Modelado de una tienda virtual mediante Diagramas de Transición entre Interfaces de Usuario y Diagramas de Secuencia Detallados: Un caso de éxito

Modelling a virtual store with User Interface Transition Diagrams and Detailed Sequence Diagrams: A success case

Luis Gerardo Ramírez-Viveros, María del Carmen Gómez-Fuentes*, Jorge Cervantes-Ojeda
Departamento de Matemáticas Aplicadas y Sistemas
Universidad Autónoma Metropolitana Unidad Cuajimalpa
Avenida Vasco de Quiroga 4871, Col. Santa Fe Cuajimalpa.
Delegación Cuajimalpa de Morelos, Ciudad de México, México, C.P. 05348

* Correo-e: mgomez@cua.uam.mx

PALABRAS CLAVE:
modelado de software, requerimientos, diseño detallado, DTIU con DSD.

RESUMEN
Aplicamos una metodología que combina los Diagramas de Transiciones entre Interfaces de Usuario (DTIU) y los Diagramas de Secuencia Detallados (DSD) para construir una tienda virtual de libros, revistas, música y películas. Los DTIU son diagramas que permiten modelar la navegación entre las interfaces de usuario de un sistema, mientras que los DSD son diagramas de secuencia construidos bajo un conjunto de reglas hechas específicamente para documentar el diseño detallado. Especificamos la navegación del usuario por el sistema mediante la combinación de los bosquejos de las Interfaces de Usuario (IU) y los DTIU. Los DSD se construyen haciendo referencia a los bosquejos de IU, a los DTIU y a los componentes del sistema. Los DTIU ayudan a relacionar los casos de uso que inician en una IU con los DSD del diseño detallado. El sistema de referencias entre la fase de requerimientos y el diseño detallado ayuda a verificar que el diseño cumpla con los requerimientos del sistema relacionados con la interacción con el usuario. Esta metodología también ayuda a detectar casos especiales y posibles fallas u omisiones en los requerimientos y el diseño, facilitando la construcción de sistemas robustos. Por otra parte, el esfuerzo que se dedica a la elaboración de los DSD durante el diseño detallado permite ahorrar tiempo de implementación, ya que, cuando el diseño detallado es suficientemente completo, la implementación consistirá en a la transcripción del diseño detallado al código y en la resolución de detalles muy particulares de la tecnología con la que se trabaje.

KEYWORDS:
software modeling, requirements, detailed design, UITD with DSD.

ABSTRACT
We apply a methodology that combines User Interface Transition Diagrams (UITD) and Detailed Sequence Diagrams (DSD) to build a virtual store (books, magazines, music and movies). The UITD are diagrams that allow modeling the navigation between the user interfaces of a system, while the DSDs are sequence diagrams built under a set of rules that are made specifically for detailed design. We specify the user navigation through the system by combining User Interface (UI) mockups and UITD. DSD are built referencing UI mockups, UITD and system components. The UITD helps matching the use cases that start in a UI with the DSD in the detailed design. The references between system requirements and detailed design helps verifying that the design meets the system user interaction requirements. This methodology also helps in detecting special cases and possible flaws or omissions in the requirements and the design, facilitating the construction of large robust systems. On the other hand, the DSD effort during the detailed design saves implementation time because, when detailed design is sufficiently complete, the implementation will consist of the transcription of detailed design decisions to code and the resolution of particular technology details.

Recibido: 16 de julio 2021 • **Aceptado:** 8 noviembre 2021 • **Publicado en línea:** 15 de febrero de 20

I. INTRODUCCIÓN

La especificación completa de las interacciones del sistema con el usuario y de las interfaces que el sistema tendrá es uno de los aspectos clave en la especificación de requerimientos. Por otra parte, la mayoría de los estudios empíricos muestran que la participación del usuario en la fase de especificación de requerimientos tiene un impacto positivo en el proyecto [1]. Sin embargo, Mendez et al. [2] reportan que las fallas de comunicación entre desarrolladores y clientes se perciben como el segundo problema más importante en la Ingeniería de Requerimientos y el principal problema que conduce al fracaso del proyecto. Los Diagramas de Transiciones entre Interfaces de Usuario (DTIU) [3], son una herramienta de modelado simplificada que sirve para describir el flujo entre las diferentes interfaces que se le presentan al usuario en un sistema de software (SW). Con los DTIU se pueden modelar sistemas de SW de forma clara e intuitiva para que los clientes (normalmente sin capacitación en lenguajes de modelado) puedan comprender los diagramas y participen en el proceso de extracción de los requerimientos.

En este trabajo mostramos a los desarrolladores de SW las ventajas potenciales de usar una metodología que combina los DTIU con los bosquejos de las Interfaces de Usuario (IU) y los Diagramas de Secuencia Detallados (DSD) [4] en la construcción de un sistema de SW. Los DSD agregan a los Diagramas de Secuencia UML, un conjunto de reglas de notación para el diseño detallado, y ayudan a los desarrolladores a garantizar que la implementación cumpla con los requerimientos del sistema y a reducir el esfuerzo en la implementación.

El caso de estudio que aquí presentamos, es el de un gestor de tienda en línea para la venta de libros, revistas, música y películas. Las partes ilustrativas que incluimos sirven para ejemplificar la especificación de requerimientos de navegación mediante DTIU y la elaboración del diseño detallado mediante DSD. Estos ejemplos ayudan a mostrar cómo es que el modelado con DTIU facilita la detección de algunas fallas en el diseño de la navegación del usuario en el sistema, por ejemplo, la identificación de: i) la falta de la opción regreso a la IU de origen en alguna IU de destino, ii) la omisión del caso en que el usuario no proporciona los datos correctamente, y iii) la omisión de los casos de error en el acceso a la base de datos, entre muchos otros. Otra de las ventajas del DTIU es que, al presentar de manera gráfica la interacción del usuario con el sistema, facilita la identificación de requerimientos faltantes.

El resto de este trabajo está organizado de la siguiente forma. En la Sección II se presentan los trabajos relacionados con la metodología que aquí se utiliza. En la Sección III se incluye una breve introducción a los DTIU. En la Sección IV se especifican de manera global los requerimientos de la tienda virtual y se incluye un ejemplo de los requerimientos de navegación mediante DTIU y bosquejos de IU. La Sección V contiene una breve introducción a los DSD. La Sección VI contiene el diseño de alto nivel del sistema. La Sección VII muestra ejemplos de diseño detallado basado en DTIU y DSD. En la Sección VIII presentamos los resultados obtenidos en la implementación del sistema. En la Sección IX se explica el alcance de la metodología propuesta. Finalmente, la Sección X contiene las conclusiones y el trabajo futuro.

II.TRABAJOS RELACIONADOS

Ricca et al. [5] demostraron la utilidad de los bosquejos de las IU (mockups) en la comprensión de los requerimientos funcionales. Los DTIU están diseñados para usarse junto con los mockups como se explica en [3][4][6]. Aquí presentamos ejemplos de la manera de trabajar relacionando los bosquejos de IU con el DTIU para especificar los requerimientos de navegación de una tienda en línea de libros, revistas, música y películas.

Si bien los DTIU se pueden elaborar con las herramientas gráficas existentes, por ejemplo, Pacestar UML diagrammer (http://www.pacestar.com/uml/) el cual es de paga, o herramientas de SW libre como lucidchart (www.lucidchart.com), etc., recientemente se puso a disposición gratuita en red el editor de DTIU [7] (http://148.206.168.145/EditorUITDEnglish/examples/indexF.html), con este editor se puede trabajar fácilmente con funcionalidades exclusivas de los DTIU, lo que permite que su elaboración sea más sencilla que con otras herramientas gráficas.

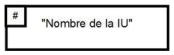
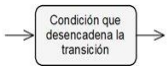
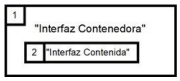
III. BREVE INTRODUCCIÓN A LOS DIGRAMAS DE TRANSICIONES ENTRE INTERFACES DE USUARIO (DTIU)

Un DTIU es un grafo dirigido con nodos que representan interfaces de usuario y aristas (flechas) que representan transiciones entre interfaces de usuario. Cada transición tiene una etiqueta. Las etiquetas de las flechas se componen de una acción de usuario y, cuando dos transiciones tienen la misma acción de usuario, la etiqueta también contiene información sobre las condiciones que deben cumplirse para desencadenar la transición. Las transiciones tienen una interfaz de usuario de origen y una de destino. También es posible que una interfaz sea el origen y el destino dada una acción del usuario.

Uno de los principios de la teoría de la Física de las Notaciones [8] es la *economía gráfica*, lo que significa que el número total de símbolos gráficos debe ser cognitivamente manejable y, según la ley de Miller, este número debe ser ≤ 9 [9]. En la Tabla 1 se muestran los símbolos gráficos del DTIU. Como se puede apreciar en

la Tabla 1, el DTIU cumple con la propiedad de economía gráfica, pues contiene únicamente 5 símbolos.

Tabla 1. Símbolos gráficos del DTIU

Símbolo	Significado	Notación
Interfaz de Usuario (IU)	Es la interfaz que se le presenta al usuario	
Transición	Contiene la acción del usuario que desencadena una transición. También puede contener el estado en el que se encuentra el sistema.	
IU anidadas	Las transiciones con origen en la interfaz *contenida* también están disponibles desde la interfaz *contenedora* y no viceversa.	
Bordes de IU con y sin negrita	El borde de IU en negrita significa que todas las posibles transiciones desde/hacia esta IU son visibles en el sub-diagrama que se está visualizando.	
Sub-diagrama abreviado	Las transiciones de/a un sub-diagrama se indican sin especificar las IU particulares dentro del bloque.	

Una Interfaz de Usuario (IU) contiene dos identificadores, un número y un nombre, lo que permite una fácil identificación de cada IU durante el diseño y etapas posteriores del proyecto.

Una *transición* entre IU se representa por medio de una flecha. La *transición* indica que el usuario ejecutó una acción en particular en una IU de origen, provocando que se despliegue la IU destino correspondiente. Como la *transición* se dispara mediante una acción de usuario, esta acción se documenta en la *etiqueta* de la flecha.

Es posible que una misma acción de usuario dentro de una IU desencadene diferentes transiciones en función de una condición dada. La manera de expresar lo anterior en un DTIU es agregando la condición a la etiqueta de cada transición después de la acción. Así, cada transición tendrá una etiqueta única, evitándose toda ambigüedad. El siguiente formato se usa para etiquetar una transición:

acción/<*condición*>

donde la condición solo es necesaria cuando dos o más transiciones, con la misma IU de origen, se activen con la misma acción.

Una característica destacada del DTIU es la capacidad de representar una interfaz de usuario que está *contenida* en otra interfaz de usuario. El significado de esto es que todas las transiciones con origen en la IU *contenida* también están disponibles en la IU *contenedora* (y no al revés). Los activadores de transición que están disponibles en la IU *contenida* son un subconjunto de los que están disponibles en la IU *contenedora*. Una interfaz de *contenedora* puede contener más de una IU y una IU puede estar contenida en más de una IU.

Poner el borde de las IU con o sin "negrita" permite fragmentar el DTIU en varios pedazos. Cuando una IU en particular tiene su borde en negrita se dice que está *completa.* Esto significa que todas las transiciones desde y hacia esa IU están documentadas en el fragmento de DTIU que se está visualizando. Cuando una IU no tiene sus bordes en negrita significa que en dicho fragmento no se incluyen todas las transiciones que conectan con esa IU. Para poder consultar todas las transiciones de esas IU habrá que visualizarlas en varios fragmentos o en algún fragmento donde sí aparezca la IU en negrita. En el DTIU completo, es decir, el que abarca a todo el sistema, los bordes de todas las IU están en negrita, ya que todas las transiciones posibles están documentadas.

El símbolo del sub-diagrama abreviado permite que, cuando el DTIU es grande, se pueda mostrar con varios sub-diagramas colapsados ocupando así menos espacio, de esta forma se puede tener una visión general del sistema en la que se omiten temporalmente algunos detalles.

IV. ESPECIFICACIÓN DE REQUERIMIENTOS

En esta sección se muestra una visión global de la tienda virtual con el objetivo de dar a conocer, en términos generales, lo que hace el sistema.

A. Visión global de los requerimientos del sistema

Propósito. - El gestor de tienda en línea, sirve para que tiendas que se dedican a la venta de libros, revistas, música y películas tengan una mayor presencia al contar con servicios en línea. El gestor permite dar a conocer los distintos productos con los que cuenta la tienda, vender productos y llevar a cabo la gestión de los productos.

Alcance. - El administrador de la tienda puede gestionar los productos, ponerlos a la venta y gestionar las ventas, mientras que los clientes pueden buscar, ver, comprar y calificar productos. Algunas de estas actividades están sujetas a que el cliente tenga una cuenta.

Funcionalidades del producto:

- Para el cliente:
  1) Ver productos
  2) Buscar productos
  3) Gestionar una *wish list* con los productos que le gustan
  4) Gestionar un *carrito* con los productos que va a comprar
  5) Comprar productos via paypal
  6) Ver información de compras
  7) Calificar y comentar productos
  8) Registrarse con cuenta y contraseña
  9) Cambiar contraseña

- Para el administrador:
  1) Gestión de cuentas del cliente
  2) Gestión de libros
  3) Gestión de revistas
  4) Gestión de música
  5) Gestión de películas
  6) Gestión de ventas

Clases y características de los usuarios. –

Administrador: tendrá acceso a todas las funcionalidades de gestión de los productos y de las ventas realizadas.

Cliente: tendrá una cuenta con la que podrá comprar y calificar productos. Para buscar y ver los productos no será necesario que tenga una cuenta.

Entorno operativo. – El sistema opera en la web, con una base de datos en el servidor.

B. Requerimientos especificos para ilustrar este caso de estudio

Requerimientos Específicos. – En este apartado se incluyen únicamente los requerimientos específicos que servirán para ilustrar nuestro caso de estudio. Estos requerimientos son los que corresponden al ***cliente***.

RE1.- El usuario podrá hacer las siguientes acciones sin tener que entrar al sistema (en logout)

RE1.1.- Búsqueda de productos por categorías.

RE1.2.- Visualización de los detalles de un producto seleccionado.

RE2.- El usuario podrá hacer las siguientes acciones cuando ingresa al sistema (en login)

RE2.1.- Búsqueda de productos por categorías.

RE2.2.- Visualización de los detalles de un producto seleccionado.

RE2.3.- Agregar productos a su *wish list*.

RE2.4.- Quitar productos de su *wish list.*

RE2.5.- Agregar productos a su *carrito*.

RE2.6.- Quitar productos de su *carrito*.

RE2.7.- Pagar los productos de su *carrito* con paypal.

RE2.8.- Ver la información de su compra.

RE2.9.- Agregar un comentario a la información de su compra.

RE2.10.- Modificar su contraseña.

C. Requerimientos de navegación entre las Interfaces de Usuario

El bosquejo de las IU se hace en paralelo con la elaboración de los DTIU. A cada bosquejo de IU se le asigna un número y un nombre que la identifica. En el bosquejo se definen los elementos que tendrá cada IU, mientras que en el DTIU se definen las transiciones entre las IU del sistema.

El fragmento de DTIU de la Fig. 1 contiene gran parte de las acciones que el usuario puede llevar a cabo y las transiciones que se disparan, tomando en cuenta los requerimientos específicos para el ***cliente***. Las IU en las que se disparan las transiciones están identificadas con su número y nombre. Las IU cuyos bordes no aparecen en negrita no contienen todas sus transiciones posibles en este fragmento de DTIU.

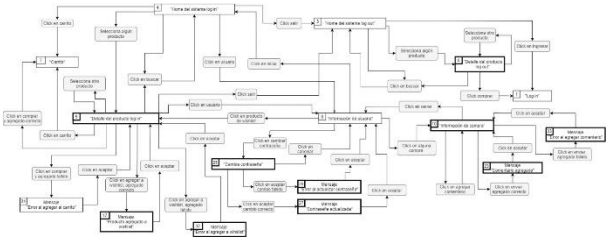


Fig. 1: DTIU del "Home del cliente"

Existen dos IU para el cliente, la IU #4 "Home del

sistema login" se presenta al usuario una vez que éste proporcionó su cuenta y su contraseña correctamente, mientras que la IU #3 "Home del sistema logout" es la interfaz que se presenta al usuario antes de entrar a su cuenta o cuando está en la IU #4 y selecciona la opción *salir*.

Desde la IU #3, el usuario puede consultar los productos existentes, buscar algún producto específico, seleccionar algún producto y ver sus detalles o ingresar con su cuenta. Para poder hacer una compra tendrá que ingresar con su cuenta. La IU #3 no tiene el borde en negrita porque en este fragmento de DTIU no se ilustra la opción *registrarse*.

Cuando el usuario está en la IU #4, tiene acceso a más acciones que en la IU #3, además de seleccionar un producto y ver sus detalles, podrá agregarlo a su *whish list* o a su *carrito*, y consultar su *carrito*. Cabe hacer notar que en el DTIU se contemplan los casos en los que falla el acceso a la base de datos y no se puede agregar un producto a la *whish list* o al *carrito.* Este tipo de falla también se tiene en cuenta en otras operaciones que implican un acceso a la base de datos. La IU #7 "Carrito" no está con los bordes en negrita porque las acciones que el usuario puede realizar en esta IU están documentadas en otro fragmento de DTIU. La IU #7 contiene, en otro fragmento de DTIU, opciones para consultar los productos del carrito, quitar o agregar productos y realizar la compra vía PayPal.

Otra de las opciones que tiene el usuario en la IU #4 es la de ver la información de su cuenta, al seleccionar esta opción se hace una transición hacia la IU #8 "Información del Usuario". En esta IU, el usuario podrá ver los productos agregados a su whish list, cambiar su contraseña y ver la información de sus compras. Cuando el usuario selecciona alguna de sus compras en la IU #8, se genera una transición hacia la IU #10 "Información de la compra" en la que se despliegan los detalles de la compra seleccionada con la opción de agregar algún comentario.

Como ya habíamos mencionado, una de las funciones importantes del DTIU es establecer una comunicación con el cliente en la fase del análisis de los requerimientos. Intencionalmente hemos elegido este primer esbozo de DTIU para ilustrar su utilidad. Con el fragmento de DTIU de la Fig. 1, clientes y desarrolladores pueden darse una idea del funcionamiento de la interacción del usuario con el sistema y detectar omisiones en los requerimientos. En este caso particular, entre dos personas diseñaron y elaboraron el DTIU y lo presentaron a una tercera persona, quien identificó la falta de una opción para borrar un producto de la whish list. Este requerimiento ya está ahora incluido en la sección IV (RE2.4).

En la Fig. 2 se observa el bosquejo de la IU #5 "Detalle de producto log in". La IU #5 se presenta al cliente solamente cuando ya inicio su sesión y eligió una categoría de productos en la IU #4 "Home del sistema login" la cual puede ser: libros, revistas, música o películas. El bosquejo muestra que se despliega una lista de los productos de la categoría que el usuario seleccionó. Cuando selecciona alguno los productos de la lista, se muestran los detalles del producto. Las acciones que el usuario puede llevar a cabo en esta interfaz son: 1.- Agregar el producto seleccionado a su "wish list" mediante el botón *agregar a whishlist*, 2.- Iniciar el proceso de compra del producto mediante el botón *comprar*, 3.- Ver el contenido de su carrito seleccionando el ícono que lo representa, y 4.- Seleccionar otro producto de la lista para ver sus detalles, 5.- Ver su información de usuario, y 6.- Salir de su cuenta mediante el botón *salir*. Cada una de las acciones anteriores determina un caso de uso en la IU #5. Los casos de uso tienen una numeración que servirá para hacer referencia a ellos durante la elaboración de los DSD en el diseño detallado, y es la siguiente:

5. IU #5 "Detalle del producto login"
   - 5.1. Agregar a wish list
   - 5.2. Click en carrito
   - 5.3. Click en comprar
   - 5.4. Seleccionar otro producto
   - 5.5. Click en usuario
   - 5.6. Click en salir
   - 5.7. Click en buscar

La mayoría de estos casos de uso requieren de transiciones hacia otras IU para poder completarlos. Los casos de uso incluidos en la IU #5 están relacionados con los requerimientos específicos RE2.2, RE2.3 y RE2.5 de la sección IV.B. Por disciplina, se contemplan los casos de falla y los casos de éxito en los accesos a la base de datos. Por ejemplo, en el DTIU de la Fig. 1 se observa que tanto en el caso de uso 5.1 como en el 5.3 de la IU #5 se toman en cuenta el éxito y la falla al procesar la solicitud.

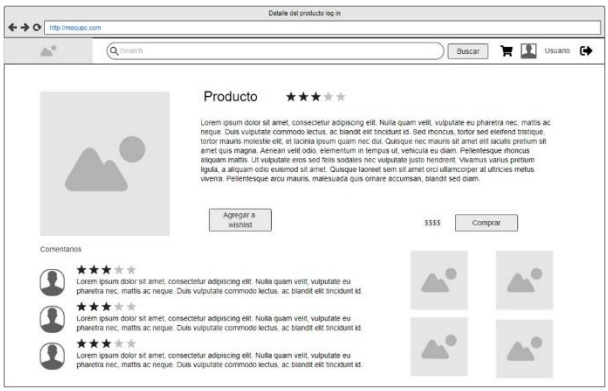


Fig. 2: Bosquejo de la IU #5 "Detalle de producto log in"

En el ejemplo de la sección VII haremos referencia desde el diseño detallado al DTIU de la Fig. 1 y a los casos de uso relacionados con la IU #5 de la Fig.2. En esta sección hemos mostrado que con la elaboración del

DTIU y de los bosquejos de IU se liga la fase de requerimientos con la de diseño de la siguiente forma: vista desde la *fase de requerimientos*, esta actividad ayuda a identificar los casos de uso y a acordar con el cliente la manera en la que se obtendrán los servicios del sistema; vista desde la *fase de diseño*, con la elaboración de DTIU y bosquejos de IU, se determina el diseño del contenido de las IU y las posibles transiciones entre éstas.

Los DTIU que usamos para documentar los requerimientos de navegación de la tienda virtual se elaboraron con *Google diagrams.net*, y los bosquejos de IU con *moqups* (https://app.moqups.com/).

V. BREVE INTRODUCCIÓN A LOS DIAGRAMAS DE SECUENCIA DETALLADOS (DSD)

Los DSD [4] son una particularización de los diagramas de secuencia UML, y tienen como objetivo ayudar en el cumplimiento de tres objetivos fundamentales de diseño: *i*) Anticipar y resolver problemas potenciales, *ii*) Especificar los requerimientos de cada módulo de la arquitectura del SW y *iii*) Garantizar que se cumplirán los requerimientos del sistema cuando la implementación se basa fielmente en el diseño.

A. Reglas para construir los DSD

1.- *Identidad de las secuencias*. - Cada secuencia y subsecuencia debe contar con un identificador y un nombre únicos. Las secuencias que comienzan con una acción del usuario deben tener un identificador que comience con un número igual al número de IU en la que se realiza la acción (en el DTIU), de lo contrario, el identificador de la secuencia debe comenzar con una letra.

2.- Identidad de las columnas. - Todos los módulos que participan en una acción de usuario o del sistema deben mostrarse en una columna separada marcada con un nombre que los identifique.

3.- Continuidad de las secuencias. - Debe haber una instrucción "ir a" al final de cada secuencia / subsecuencia con una referencia a los identificadores de todas las posibles secuencias de continuación.

4.- Funciones internas de un módulo. - Las funciones que se llevan a cabo internamente dentro de un módulo se explican en un cuadro de texto en lenguaje natural.

5.- Identidad de los mensajes. - Los mensajes de un módulo a otro deben tener una etiqueta que indique la identidad del mensaje y los parámetros que “viajan” en él. Cuando se trata de la invocación de un método, la identidad del mensaje es el nombre del método. Los mensajes de retorno de la invocación a un método no tienen identidad (ver la regla 6).

6.- Mensajes de retorno. - Los mensajes de retorno de la invocación a un método, se indican mediante una flecha con línea punteada. En este tipo de mensaje, se escriben los tipos de datos devueltos (ver regla 7).

7.- Valor específico de retorno. - En caso de que la secuencia dependa del valor específico enviado en el parámetro de un mensaje de retorno, la etiqueta del mensaje de retorno debe incluir este valor específico. Posteriormente se especifica una secuencia por separado para cada posible caso del valor de retorno.

Los DSD que usamos para documentar el diseño detallado de la tienda virtual se elaboraron con diagrams.net.

VI. DISEÑO DE ALTO NIVEL

Durante el diseño de alto nivel, se elaboró el modelado de la base de datos y el diseño arquitectónico, en el cual se especificaron los módulos de SW que componen al sistema. Estos módulos estuvieron determinados por la tecnología en la que se desarrolló el sistema, que fue Angular [10] más PHP. En la Fig. 3 se muestran las cuatro capas con las que opera el sistema. La vista, el controlador y el servicio corren en el cliente, mientras que el modelo corre en el servidor. Un componente contiene a la vista y al controlador, ambos se ejecutan en el navegador del usuario. La vista está codificada en html, y se comunica con el controlador mediante la notificación de eventos. El controlador, en javascript, responde a las acciones del usuario y se comunica con el servicio mediante la llamada a funciones. El servicio, en javascript, es el intermediario entre el controlador y el modelo. El modelo, en PHP, contiene la lógica de negocio y los mecanismos para comunicarse con la base de datos. La información obtenida de la base de datos la regresa en formato JSON al servicio. Cada una de las IU del sistema contiene estas cuatro capas. Como la comunicación entre la vista y el controlador la maneja Angular automáticamente, en el diseño detallado se consideran como un solo módulo llamado *Componente*.

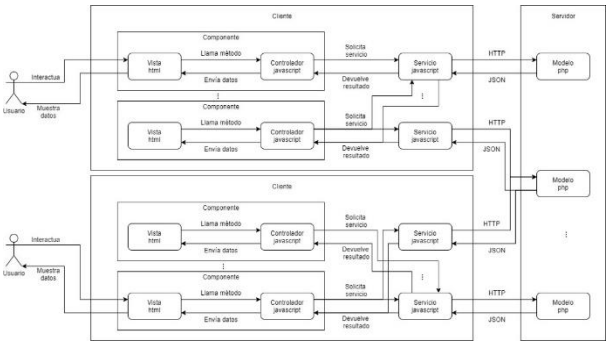


Fig. 3: Capas de un sistema web implementado en *Angular+PHP*

VII. DSD EN EL DISEÑO DETALLADO

Otra de las funciones clave del DTIU es ayudar en la transición de la fase de requerimientos a la fase de diseño mediante la liga entre los DTIU y los DSD. Con esta metodología se hacen referencias cruzadas entre las IU del sistema definidas en la fase de requerimientos, y los diagramas de secuencia del diseño detallado. A continuación, mostramos un ejemplo mediante el DSD del caso de uso 5.1 "Agregar a wish list" (ver Fig. 4). Este caso de uso está incluido en la IU #5 "Detalle de producto log in", cuyo bosquejo está en la Fig. 2. Las transiciones desde y hacia la IU #5 están documentadas en el DTIU de la Fig. 1.

El DSD de la Fig. 4 comienza con el número 5 porque este caso de uso se lleva a cabo dentro de la IU #5. La primera columna del diagrama representa al usuario, ya que el caso de uso inicia con la acción del usuario: *click en agregar a whishlist*. Los módulos que intervienen en esta secuencia son: "Detalle del Producto login component. html", "Swal", "Detalle del Producto login component.ts", "Producto Service.ts", "Producto model.php" y "Base de datos".

Cuando "Detalle del Producto login component.Html" recibe la acción del usuario, llama al método clickWhishlist( ) en "Detalle del Producto login component.ts", que a su vez llama al método agregaWishlist( ) que está en "Producto Service.ts", con los parámetros: identidad del usuario e identidad del producto que seleccionó.

"Producto Service.ts" se encuentra en el cliente y se comunica con "Producto model.php" que está en el servidor mediante el protocolo HTTP con un POST que lleva el comando *agregaWishlist* y la información necesaria para agregar el producto correspondiente a la tabla *wishlist* en la base de datos. Nótese que el mensaje entre "Producto model.php" y la base de datos contiene el comando exacto que es necesario para insertar la tupla en la base de datos.

Posteriormente se indica cómo continúa la secuencia para el caso no exitoso (secuencia 5.1.1) y para el caso exitoso (secuencia 5.1.2). Cuando se recibió un *false*, la ventana muestra un mensaje de error, y cuando se recibió un *true* se muestra un mensaje de confirmación de la operación. Esto lo hace en ambos casos "Detalle del Producto login component.ts" quien invoca al método fire de la librería "Swal" para desplegar una ventana emergente (modal). La secuencia termina cuando el usuario cierra la ventana modal y se reactiva la IU #5.

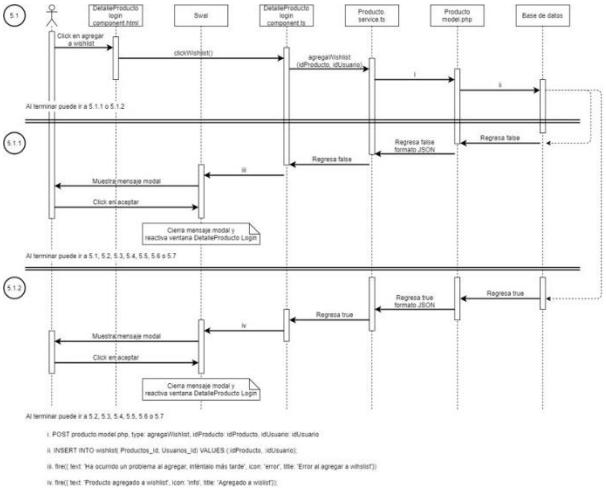


Fig. 4: DSD del caso de uso: "5.1 Agregar a wishlist"

Al terminar cada secuencia, 5.1, 5.1.1 y 5.1.2, hay una nota que indica cuál puede ser la continuación de la secuencia. En los casos 5.1.1 y 5.1.2, como el usuario termina en la IU #5, las secuencias posibles son las de los casos de uso contenidos en interfaz #5, es decir: 5.1, 5.2, 5.3, 5.4, 5.5, 5.6 o 5.7, los cuales están documentados en la especificación de requerimientos (sección IV.C).

Cabe señalar que, al elaborar un DSD, quedan definidos los nombres de los métodos y de los parámetros que se utilizarán durante la implementación. Esto ayuda a que los nombres sean ilustrativos y tengan homogeneidad en el código, haciéndolo más legible.

Un buen diseño detallado con DSD, define las tareas que hace cada módulo y la forma en la que se envía y recibe la información entre los módulos, lo que evita dejar cabos sueltos que pudieran ocasionar fallas en etapas posteriores.

VIII. RESULTADOS: IMPLEMENTACIÓN DEL SISTEMA DESDE EL DTIU Y LOS DSD

La implementación del sistema se hizo en Angular [10], una de las tecnologías más usadas en la actualidad para construir aplicaciones web. La base de datos se implantó con MySQL con una interfaz hecha en php. El desarrollo de la tienda virtual fue un proyecto que se desarrolló a lo largo de 41 semanas. El tiempo de desarrollo no se contabilizó de forma rigurosa, es decir, por horas, ya que no era viable logísticamente. Sin embargo, se hizo una medición por semana. La fase de requerimientos, que incluyó la elaboración de los DTIU y mockups de las IU, duró 11 semanas; la de diseño, que incluyó el diseño detallado y la elaboración de los DSD, duró 21 semanas; y la de implementación, que incluyó las pruebas de aceptación se hizo en 9.

En la fase de requerimientos se utilizó el 26.8% del esfuerzo total de desarrollo. La fase de diseño implicó

aproximadamente el 51.3% de este esfuerzo, mientras que la implementación implicó el 21.9 % del esfuerzo de desarrollo. Cabe mencionar que una semana y media de las 9 semanas del esfuerzo de implementación, se empleó en la solución de la incorporación del sistema de pago "paypal" al proyecto, es decir, a la solución de un problema técnico. Esto es interesante porque significa que la implementación de prácticamente todos los requerimientos del sistema fue de tan solo 7 semanas y media, que es un muy buen resultado dado el tamaño del sistema y que fue desarrollado por un solo alumno, es decir, un programador novato que no se dedicó de tiempo completo al desarrollo del sistema.

IX. ALCANCE DE LA METODOLOGÍA PROPUESTA

A. Modelos de desarrollo de Software

Un *proceso de desarrollo* de SW es el conjunto estructurado de las actividades requeridas para realizar un sistema de SW. Estas actividades son: especificación de requerimientos, diseño, codificación, validación (pruebas) y mantenimiento. Al proceso de desarrollo de SW también se le conoce como ciclo de vida del SW. Por otra parte, un *modelo de desarrollo* de SW es una representación abstracta de este proceso. Un *modelo de desarrollo* de SW determina el orden en el que se llevan a cabo las actividades del proceso de desarrollo de SW [11]. Los *Modelos Tradicionales* (también llamados *pesados*), promueven la disciplina por medio de la planificación y la comunicación escrita, dentro de los modelos tradicionales se encuentran los modelos *tipo cascada*, los *evolutivos* y los de *reutilización de componentes*. Los *modelos en cascada* son recomendables cuando se tienen requerimientos bien definidos y estables; requerimientos fuertemente acoplados o complejos; proyectos donde interviene una gran cantidad de personas y, cuando es más importante entregar un sistema funcionando correctamente que cumplir con una fecha de entrega preestablecida. En los *modelos evolutivos*, se trabajan los requerimientos al inicio de cada iteración para aumentarlos, corregirlos o redefinirlos. Estos modelos permiten entregar al cliente una versión parcial de un sistema de manera que permita obtener retroalimentación y evite problemas con la integración de un código muy grande. En los *modelos de reutilización de componentes* las funcionalidades de los componentes o módulos deben ser similares a las nuevas funcionalidades que se requieren, de tal forma que el esfuerzo en modificar los componentes base sea el menor posible. En [12] se puede consultar información detallada acerca de cada uno de estos modelos.

Por otra parte, los *Modelos* y las *Metodologías Ágiles* surgieron a raíz de que los modelos tradicionales no están preparados para hacer frente a los cambios rápidos en los requerimientos. Los modelos ágiles son técnicas de desarrollo de SW basados en un desarrollo iterativo e incremental en ciclos muy cortos. Uno de sus principales objetivos es minimizar el costo de los cambios en los requerimientos. Estos modelos y metodologías dan prioridad a la interacción entre los individuos y a la comunicación con el cliente. En [13] se puede consultar una revisión de los modelos y las metodologías ágiles. Si bien las metodologías ágiles tienen una gran aceptación en la industria del SW (según sus fundadores), éstas solo son aplicables cuando se dan las siguientes condiciones: proyectos pequeños y equipos con menos de 100 personas, requerimientos cambiantes, equipo de desarrollo competente y cliente dispuesto a participar con el equipo [14].

El modelo de desarrollo que se adopte depende de cada proyecto, ya que cada uno tiene necesidades diferentes. Hay que tomar en cuenta la capacidad y experiencia del personal y el tipo de proyecto a desarrollar para elegir algún método específico. La cantidad de personal con el que se cuenta también puede llegar a ser decisivo. No es necesario limitar la elección a un solo modelo de desarrollo pues, en ocasiones, es mejor combinar varios modelos.

B. ¿Cuándo es útil la metodología propuesta?

La metodología propuesta es aplicable a la parte final de la fase de recolección de requerimientos, a la fase diseño y al inicio de la fase de implementación de cualquier otra metodología, tanto tradicionales como ágiles. Es por esto que no se incluyen aspectos administrativos como la planeación, control de calidad, aseguramiento de la calidad, pruebas ni mantenimiento.

Cuando se habla de metodologías para desarrollar SW es importante mencionar el contexto, pues hay metodologías para: la mejora de la calidad como CMMI y MOPROSOFT; el desarrollo técnico de los proyectos, como las mencionadas en la subsección anterior; resolver algún problema específico, como es el caso de la metodología que aquí proponemos.

Independientemente del modelo de desarrollo que se elija para desarrollar un sistema de SW, las empresas suelen enfrentarse a un problema común: la incorporación de desarrolladores novatos en algún proyecto. Es sabido que el crear un buen diseño, antes de comenzar a codificar, mejora la robustez de un producto de SW. Además, si hay cambios o modificaciones durante el proceso de desarrollo, es mucho más conveniente atacar el problema desde el diseño que desde el código. Nuestra metodología hace énfasis en el diseño para obtener estas ventajas y además facilitar la implementación. Documentar el diseño detallado mediante DTIU y DSD ayuda a resolver el problema de manera conceptual y por lo tanto ayuda a que los programadores entiendan mejor lo que deben codificar.

X. CONCLUSIONES Y TRABAJO FUTURO

Es más probable que los desarrolladores de SW estén interesados en adoptar una metodología si hay casos de estudio que evidencien sus ventajas y ejemplifiquen la forma de aplicarla. En este trabajo presentamos la aplicación de una metodología que combina los Diagramas de Transiciones entre Interfaces de Usuario (DTIU) [3] y los Diagramas de Secuencia Detallados (DSD) [4] para construir una tienda virtual de libros, revistas, música y películas. Los DTIU son diagramas que permiten modelar la navegación entre las interfaces de usuario de un sistema, mientras que los DSD son diagramas de secuencia construidos bajo un conjunto de reglas hechas especialmente para documentar el diseño detallado. Los bosquejos de las Interfaces de Usuario (IU) en combinación con los DTIU permitieron especificar claramente la navegación del usuario por la tienda virtual. Mientras que los DSD sirvieron para elaborar el diseño detallado de la tienda.

Los requerimientos específicos se relacionan con los bosquejos de IU en función de los casos de uso que pueden iniciarse desde cada IU. Posteriormente, estos casos de uso se relacionan con los DSD del diseño detallado, lo que facilita la transición de los requerimientos al diseño. Los DSD se construyen haciendo referencia a los bosquejos de IU, a los DTIU y a los componentes del sistema.

Esta metodología ayuda a verificar que la implementación del diseño cumpla con los requerimientos del sistema relacionados con la interacción con el usuario. Además, ayuda a detectar casos especiales y posibles fallas u omisiones en los requerimientos y el diseño, facilitando la construcción de sistemas robustos. Para dar un ejemplo concreto de una de las ventajas de los DTIU en la fase de requerimientos, reportamos un caso específico en el que, mediante la visualización gráfica de la interacción del usuario con el sistema, identificamos un requerimiento faltante.

De nuestros resultados se aprecia que el mayor esfuerzo se llevó a cabo durante la fase de diseño detallado. Este esfuerzo implicó la determinación de: *i*) las tareas que hace cada módulo, *ii*) la forma en la que se comunican los módulos del sistema y la secuencia en la que lo hacen, *iii*) los casos exitosos y los posibles casos no exitosos, *iv*) los nombres de los métodos y de sus parámetros y *v*) las instrucciones para la base de datos. Pudimos observar que esto permitió ahorrar tiempo durante la fase de implementación. Una de las grandes ventajas de dedicar un esfuerzo importante al diseño detallado, es que el tiempo de implementación disminuye considerablemente, ya que, si el diseño detallado es lo suficientemente completo, la implementación solo se limitará a la transcripción del diseño detallado al código y a resolver detalles muy particulares de la tecnología con la que se trabaje.

La metodología ilustrada en este caso de estudio facilita la modificación y el mantenimiento de los sistemas, ya que, los DTIU hacen referencia a las IU, los DSD hacen referencia a las IU y a los casos de uso que contienen, a su vez, estos casos de uso están relacionados con los requerimientos específicos. Entonces, si cada secuencia que se implementa en el código tiene documentado su DSD correspondiente, se facilita el seguimiento inverso desde el código hacia el diseño detallado y los requerimientos.

Esperamos que este caso de estudio sirva de motivación y de guía para aplicar la metodología DTIU con DSD en la construcción de otros sistemas de SW y que, de esta manera, otros desarrolladores de SW puedan aprovechar sus ventajas.

Estamos trabajando en una aplicación que automatiza parte de la implementación generando automáticamente el esqueleto del código de una aplicación web a partir de un DTIU. Este generador de código trabaja con la tecnología React [15]. El objetivo es brindar otra manera más en la que los DTIU faciliten la implementación de sistemas de SW.

REFERENCIAS

[1] Bano, M., Zowghi, D. A systematic review on the relationship between user involvement and system success. *Information and Software Technology*, 2015, 58, 148-169. https://doi.org/10.1016/j.infsof.2014.06.011

[2] Mendez Fernandez D, Wagner S., Kalinowski M., Felderer M., Mafra P., Vetro A., Conte T., Christiansson M.-T., Greer D., Lassenius C., Mannisto T., Nayabi M., Oivo M., Penzenstadler B., Pfahl D., Prikladnicki R., Ruhe G., Schekelmann A. , Sen S., Spinola R., Tuzcu A., de la Vara J. L., and Wieringa R. 2017. Naming the pain in requirements engineering. *Empir. Software Eng*. 2017, 22 (5), 2298–2338 https://link.springer.com/article/10.1007/s10664-016-9451-7

[3] Gómez, M. C., Cervantes, J.: User Interface Transition Diagrams for Customer-Developer Communication Improvement in Software Development Projects. *Journal of Systems and Software* 2013. 86 (9), 2394-2410. https://doi.org/10.1016/j.jss.2013.04.022

[4] Gómez-Fuentes, M. C., Cervantes-Ojeda J. Sequence Diagrams Tailored for Software Design used to Build a Carpooling Management System.7th International Conference in Software Engineering Research and Innovation (*CONISOFT*) IEEE, October 2019, 116-122. https://doi.org/10.1109/CONISOFT.2019.00025

[5] Ricca, F., Scanniello, G., Torchiano, M., Reggio, G., Astesiano, E. Assessing the effect of screen mockups on the comprehension of functional

requirements. *ACM Transactions on Software Engineering and Methodology*. 2014, 24, (1) 1-38.
https://dl.acm.org/doi/abs/10.1145/2629457

[6] Gómez-Fuentes M. Cervantes-Ojeda J. Application of User Interface Transition Diagrams in the Construction of a Software System. 7th International Conference in Software Engineering Research and Innovation (*CONISOFT*), IEEE, October 2019,123-131.
https://doi: 10.1109/CONISOFT.2019.00026

[7] Cervantes-Ojeda J., Badillo-Salas A., Gómez-Fuentes M.C. Specialized Tool for Editing User Interface Transitions Diagrams (UITD). 9th International Conference in Software Engineering Research and Innovation (*CONISOFT*), IEEE, (2021, in press)

[8] Moody, D.L. The "Physics" of notations: towards a scientific basis for constructing visual notations in software engineering. *IEEE Transactions on Software Engineering*, 2009, 35 (6), 756-779.
https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5353439

[9] Van Der Linden, D., Hadar, I. A systematic literature review of applications of the physics of notation. *IEEE Transactions on Software Engineering*. 2019, 45 (8), 736-759.
https://ieeexplore.ieee.org/abstract/document/8283537

[10] Angular, https://angular.io/

[11] Sommerville I., Software Engineering, 2007, 8ª ed., Addison-Wesley.

[12] Cervantes J. Gómez M. C., "Taxonomía de los Modelos y Metodologías de Desarrollo de Software más utilizados", Revista Universidades No. 52 enero-marzo 2012, pp. 37-47.
http://www.udual.org/CIDU/Revista/52/Revista52.pdf

[13] Gómez M. C., Cervantes J., González P.P., "Fundamentos de Ingeniería de SW", Universidad Autónoma Metropolitana, 2019.
http://www.cua.uam.mx/publicaciones-electronicas/libros/division-de-ciencias-naturales-e-ingenieria

[14] Fowler, M., (2000), The new methodology
http://www.martinfowler.com/articles/newMethodology.html

[15] React, https://es.reactjs.org/

Acerca de los autores

Luis Gerardo Ramírez-Viveros. Es estudiante de Ingeniería en Computación de la Universidad Autónoma Metropolitana Unidad Cuajimalpa. Actualmente trabaja como desarrollador web en el área de dirección de tecnología en Softliu. Sus principales áreas de interés son Inteligencia Artificial e Ingeniería de Software.

María del Carmen Gómez-Fuentes. Es Doctora en Ciencias (Computación) por la Universidad Nacional Autónoma de México. Estudió Ingeniería Electrónica en la Universidad Autónoma Metropolitana (UAM). Actualmente es Profesor Asociado de tiempo completo en la UAM Unidad Cuajimalpa, en el Departamento de Matemáticas Aplicadas y Sistemas. Trabajó 8 años como ingeniero de software en la empresa Alcatel participando durante tres años y medio en el departamento de Diseño y Desarrollo de Software en Alcatel Bell, Bélgica. Su principal área de interés es la Ingeniería de Software, particularmente la Ingeniería de Requerimientos, el modelado y el desarrollo de sistemas.

Jorge Cervantes-Ojeda. Es Doctor en Ciencias (Computación) por la Universidad Nacional Autónoma de México. Hizo su maestría en Ciencias de la Computación en la UNAM. Estudió Ingeniería Electrónica en la Universidad Autónoma Metropolitana. Trabajó durante ocho años y medio en la empresa Alcatel-Indetel como diseñador de Software para centrales telefónicas digitales. Como Diseñador y líder de Inspecciones de Software. Colaboró en Alcatel-Bell (Bélgica) en el área de Diseño de Software durante tres años y medio. Fue profesor en la Facultad de Estudios Superiores Cuautitlán (UNAM) durante ocho años impartiendo materias de electrónica y matemáticas. Actualmente es Profesor Asociado de Tiempo Completo en la UAM Unidad Cuajimalpa en donde imparte cursos de computación. Ha publicado trabajos de investigación en el área de Inteligencia Artificial, específicamente en Computación Evolutiva y Redes Neuronales y también en el área de Ingeniería de Software, que son sus principales áreas de interés.