o enlace que se desea eliminar y oprimir "Supr" o "Delete" en el teclado.

82 Kirsling, R. (2017). “Directed Graph Editor”. Disponible en: <http://bl.ocks.org/rkirsling/5001347>



Edición Libre De Grafo Dirigido

Instrucciones:

- Dar click en un área libre para agregar un nuevo nodo.
- Para unir 2 nodos: Click en el primer nodo y arrastrar el mouse hasta el segundo, soltar para crear el enlace entre ellos.
- Para mover de lugar el grafo completo: Oprimir "CTRL + Click en algún nodo" y arrastrar el mouse hasta la posición deseada.
- Cuando se selecciona un nodo, al apretar "R" en el teclado, se cambia el grueso de su contorno.
- Cuando se selecciona un enlace entre 2 nodos, al apretar "L" (cambia la dirección de la flecha a la izquierda), "R" (cambia la dirección a la derecha) y "B" (crea una flecha con ambas direcciones).
- Para borrar: Elegir el nodo o enlace que se desea eliminar y oprimir "Supr" o "Delete" en el teclado.

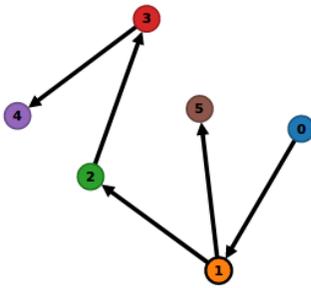


Figura 38. Sección "Edición de Grafo Dirigido" dentro de la aplicación web.

La intención es que el usuario se familiarice rápidamente con las instrucciones y domine la edición de grafos para que pueda practicar y elaborarlos según sus necesidades o prácticas, ya sea porque desea representar grafos que tiene en sus notas o bien expresar algún grafo que el profesor haya expuesto en clase.

5.2.2. Desarrollo de la sección “Edición Libre de Grafo No Dirigido”

Para elaborar esta sección de la aplicación web, se comenzaron a realizar modificaciones al código fuente después de haberlo analizado. Principalmente se necesitaba eliminar las flechas que denotaban la dirección del enlace entre grafo y grafo (y que por consiguiente lo convierten en grafo dirigido), para así poder hablar de grafos no dirigidos.

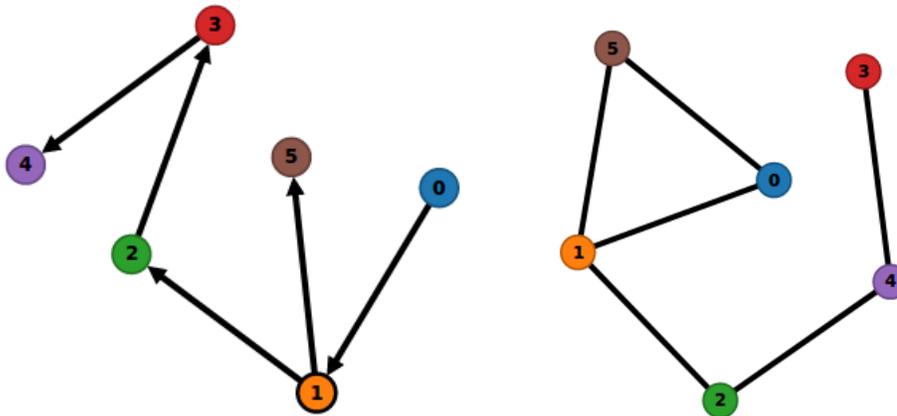


Figura 39. Del lado izquierdo se muestran las flechas que se necesitaban eliminar para poder conseguir mostrar un grafo no dirigido. Del lado derecho se muestra el cambio realizado.

Ahora bien, para poder realizar dicho cambio, fue necesario modificar algunas líneas en el código, estas son:

Grafo inicial:

Cuando se declara el grafo inicial, se deben indicar cuántos nodos y cuántos enlaces existirán entre los mismos.

```

var nodes = [
  {id: 0, reflexive: false},
  {id: 1, reflexive: false },
  {id: 2, reflexive: false}
],
lastNodeId = 2,
links = [
  {source: nodes[0], target: nodes[1], left: false, right: false },
  {source: nodes[1], target: nodes[2], left: false, right: false }
];

```

Figura 40. Declaración de los nodos y enlaces que contiene el grafo inicial

La variable “nodes” es la encargada de guardar cuántos nodos existirán en el grafo del inicio, mismos que son enumerados con su respectivo “id” para ir llevando un orden, se parte desde el 0; el campo “reflexive” hace mención a si se desea que el nodo se vea más grueso de lo normal, en este caso se marca “false” pues no se necesita hacer dicha acción.

Por otro lado, “lastNodeId” es el que indica al programa cuál es el último nodo generado (en este caso el 2), esto con la finalidad de que cuando se creé uno nuevo, se sepa que el siguiente en aparecer deberá ser el 3, después el 4, luego 5, etc.

Por su parte, “link” hace mención a los enlaces existentes, aquí se debe especificar el “source” (es decir, donde inicia la unión) y “target” que es donde termina la misma, por ejemplo si se requiere que haya un enlace entre el nodo 0 y el nodo 1, entonces en “source” debe aparecer el número 0 y en “target” el número 1. Los campos con “left” y “right” son donde deberíamos marcar “true” si se desea que la línea tenga forma de flecha, pero como en este caso no se desea así entonces se escribe “false”

Creación de nuevos nodos y enlaces:

Pero para poder indicar que los siguientes enlaces y nodos tampoco deben llevar una flecha, se deben modificar los siguientes aspectos:

<pre> svg.append('svg:defs').append('svg:marker') .attr('id', 'end-arrow') .attr('viewBox', '0 -5 10 10') .attr('refX', 6) .attr('markerWidth', 3) .attr('markerHeight', 3) .attr('orient', 'auto') .append('svg:path') .attr('d', 'M0,-5L10,0L0,5') .attr('fill', '#000'); svg.append('svg:defs').append('svg:marker') .attr('id', 'start-arrow') .attr('viewBox', '0 -5 10 10') .attr('refX', 4) .attr('markerWidth', 3) .attr('markerHeight', 3) .attr('orient', 'auto') .append('svg:path') .attr('d', 'M10,-5L0,0L10,5') .attr('fill', '#000'); </pre>	<pre> svg.append('svg:defs').append('svg:marker') .attr('id', 'end-arrow') .attr('viewBox', '0 -5 0 0') .attr('refX', 6) .attr('markerWidth', 3) .attr('markerHeight', 3) .attr('orient', 'auto') .append('svg:path') .attr('d', 'M0,-5L10,0L0,5') .attr('fill', '#000'); svg.append('svg:defs').append('svg:marker') .attr('id', 'start-arrow') .attr('viewBox', '0 -5 0 0') .attr('refX', 4) .attr('markerWidth', 3) .attr('markerHeight', 3) .attr('orient', 'auto') .append('svg:path') .attr('d', 'M10,-5L0,0L10,5') .attr('fill', '#000'); </pre>
--	--

Figura 41. Del lado izquierdo código grafo dirigido y del lado derecho código del grafo no dirigido

Como se explicó anteriormente, “svg” se utiliza para generar el espacio gráfico (lienzo) que se ocupará con D3 (en este caso, el grafo), y al usar “append” estamos indicando que vamos a agregar un nuevo elemento svg: “start-arrow” y “end-arrow” (que en este caso es la línea con la flecha), no obstante, para que aparezca únicamente como una línea en el grafo no dirigido, se debe sustituir el dato 10 por 0. Ver figura 41.

También es necesario modificar la variable “link”, específicamente en “direction”, pues si está en “true” se le está solicitando que aparezca en forma de flecha la dirección hacia donde apunta el enlace, pero cuando se coloca “false” se omite dicha acción.

<pre> var link; link = links.filter(function(l) { return (l.source === source && l.target === target); })[0]; if(link) { link[direction] = true; } else { link = {source: source, target: target, left: false, right: false}; link[direction] = true; links.push(link); } </pre>	<pre> var link; link = links.filter(function(l) { return (l.source === source && l.target === target); })[0]; if(link) { link[direction] = false; } else { link = {source: source, target: target, left: false, right: false}; link[direction] = false; links.push(link); } </pre>
---	---

Figura 42. Del lado izquierdo el código del grafo dirigido y del lado derecho el código del grafo no dirigido.

Paralelo a lo anterior, en el código existe una sección donde se declaran las acciones que desempeñará la aplicación referente a la tecla que se oprima en el teclado (dentro de un switch-case), además de apoyarse del uso de la sentencia “if”. En el caso del grafo no dirigido, es en ésta área donde se modifican las secciones porque la “punta” de la flecha no debe aparecer, así que se debe cambiar de “true” a “false” los campos pertinentes, para mayor claridad se subrayaron de rojo estas partes en la figura 43.

```

if(!selected_node && !selected_link) return;
switch(d3.event.keyCode) {
case 8: // Tecla "backspace"
case 46: // Tecla SUPR
    if(selected_node) {
        nodes.splice(nodes.indexOf(selected_node), 1);
        spliceLinksForNode(selected_node);
    } else if(selected_link) {
        links.splice(links.indexOf(selected_link), 1);
    }
    selected_link = null;
    selected_node = null;
    restart();
    break;
case 66: // Tecla B
    if(selected_link) {
        // Establece la flecha en ambas direcciones, derecha e izquierda
        selected_link.left = true;
        selected_link.right = true;
    }
    restart();
    break;
case 76: // Tecla L
    if(selected_link) {
        // Establece la flecha sólo hacia la izquierda
        selected_link.left = true;
        selected_link.right = false;
    }
    restart();
    break;
case 82: // Tecla R
    if(selected_node) {
        // Vuelve más grueso el nodo (si se elige un nodo)
        selected_node.reflexive = !selected_node.reflexive;
    } else if(selected_link) {
        // Establece la flecha sólo hacia la derecha (si se elige un enlace)
        selected_link.left = false;
        selected_link.right = true;
    }
    restart();
    break;
}

if(!selected_node && !selected_link) return;
switch(d3.event.keyCode) {
case 8: // Tecla "backspace"
case 46: // Tecla SUPR
    if(selected_node) {
        nodes.splice(nodes.indexOf(selected_node), 1);
        spliceLinksForNode(selected_node);
    } else if(selected_link) {
        links.splice(links.indexOf(selected_link), 1);
    }
    selected_link = null;
    selected_node = null;
    restart();
    break;
case 66: // Tecla B
    if(selected_link) {
        // Establece la flecha en ambas direcciones, derecha e izquierda
        selected_link.left = false;
        selected_link.right = false;
    }
    restart();
    break;
case 76: // Tecla L
    if(selected_link) {
        // Establece la flecha sólo hacia la izquierda
        selected_link.left = false;
        selected_link.right = false;
    }
    restart();
    break;
case 82: // Tecla R
    if(selected_node) {
        // Vuelve más grueso el nodo (si se elige un nodo)
        selected_node.reflexive = !selected_node.reflexive;
    } else if(selected_link) {
        // Establece la flecha sólo hacia la derecha (si se elige un enlace y no un nodo)
        selected_link.left = false;
        selected_link.right = false;
    }
    restart();
    break;
}

```

Figura 43. Del lado izquierdo el código grafo dirigido y del lado derecho el código del grafo no dirigido.

5.2.3. Desarrollo de la sección “Ejercicios”

Los ejercicios elaborados en la aplicación web están basados de la siguiente bibliografía:

- Jordan, C. (1996). "Introducción a la teoría de grafos y sus algoritmos". España: Universidad Politécnica de Valencia Servicio de Publicaciones.
- Vieites Rodríguez, A. (2014). "Teoría de grafos". Madrid: Paraninfo.

- Caicedo, A. (2010). Introducción a la Teoría de Grafos. Armenia. Ediciones Elizcom.

Además, para organizar mejor los ejercicios que están presentes en la aplicación web, se decidió organizarlos conforme el tipo de problemática que representan.

- Determinar matriz de adyacencia
- Ejercicios de subgrafos
- Elegir matriz del grafo

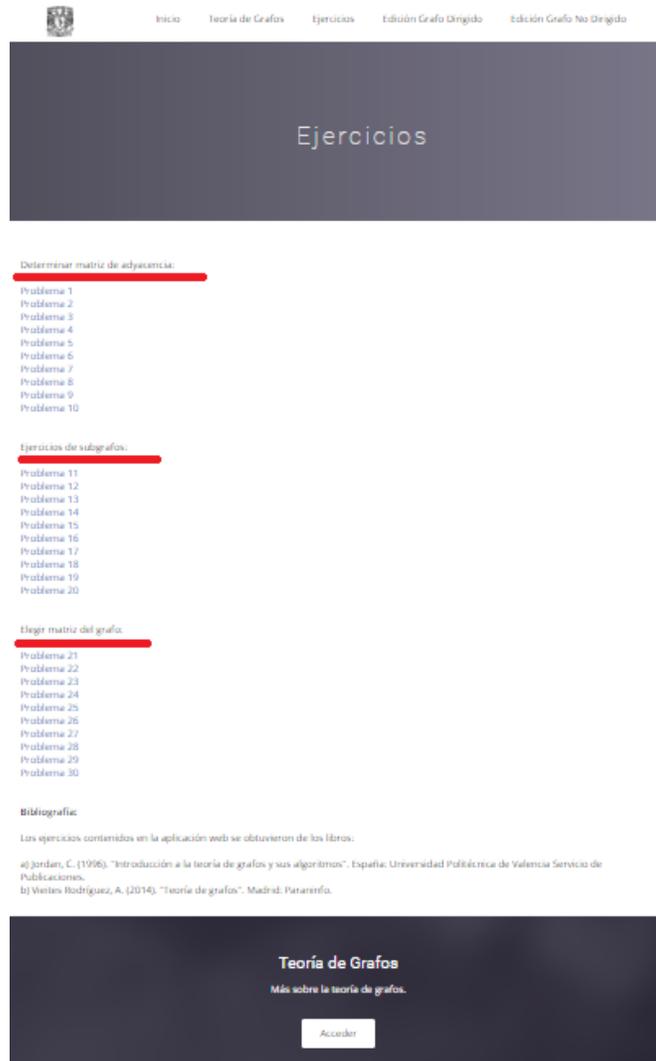


Figura 44. Representación de la organización de los ejercicios en la aplicación web interactiva.

a) *Determinar matriz de adyacencia:*

Básicamente, la pantalla que se le muestra al usuario es la siguiente:

Ejercicio 1:

Sea el grafo siguiente, encontrar su matriz de adyacencia.

	0	1	2	3	4
0	<input type="checkbox"/>				
1	<input type="checkbox"/>				
2	<input type="checkbox"/>				
3	<input type="checkbox"/>				
4	<input type="checkbox"/>				

Evaluar Matriz

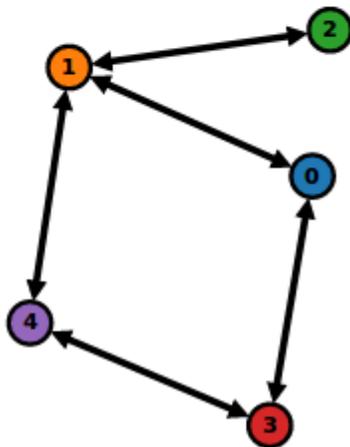


Figura 45. Ejercicio dentro de la aplicación web.

El usuario encontrará un ejercicio propuesto donde se le muestra un grafo (ya sea dirigido o no dirigido) y tendrá que definir la matriz de adyacencia que sea la correcta.

Para crear éste tipo de ejercicios (del 1 al 10), en primer lugar se generó un grafo utilizando D3 pero con la particularidad de evitar que pueda ser alterado, y para conseguirlo se modificó el código para que no se le agreguen más grafos o nodos. Lo anterior se consiguió de la siguiente forma:

```
65  /* var drag_line = svg.append('svg:path')
66     .attr('class', 'link dragline hidden')
67     .attr('d', 'M0,0L0,0');
68  Eliminado para que no pueda alterar el grafo el usuario
69  */
```

Figura 46. Código de la aplicación web.

La sección de código mostrada en la figura 46 es la encargada de dibujar la nueva línea que aparece cuando se intenta unir un grafo con otro. Para evitar que el usuario altere el grafo del ejercicio mostrado, se debe eliminar (o comentar) esta sección para que dicha línea no pueda generarse.

```
261  //Eliminado para evitar que el usuario agregue más nodos
262  /* var point = d3.mouse(this),
263     node = {id: ++lastNodeId, reflexive: false};
264     node.x = point[0];
265     node.y = point[1];
266     nodes.push(node);
267  */
```

Figura 47. Código de la aplicación web.

Por otro lado, también es necesario eliminar la sección de código de la figura 47, pues ésta es la encargada de crear un nuevo nodo cuando el usuario da click dentro del espacio de trabajo. Para que el ejercicio que encontrará el usuario no se vea afectado, debe ser suprimido este elemento (o comentado) y así no habrán alteraciones.

Además de lo anterior, se debe modificar el área correspondiente al “switch-case” en el que se define la función que tendrá la tecla SUPR, pues de dejarse definida el usuario podría alterar el grafo al eliminar nodos y, entonces, hacer que la matriz de adyacencia ya no corresponda a la del grafo. Ver figura 48.

```

320     if(!selected_node && !selected_link) return;
321     switch(d3.event.keyCode) {
322     case 8: // Tecla "backspace" (borrar)
323     case 46: // Tecla SUPR
324         /* if(selected_node) {
325             nodes.splice(nodes.indexOf(selected_node), 1);
326             spliceLinksForNode(selected_node);
327         } else if(selected_link) {
328             links.splice(links.indexOf(selected_link), 1);
329         }
330         selected_link = null;
331         selected_node = null;
332         restart(); */
333         break;

```

Figura 48. Código de la aplicación web.

Particularmente, en el caso de los grafos dirigidos, también se debe eliminar la opción en el teclado de que cuando se oprime R la flecha apunta hacia el lado derecho y cuando se tecléa L la misma apunta hacia el lado izquierdo. Ello se puede observar en la figura 49.

```

342     case 76: // Tecla L
343         /* if(selected_link) {
344             // set link_direction to left only
345             selected_link.left = true;
346             selected_link.right = false;
347         }
348         restart(); */
349         break;
350     case 82: // Tecla R
351         /* if(selected_node) {
352             // toggle node reflexivity
353             selected_node.reflexive = !selected_node.reflexive;
354         } else if(selected_link) {
355             // set link direction to right only
356             selected_link.left = false;
357             selected_link.right = true;
358         }
359         restart(); */
360         break;

```

Figura 49. Código de la aplicación web.

Por último, lo referente a la matriz de adyacencia, se decidió ocupar código HTML para general los “checkboxes” que representan a la matriz.

	0	1	2	3	4
0	<input type="checkbox"/>				
1	<input type="checkbox"/>				
2	<input type="checkbox"/>				
3	<input type="checkbox"/>				
4	<input type="checkbox"/>				

Figura 50. Matriz de adyacencia representada en la aplicación web.

Y en cuanto a la evaluación sobre si la respuesta del usuario es la correcta, se utilizó JavaScript para generar las validaciones correspondientes.

```

10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
function capturar()
{
    // Obtenemos el valor por el id 0
    var porId00=document.getElementById("a00").checked;
    var porId01=document.getElementById("a01").checked;
    var porId02=document.getElementById("a02").checked;
    var porId03=document.getElementById("a03").checked;
    var porId04=document.getElementById("a04").checked;

    // Obtenemos el valor por el id 1
    var porId10=document.getElementById("a10").checked;
    var porId11=document.getElementById("a11").checked;
    var porId12=document.getElementById("a12").checked;
    var porId13=document.getElementById("a13").checked;
    var porId14=document.getElementById("a14").checked;

    // Obtenemos el valor por el id 2
    var porId20=document.getElementById("a20").checked;
    var porId21=document.getElementById("a21").checked;
    var porId22=document.getElementById("a22").checked;
    var porId23=document.getElementById("a23").checked;
    var porId24=document.getElementById("a24").checked;

    // Obtenemos el valor por el id 3
    var porId30=document.getElementById("a30").checked;
    var porId31=document.getElementById("a31").checked;
    var porId32=document.getElementById("a32").checked;
    var porId33=document.getElementById("a33").checked;
    var porId34=document.getElementById("a34").checked;

    // Obtenemos el valor por el id 4
    var porId40=document.getElementById("a40").checked;
    var porId41=document.getElementById("a41").checked;
    var porId42=document.getElementById("a42").checked;
    var porId43=document.getElementById("a43").checked;
    var porId44=document.getElementById("a44").checked;
}

```

Figura 51. Declaración de los checkboxes en Javascript para conocer cuándo están seleccionados y cuando no.

En la figura 51, se expone cómo se tiene que declarar todos y cada uno de los checkboxes declarados en HTML dentro de variables para que Javascript los reconozca. Utilizando “.checked” se le hará saber al programa si está seleccionado o no, lo anterior es para que el condicional de “respuesta correcta” o “respuesta incorrecta” pueda entrar en función.

```
58
59     if (porId01 == true && porId03 == true && porId10 == true && porId12 == true && porId14 == true && porId21 == true && porId30 ==
60         true && porId34 == true && porId41 == true && porId43 == true) {
61         alertify.alert('¡Correcto!');
62     }
63     else {
64         alertify.error('Error');
65     }
66
67 }
68 </script>
69
```

Figura 52. Condicional de respuesta correcta o incorrecta.

Dado que cada checkbox de la matriz de adyacencia tiene una variable asignada en javascript, podemos conocer cuando está seleccionada (true) o cuando no lo está (false) alguna casilla; dependiendo del tipo de ejercicio habrá alguna combinación que sea la correcta y se le informará al usuario mediante un mensaje.

Respecto al mensaje que informa si la respuesta del alumno fue correcta o incorrecta, se utilizó AlertifyJs (como se mencionó anteriormente) para generar un cuadro de dialogo atractivo para el usuario.

Si la respuesta es correcta, aparecerá un mensaje en el centro de la ventana indicando que la información ingresada es acertada. Ver figura 53.

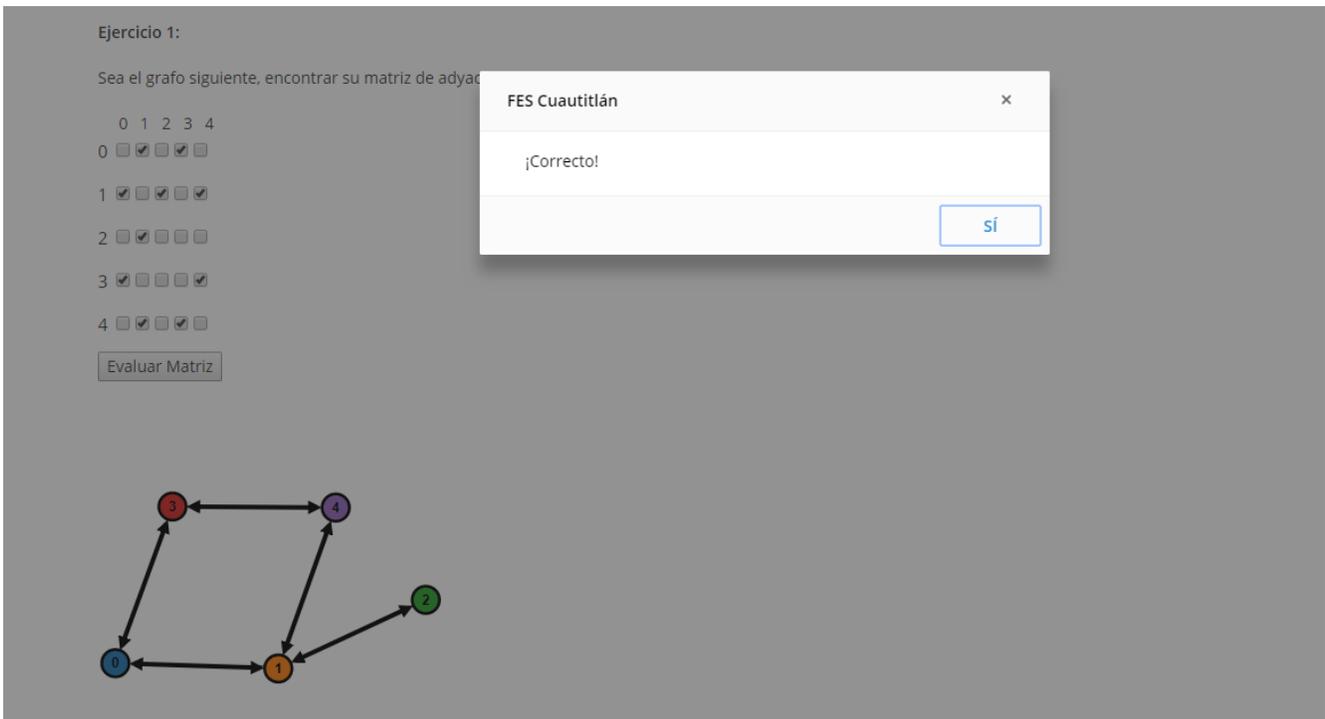


Figura 53. Mensaje de respuesta correcta.

Y si el usuario comete un error, aparecerá un mensaje en el extremo inferior derecho de la ventana. Ver figura 54.

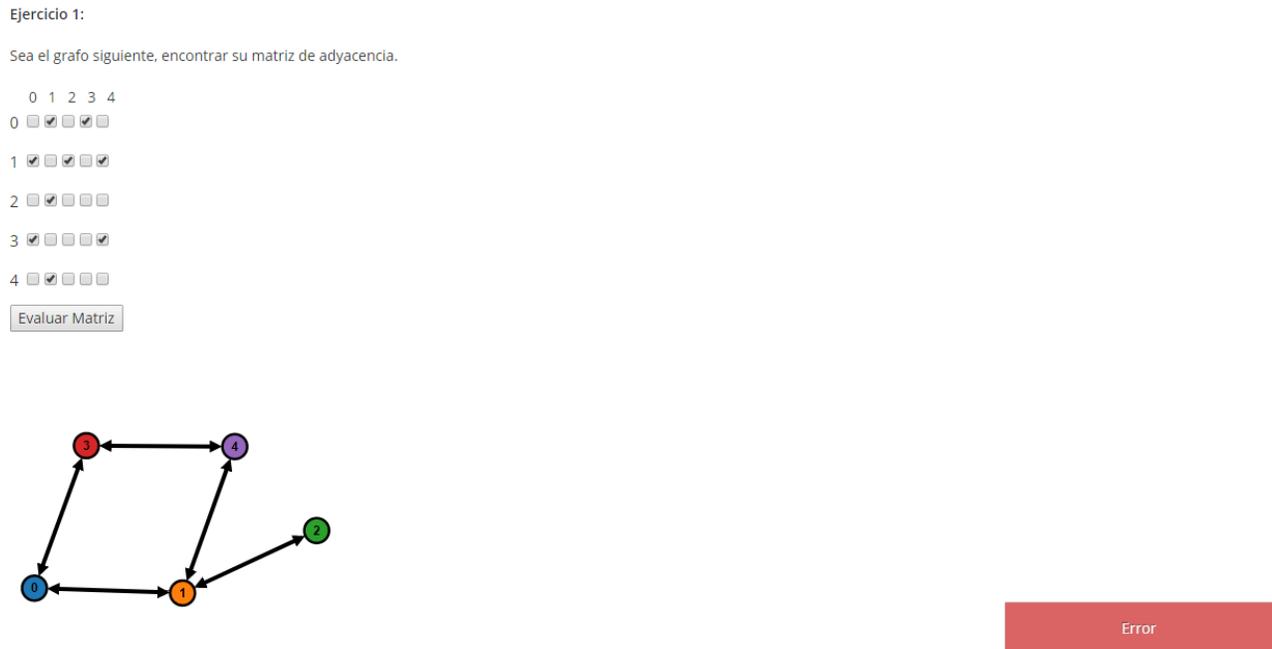


Figura 54. Mensaje de respuesta errónea.

b) Ejercicios de subgrafos:

En éste tipo de ejercicios (del 11 al 20), la pantalla que visualizará el usuario será la siguiente:

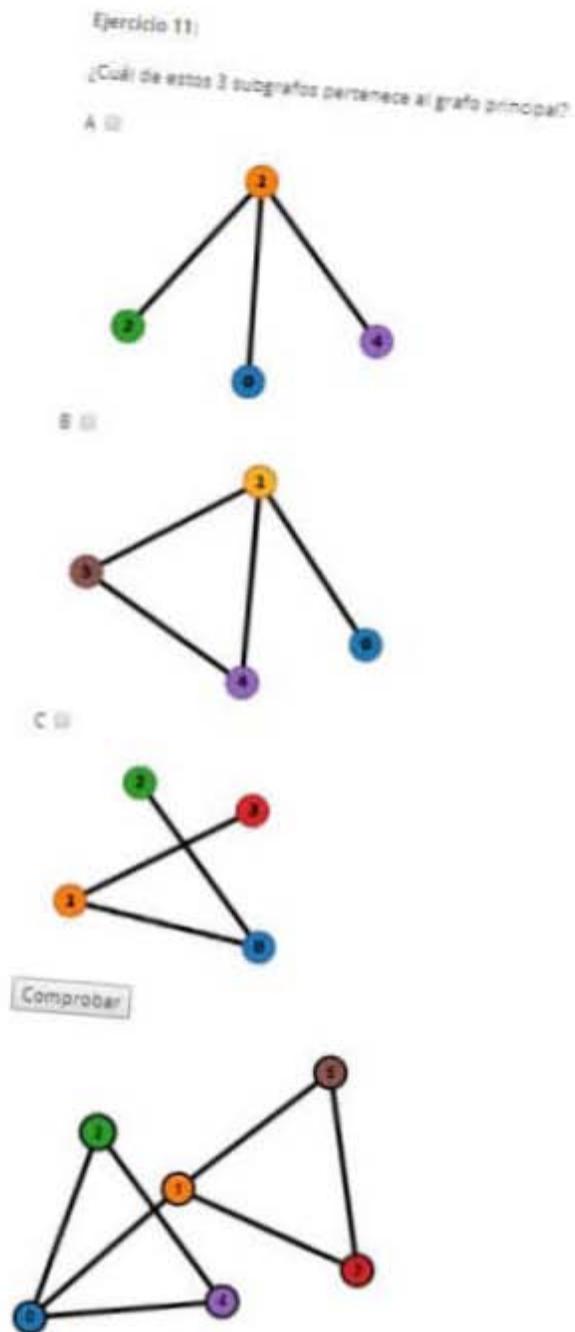


Figura 55. Ejemplo de un ejercicio de subgrafos.

En esta variedad de ejercicio, se le expone al alumno un grafo y él tendrá que determinar cuál de los 3 subgrafos es el perteneciente al grafo padre, sólo una respuesta es correcta y al igual que en el ejercicio anterior se define en JavaScript una variable por cada checkbox presente (en este caso 3, por las opciones A, B, C) y dependiendo de la elección del alumno se mostrará un mensaje de “respuesta correcta” o de “error”.

Destacar que lo interesante en éste ejercicio está en que, como se explicó anteriormente, el grafo padre también está protegido ante el accionar del alumno y no podría modificarlo por más que intente borrar o añadir nodos con el mouse y teclado, dando así la certeza de que el resultado no puede verse afectado.

Por el lado de Javascript, el condicional “if” sólo admite una selección como respuesta correcta, esto se asegura integrando operadores lógicos en la declaración.

```
22 // Obtenemos el valor por el id A
23 var porIdA=document.getElementById("a").checked;
24
25
26 // Obtenemos el valor por el id B
27 var porIdB=document.getElementById("b").checked;
28
29
30 // Obtenemos el valor por el id C
31 var porIdC=document.getElementById("c").checked;
32
33
34 if (porIdA == false && porIdB == false && porIdC == true) {
35     //alert('Respuesta Correcta');
36     alertify.alert('¡Correcto!');
37
38 }
39
40 else {
41     alertify.error('Error');
42 }
43
44 }
45 </script>
```

Figura 56. Condicional “if” y operadores lógicos para asegurar una correcta evaluación

Ejercicio 1:

Sea el grafo siguiente, encontrar su matriz de adyacencia

0	1	2	3	4
0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
2	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
4	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

FES Cuautitlán x

¡Correcto!

Figura 57. Mensaje de respuesta correcta en ejercicios de subgrafos.

Ejercicio 11:

¿Cuál de estos 3 subgrafos pertenece al grafo principal?

A

B

C

Error

Figura 58. Mensaje de error en ejercicios de subgrafos.

b) Elegir matriz del grafo:

Estos ejercicios (del 21 al 30) presentan al usuario otra variedad de problemas a resolver.

Ejercicio 21:
Sea el grafo siguiente, encontrar su matriz de adyacencia.

A)

	0	1	2	3	4	5	6
0	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

B)

	0	1	2	3	4	5	6
0	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

C)

	0	1	2	3	4	5	6
0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

```
graph TD; 0((0)) --> 1((1)); 0((0)) --> 2((2)); 1((1)) --> 2((2)); 1((1)) --> 3((3)); 2((2)) --> 3((3)); 3((3)) --> 4((4)); 3((3)) --> 5((5)); 4((4)) --> 5((5)); 5((5)) --> 6((6)); 6((6)) --> 4((4));
```

Figura 59. Ejemplo de un ejercicio de elección de matriz de adyacencia.

En esta sección, el usuario debe elegir cuál de las 3 matrices de adyacencia presentadas es la correcta, sólo una opción es válida y recibirá mensaje de “respuesta correcta” o “error” según sea su elección.

Al igual que los ejercicios anteriores, se creó código Javascript para identificar cada checkbox presente y un condicional que evalúa la respuesta del usuario. Nuevamente se garantiza que el alumno no puede alterar el grafo generado y que también sólo una opción (A, B o C) es correcta gracias al uso de los respectivos operadores lógicos.

Pero además de lo anterior, en ésta modalidad de ejercicio se decidió interactuar aún más con el usuario. Cuando éste elija el inciso que considera resuelve el problema, aparecerá una ventana dentro de la aplicación donde tendrá que confirmar la selección de su respuesta, pudiendo continuar o cancelar para volver a reflexionar el problema.

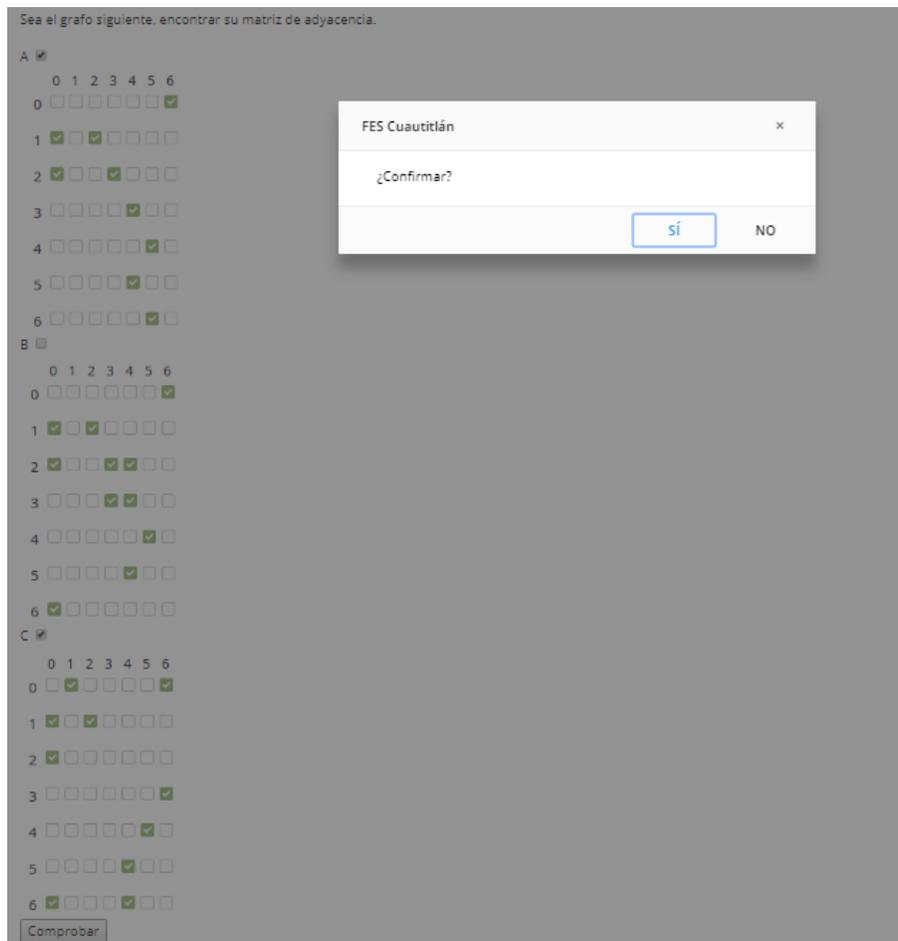


Figura 60. Ejemplo del cuadro de diálogo de confirmación de respuesta.

Si elige confirmar su respuesta, el programa evaluará la selección y dependiendo de ello informará al alumno si acertó o si se equivocó.

Sea el grafo siguiente, encontrar su matriz de adyacencia.

A

	0	1	2	3	4	5	6
0	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

B

	0	1	2	3	4	5	6
0	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

C

	0	1	2	3	4	5	6
0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input checked="" type="checkbox"/>					
4	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>				
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

¡Correcto!

Figura 61. Ejemplo del mensaje de respuesta correcta.

Sea el grafo siguiente, encontrar su matriz de adyacencia.

A

	0	1	2	3	4	5	6
0	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

B

	0	1	2	3	4	5	6
0	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

C

	0	1	2	3	4	5	6
0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input checked="" type="checkbox"/>					
4	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>				
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

¡Error!

Figura 62. Ejemplo del mensaje de error.

Si decide cancelar y no confirmar aún su respuesta, el mensaje que aparecerá en pantalla es el siguiente.

Sea el grafo siguiente, encontrar su matriz de adyacencia.

A

	0	1	2	3	4	5	6
0	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

B

	0	1	2	3	4	5	6
0	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

C

	0	1	2	3	4	5	6
0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
4	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Cancelado

Figura 63. Ejemplo del mensaje de cancelación en la confirmación.

Cuando la aplicación entra a la evaluación sobre la respuesta del usuario, en el código de Javascript se define dentro del condicional qué hacer dependiendo de la situación, ya sea lanzar un cuadro de diálogo para que el usuario reafirme su respuesta, el mensaje de “respuesta correcta”, el mensaje de “error” o el mensaje de “decisión cancelada”.

```

24 // Obtenemos el valor por el id A
25 var porIdA=document.getElementById("a").checked;
26
27
28 // Obtenemos el valor por el id B
29 var porIdB=document.getElementById("b").checked;
30
31
32 // Obtenemos el valor por el id C
33 var porIdC=document.getElementById("c").checked;
34
35
36 if (porIdA == true && porIdB == false && porIdC == false) {
37
38     //alert('Respuesta Correcta');
39     //alertify.alert('¡Correcto!');
40
41     alertify.confirm("¿Confirmar?",
42     function(){
43     alertify.success('¡Correcto!');
44     },
45     function(){
46     alertify.error('Cancelado');
47     });
48
49 }
50 else {
51
52     //alertify.error('Error');
53     alertify.confirm("¿Confirmar?",
54     function(){
55     alertify.error('¡Error!');
56     },
57     function(){
58     alertify.error('Cancelado');
59     });
60 }

```

Figura 64. Código que define qué hacer cuando el usuario confirma o cancela su respuesta.

La intención de esto es mostrar las capacidad a futuro que tiene la aplicación web, podría establecerse que ciertos ejercicios sólo tengan una oportunidad para responderse, si se acierta aparecerá el mensaje de “respuesta correcta” pero si se equivoca, además de recibir un mensaje de “error”, podría definirse que ya no puede volver a contestarse la pregunta, quedando registrado que ya se hizo un intento y se falló (simulando un tipo examen).

Los 30 ejercicios de la aplicación web (en sus distintas variedades) abarcan los temas de la Teoría de Grafos que se especificaron en los objetivos de la tesis, es decir:

- Grafos Simples Dirigidos
- Grafos Simples No Dirigidos
- Subgrafos
- Matrices de adyacencia

El software podría enriquecerse con más información de la teoría de grafos, sin embargo, considerando el poco tiempo que puede asignar el profesor de la materia “Matemáticas Computacionales” al tema, se encontró apropiado manejar sólo ésta información para que los alumnos conozcan el tópico y no se quede a la deriva en su formación académica.

Conclusiones

La realización de la “Aplicación web interactiva para el estudio y la edición de grafos” se desarrolló de manera correcta al momento de perseguir los objetivos trazados, la utilización de las tecnologías web sin duda significan una oportunidad inmejorable para transmitir el software a una mayor cantidad de usuarios sin tener las limitantes propias de la distribución física. No hay que dejar de mencionar que se puede ocupar cualquier computadora (y desde cualquier lugar) pues el programa no se instala en ningún equipo concreto del alumno, siempre estará disponible vía internet gracias a la computación en la nube.

La Facultad de Estudios Superiores Cuautitlán, UNAM, ahora cuenta con una herramienta web dirigida a sus alumnos (de licenciatura en informática, concretamente) que les brinda apoyo durante la enseñanza de la materia “Matemáticas Computacionales” (del cuarto semestre) en el tema de “Teoría de Grafos”. La aplicación les permite visualizar de manera didáctica los grafos y otorga al profesor una alternativa más para impartir la materia (y ocupar los equipos de cómputo, como sugiere el plan de estudios), además de poner en práctica los conocimientos obtenidos en el salón de clases. Ahora se tiene a su alcance un espacio virtual para continuar desarrollando lo aprendido en clase y, ¿por qué no?, incrementar lo estudiado en la materia.

Se debe mencionar que al elaborar la aplicación, y aprender a usar un lenguaje de programación como lo es JavaScript, paralelo a conocer y trabajar con una librería tan potente como D3.js, significó un gran apoyo en la formación profesional, además de aplicar los conocimientos de la teoría de grafos. Es necesario utilizar las tecnologías más recientes y competitivas presentes en el mercado, y el ocupar D3.js implica estar a la vanguardia en cuanto al desarrollo web. No se puede dejar de destacar el papel tan importante que desempeña el asesor al motivar y dirigir el proyecto para alcanzar los objetivos.

Las aplicaciones que se pueden ejecutar actualmente vía web son prácticamente incontables, y el poder tomar como base una idea que ya se había desarrollado (software de grafos) pero ahora retransmitirlo usando internet y sus bondades, resulta favorable para acercar a más usuarios a programas que pueden representarles una solución efectiva a sus problemáticas.

Durante la elaboración de la aplicación web se presentaron retos y desafíos que tuvieron que analizarse, explorarse y solucionarse adecuadamente para mantener una aplicación estable y competitiva en todo momento. Por ejemplo, surgió la interrogante de ¿cómo hacer para que los grafos generados por D3.js no fueran alterados por los alumnos? En ese caso, lo más sensato fue modificar el código para evitar que algunos comandos definidos en el teclado (funciones para la tecla L, R, B, etc) tuvieran una acción, logrando así que aunque los usuarios oprimieran teclas o acciones con el mouse sobre el grafo, éste no se vea afectado, garantizándole al profesor que los ejercicios plasmados serán respetados y eventualmente resueltos sin ninguna modificación inoportuna.

Incluso a nivel personal, el utilizar software libre y conocer a fondo su filosofía es de mucha ayuda para la formación laboral y profesional, específicamente en las actividades que se desenvuelven cotidianamente en el trabajo actual. Por diversas políticas de la empresa donde se labora cotidianamente, no se puede hacer uso de lenguajes de programación robustos como PHP, sin embargo, en JavaScript hay una opción muy competitiva, fiable y oportuna para continuar desarrollando la programación, esto es porque el lenguaje no requiere de instalaciones complejas, todo se ejecuta desde el lado del cliente. De hecho, no es de sorprender el poder asegurar que tarde o temprano se podrá emplear, en algún proyecto laboral, el uso total de D3.js pues la experiencia obtenida se va a aprovechar para adaptar aplicaciones ya existentes, de la compañía, a las nuevas tecnologías web.

Referencias

1. Pressman, R. (2009). Web Engineering: A Practitioner's Approach. Estados Unidos: McGraw-Hill.
2. Maclean, M. (2013). D3 Tips and Tricks: Because Life's too Short to Learn All the Things. Estados Unidos: Leanpub.
3. Stefanov, S. JavaScript Patterns. Estados Unidos: O'Reilly & Yahoo! Press.
4. Murray, S. Interactive Data Visualization for the Web. Estados Unidos: O'Reilly.
5. Murray, S.. An introduction to designing with D3: Interactive Data Visualization. Estados Unidos: O'Reilly.
6. Dewar, M. Getting Started with D3. Estados Unidos: O'Reilly.
7. Jordan, C. (1996). Introducción a la teoría de grafos y sus algoritmos. España: Universidad Politécnica de Valencia Servicio de Publicaciones.
8. Vieites Rodríguez, A. (2014). Teoría de grafos. Madrid: Paraninfo.
9. Caicedo, A. (2010). Introducción a la Teoría de Grafos. Armenia. Ediciones Elizcom.
10. Gutiérrez, E. (2009). JavaScript: conceptos básicos y avanzados. España: ENI ediciones.
11. Suehring, S. (2008). JavaScript: paso a paso. España: Anaya Multimedia.
12. McCracken, S. (2012). Curso de programación web : con HTML5, CSS, JavaScript, PHP 5/6 y MySQL. España: Inforbook's.
13. Yworks. (2017). "yEd – Graph Editor". Consultado el 6 marzo de 2017. Disponible en: <https://www.yworks.com/products/yed>
14. Rodríguez, A. (2013). "Grafos - software para la construcción, edición y análisis de grafos". Consultado el 6 de marzo de 2017. Disponible en: <http://arodrigu.webs.upv.es/grafos/doku.php?id=software>
15. Bostock, M. (2015). "D3.js - Data-Driven Documents". Consultado el 6 de marzo de 2017. Disponible en: <https://d3js.org/>
16. Cancela, H. (2009). "Curso de Bioinformática". Consultado el 8 de marzo de 2017. Disponible en: <https://www.fing.edu.uy/inco/grupos/bioinf/bioinfo1/index.html>
17. Cancela, H. (2009). "Material teórico sobre grafos". Consultado el 8 de marzo de 2017. Disponible en: <https://www.fing.edu.uy/inco/grupos/bioinf/bioinfo1/teorico/grafos.pdf>
18. Universidad de Pamplona. (2012). "Teoría de grafos". Consultado el 8 de marzo de 2017. Disponible en: http://www.unipamplona.edu.co/unipamplona/portaIIG/home_23/recursos/general/11072012/grafos3.pdf
19. Arteaga, L. (2001) "¿Qué es el Software Libre". Consultado el 16 de marzo de 2017. Disponible en: <https://www.gnu.org/philosophy/free-sw.es.html>
20. Garzón, E. (2012). "Biografía de Richard Stallman". Consultado el 22 de marzo de 2017. Disponible en: http://e-logicasoftware.com/tutoriales/curso-virtual-software-libre-alcaldia/1_introduccion/recursos/richardstallman.pdf
21. Garzón, E. (2012). "Biografía de Richard Stallman". Consultado el 22 de marzo de 2017. Disponible en: http://e-logicasoftware.com/tutoriales/curso-virtual-software-libre-alcaldia/1_introduccion/recursos/richardstallman.pdf
22. Free Software Foundation. (2016) "Licencias". Consultado el 23 de marzo de 2017. Disponible en: <https://www.gnu.org/licenses/licenses.es.html#GPL>

23. Yúbal, F. (2016). “Entrevista a Richard Stallman”. Consultado el 23 de marzo de 2017.
Disponible en: <https://www.genbeta.com/especial/el-usuario-merece-que-sea-obligatorio-que-se-respeta-su-libertad-entrevista-a-richard-stallman>
24. Free Software Foundation. (2016). “¿Qué es el copyleft?”. Consultado el 23 de marzo de 2017.
Disponible en: <https://www.gnu.org/licenses/copyleft.es.html>
25. Alegsa, L. (2016). “Definición de Tecnología”. Consultado el 24 de marzo de 2017. Disponible en: <http://www.alegsa.com.ar/Dic/tecnologia.php>
26. Gago, F. (). “¿Qué es la World Wide Web?”. Consultado el 24 de marzo de 2017. Disponible en: http://www3.uah.es/farmamol/Public/Curso_Internet/CERN/quees.html
27. Alegsa, L. (2016). “Definición de Navegador”. Consultado el 24 de marzo de 2017. Disponible en: <http://www.alegsa.com.ar/Dic/navegador.php>
28. Flores, J. (2016). “¿Qué es HTML”. Recuperado el 27 de marzo de 2017. Disponible en: <https://codigofacilito.com/articulos/que-es-html>
29. Mozilla Foundation (2017). “HTML”. Recuperado el 27 de marzo de 2017. Disponible en: <https://developer.mozilla.org/es/docs/Web/HTML>
30. Álvarez, M. (2001). “Qué es HTML”. Recuperado el 27 de marzo de 2017. Disponible en: <https://desarrolloweb.com/articulos/que-es-html.html>
31. Pimentel, V. (2009). “Las novedades de HTML5”. Recuperado el 27 de marzo de 2017.
Disponible en: <http://www.nobbot.com/personas/las-novedades-de-html5-i/>
32. Mozilla Foundation (2017). “Tecnología web para desarrolladores”. Recuperado el 27 de marzo de 2017. Disponible en: <https://developer.mozilla.org/es/docs/Web/>
33. Mozilla Foundation (2017). “Javascript”. Recuperado el 27 de marzo de 2017. Disponible en: <https://developer.mozilla.org/es/docs/Web/Javascript>
34. Álvarez, M. (2009). “Diferencias entre Java y JavaScript”. Recuperado el 27 de marzo de 2017.
Disponible en: <https://www.desarrolloweb.com/articulos/492.php>
35. Digital Learning. (2012). “¿Qué es AJAX?” Recuperado el 27 de marzo de 2017. Disponible en: <http://www.digitallearning.es/blog/que-es-ajax/>
36. Aguilar, A. (2015). “¿Qué es y cómo opera un ataque de Cross-Site Scripting?”. Recuperado el 27 de marzo de 2017. Disponible en: <http://www.seguridad.unam.mx/documento/?id=35>
37. Área Tecnología (2017). “Qué es un servidor y tipo de servidores”. Recuperado el 28 de marzo de 2017. Disponible en: <http://www.areatecnologia.com/informatica/servidor-y-tipos.html>
38. Autor Desconocido (2016). “¿Qué es el servicio FTP – File Transfer Protocol?”. Recuperado el 28 de marzo de 2017. Disponible en: <http://www.internetya.co/que-es-el-servicio-ftp-file-transfer-protocol/>
39. Aprigiliano, D. (2016). “¿Qué es un hosting o servidor web? ¿para qué sirve?”. Recuperado el 29 de marzo de 2017. Disponible en: <http://www.dosd.com.ar/que-es-un-hosting-o-servidor-web-para-que-sirve/>
40. Alegsa, L. (2016). “Definición de Aplicación Web”. Consultado el 29 de marzo de 2017.
Disponible en: http://www.alegsa.com.ar/Dic/aplicacion_web.php
41. Mocholí, A. (2015). “6 Tipos de desarrollo de aplicaciones web más relevantes”. Recuperado el 30 de marzo de 2017. Disponible en: <https://www.yeeply.com/blog/6-tipos-desarrollo-de-aplicaciones-web/>
42. Aznar, J.(2014). “Qué es SVG”. Recuperado el 5 de abril de 2017. Disponible en: <https://desarrolloweb.com/articulos/que-es-svg.html>
43. Rivera, C. (2013). “Un poco de acerca del mapeo de la información”. Consultado el 18 de abril de 2017. Disponible en <http://www.infotecarios.com/mapeo-de-la-informacion/>

44. Google Developers. (2014). “Minificar recursos (HTML, CSS y Javascript)”. Consultado el 20 de abril de 2017. Disponible en <https://developers.google.com/speed/docs/insights/MinifyResources?hl=es-419>
45. Fernández, R. (2014). “Patrones de diseño: qué son y por qué debes usarlos”. Consultado el 10 de junio de 2017. Disponible en <https://www.genbetadev.com/metodologias-de-programacion/patrones-de-diseno-que-son-y-por-que-debes-usarlos>
46. Garzías, J. (2014). “¿Qué tipos de patrones hay?”. Consultado el 10 de junio de 2017. Disponible en <http://www.javiergarzas.com/2014/08/tipos-patrones-software.html>
47. Flores, W. (2015). “Patrón Modular en JavaScript”. Consultado el 10 de junio de 2017. Disponible en <https://frontendlabs.io/2643--patron-modular-en-javascript>
48. Bostock, M. (2017). “Bullet Charts”. Consultado el 21 de junio de 2017. Disponible en: <https://bl.ocks.org/mbostock/4061961>
49. Bostock, M. (2017). “Cluster Dendrogram”. Consultado el 21 de junio de 2017. Disponible en: <https://bl.ocks.org/mbostock/4063570>
50. Bostock, M. (2017). “Force-Directed Graph”. Consultado el 21 de junio de 2017. Disponible en: <https://bl.ocks.org/mbostock/4062045>
51. Bostock, M. (2017). “Circle Packing”. Consultado el 21 de junio de 2017. Disponible en: <https://bl.ocks.org/mbostock/4063530>
52. Bostock, M. (2017). “Radial Tidy Tree”. Consultado el 21 de junio de 2017. Disponible en: <https://bl.ocks.org/mbostock/4063550>
53. Bostock, M. (2017). “OMG! Particles”. Consultado el 21 de junio de 2017. Disponible en: <https://bl.ocks.org/mbostock/1062544>
54. Zubkov (2013). “World Bank Global Development Sprint”. Consultado el 22 de junio de 2017. Disponible en: <http://d3.artzub.com/wbca/>
55. Wood, A. (2013). “Health and Wealth of Nations”. Consultado el 22 de junio de 2017. Disponible en: <http://cesiumjs.org/demos/d3.html>
56. Smirnov, N. (2017). “Game Shooter”. Consultado el 23 de junio de 2017. Disponible en: <http://d3shooter.herokuapp.com/>
57. Usuario “fhernand” de Github. (2016). “Visualization of passes in a soccer game”, Consultado el 23 de junio de 2017. Disponible en: <https://bl.ocks.org/fhernand/9a9f93f2a6b0e83a9294>
58. Kirsling, R. (2017). “Directed Graph Editor”. Consultado el 30 de junio de 2017. Disponible en: <http://bl.ocks.org/rkirsling/5001347>
59. Otto, M. (2017). “Bootstrap. The most popular HTML, CSS, and JS framework in the world”. Consultado el 30 de junio de 2017. Disponible en: <https://v4-alpha.getbootstrap.com/>