COMPUTACIÓN E INFORMÁTICA

ReCIBE, Año 5 No. 1, Abril 2016

Trazabilidad y validación de requerimientos funcionales de sistemas informáticos mediante la transformación de modelos conceptuales

Oscar Carlos Medina
Grupo de Investigación, Desarrollo y Transferencia de Sistemas de Información Universidad Tecnológica Nacional
oscarcmedina@gmail.com

Marcelo Martín Marciszack
Grupo de Investigación, Desarrollo y Transferencia de Sistemas de Información Universidad Tecnológica Nacional
marciszack@gmail.com

Mario Alberto Groppo
Grupo de Investigación, Desarrollo y Transferencia de Sistemas de Información Universidad Tecnológica Nacional
sistemas@groppo.com.ar

Resumen: El objetivo de este trabajo es caracterizar un método que permita la trazabilidad y validación de requerimientos funcionales de un sistema de información mediante la transformación de modelos conceptuales. Para lo cual se construyó un

software denominado SIAR (Sistema Integral de Administración de Requerimientos) que administra los requerimientos funcionales y utiliza UML (Lenguaje Unificado de Modelado) para su representación como Casos de Uso. La finalidad principal de esta aplicación web es la gestión de Casos de Uso con una herramienta que agilice su registro, normalice su contenido y posibilite la trazabilidad de los cambios e implemente validaciones funcionales. Por ejemplo, un procedimiento automatizado de análisis de consistencia de Casos de Uso, para lo cual el sistema genera un grafo con la transición de estados de cada Caso de Uso que es analizado en un simulador de autómata finito determinista para verificar la cohesión de los escenarios en él definidos.

Palabras clave: Trazabilidad, Validación, Requerimiento funcional, Modelo conceptual, UML, Autómata finito determinista.

Abstract: The goal of this paper is to characterize a method that allows the traceability and validation of functional requirements of a computer system by transforming conceptual models with SIAR (Integrated System of Requirements Management). This software manages the functional requirements describing them as Use Cases using UML (Unified Modeling Language). The main purpose of this web application is the Use Cases management with a tool to expedite registration, normalize its contents and enable the traceability of changes and implement functional validations. For example, an automated process for consistency analysis of Use Cases, for which the system generates a graph with the state transition of each Use Case. Finally, the graph is analyzed in a deterministic finite automata simulator to verify the consistency of the defined scenarios defined.

key words: Traceability, Validation, Functional requirement, Conceptual model, UML, Deterministic finite automata.

1. Introducción y Motivación

Construir una aplicación web que gestione los requerimientos funcionales de un sistema de información según los lineamientos de UML (Lenguaje Unificado de Modelado), fue lo que motivó al equipo de proyecto a diseñar SIAR (Sistema Integral de Administración de Requerimientos).

El proyecto de investigación se denomina “Validación de Requerimientos a través de Modelos Conceptuales” del GIDTSI (Grupo de Investigación, Desarrollo y Transferencia de Sistemas de Información), el cual depende del Departamento Ingeniería en Sistemas de Información de la Universidad Tecnológica Nacional, Facultad Regional Córdoba y “busca dar solución a uno de los principales problemas de la Ingeniería de Requerimientos relacionado a la elicitación y especificación de requerimientos, que vincula las distintas etapas del proceso de desarrollo de software manteniendo la trazabilidad de los mismos hasta su validación e implementación”.

Como el Modelo Conceptual tiene por objetivo el registro y la estandarización de requerimientos, el desarrollo de SIAR tiene por fin administrar en forma integral los requerimientos funcionales de un proyecto de software según la metodología UML.

SIAR es una aplicación web que permitió registrar en forma normalizada los casos de uso y cuya primera versión comprende el siguiente alcance:

Administración de los atributos de un proyecto de software.

- Diseño y validación del Modelo Conceptual.
- Trazabilidad en los cambios de diseño y su versionado.
- Gestión de los alcances de cada versión del proyecto y los casos de uso asignados.
- Gestión de los atributos de un caso de uso, incluyendo actores, pre-condiciones, post-condiciones, cursos de acción, normal y alternativos, y su versionado.
- Clasificación y priorización de los casos de uso.
- Visualización de consultas y generación de reportes en distintos formatos, inclusive XPDL, para comunicarse con otras aplicaciones.
- Vinculación con atributos de procesos de negocio, de actividades de negocio que los componen y los casos de uso asociados a estas actividades.

- Vinculación con atributos de procesos de negocio, de actividades de negocio que los componen y los casos de uso asociados a estas actividades.

2. Construcción de una herramienta para la administración integral de Casos de Uso

La herramienta ha sido diseñada con un criterio de lógica que permite describir una serie de relaciones en cascada con distintos grados de desagregación. Estas relaciones de SIAR se detallan en el gráfico de la **Fig. 1.**

Figura 1. *Relaciones de SIAR.*

SIAR permite trabajar con diferentes Sistemas, cada uno con sus propias Versiones, cada Versión cubre un número limitado de Alcances, cada Alcance

gestiona un grupo de Casos de Uso, cada Caso de Uso está compuesto por una secuencia ordenada de Pasos, finalmente un paso puede o no tener Alternativas.
Los Pasos de un Caso de Uso permiten efectuar tareas que se describen con el soporte del sistema que se ha desarrollado.
Un Caso de Uso puede ser del tipo Concreto o Abstracto. Un caso de uso es Abstracto si no puede ser realizado por sí mismo, o si no es Concreto ya que puede ser iniciado por un actor y realizado por sí mismo.
Los pasos son las actividades que deberán realizarse para llevar a cabo algún proceso (Rumbaugh, Jacobson, Booch, 1999).
Cada paso posee alternativas que a su vez puede tener pasos y estos volver a tener alternativas.

3. Alcance de la primera versión de SIAR

A continuación se explica brevemente las funcionalidades más importantes del desarrollo informático que se presenta en este trabajo.

3.1 Selección del tipo de pruebas

Se planteó la necesidad de generar un sistema que no solo administre los casos de usos por separado, sino también gestione en forma integral todo el proyecto que lo contiene. Por ejemplo, la **Fig. 2** muestra la consulta de Casos de Uso de un sistema a partir de su versionado.

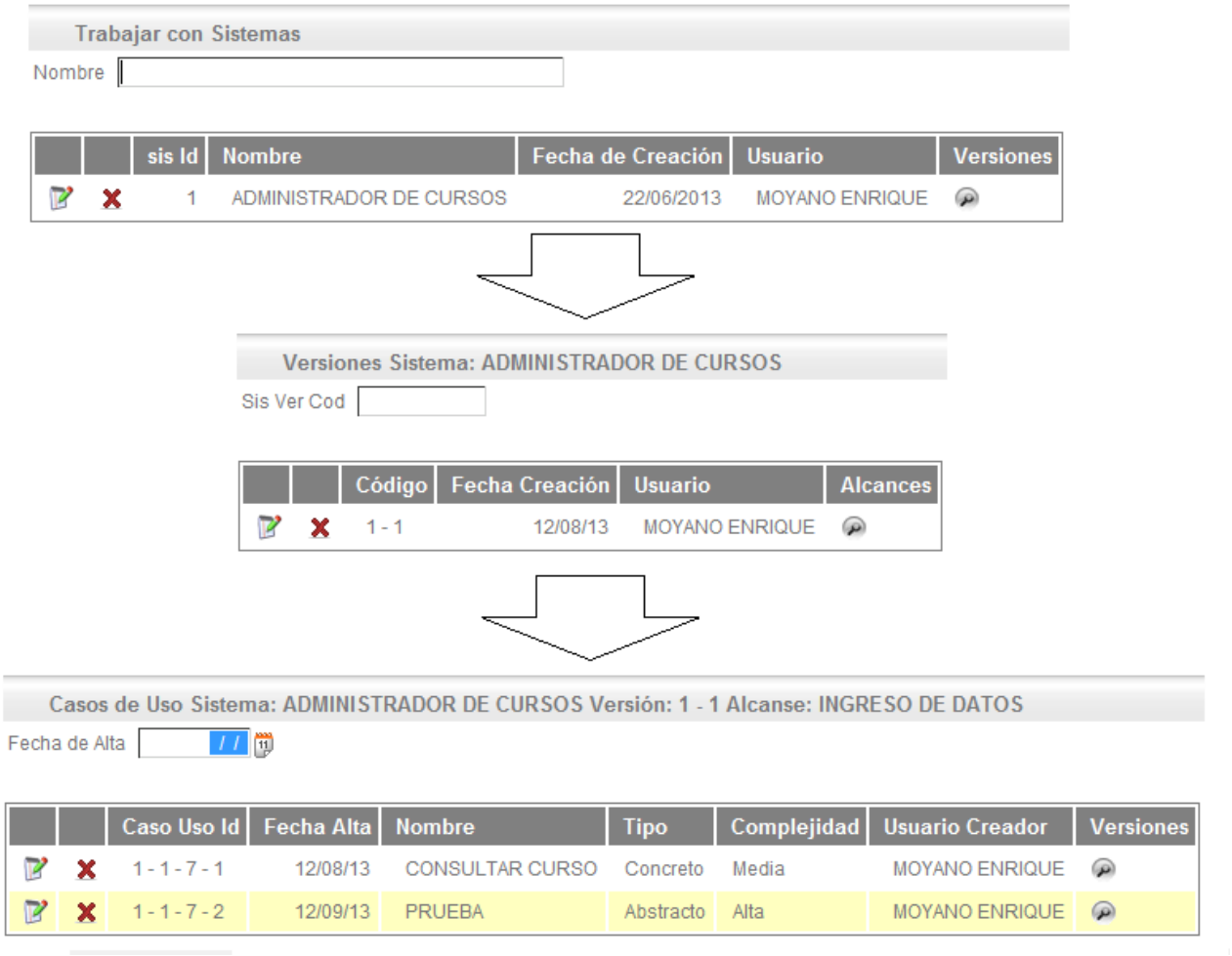


Figura 2. *Consulta de Casos de Uso de un Sistema.*

Para cumplir con este objetivo se incluyó también el concepto de sistema, versiones por sistema y alcances de cada versión. A su vez cada caso de uso incluye pasos y alternativas.

3.1 Trazabilidad de los cambios de Casos de Uso

La aplicación permite versionar Casos de Uso. Se ofrecen distintas herramientas para llevar a cabo el seguimiento de la trazabilidad, por ejemplo el listado de comparación de versiones de un caso de uso, detalla los cambios entre dos versiones seleccionadas.

3.3 Exportación a archivos XML

Esta es una funcionalidad que proporciona el sistema para poder intercambiar datos con otras aplicaciones como por ejemplo un autómata finito. El archivo XML representa en este protocolo de intercambio de datos el grafo de estados del Caso de Uso que surge de la equivalencia con sus pasos y alternativas.

4. Validación de consistencia de Casos de Uso con simuladores de autómatas finitos

En la actualidad UML es reconocido como el estándar para el modelado de proyectos de software. Los componentes principales de este lenguaje de modelado son los Casos de Uso ya que especifican el comportamiento deseado del sistema. Por definición el caso de uso “especifica una secuencia de acciones, incluyendo variantes, que el sistema puede ejecutar y que produce un resultado observable de valor para un particular actor”.

Una vez generado un caso de uso, es necesario comprobar y asegurar su validez. La verificación de consistencia de la secuencia de acciones descripta en el caso de uso es una de las tareas que permiten su validación.

Es deseable que esta verificación pueda realizarse de manera automatizada para lo cual se podría trabajar con autómatas finitos deterministas, ya que un autómata finito es un conjunto de estados y un control que se mueve de un estado a otro en respuesta a entradas externas (Marciszack, Pérez, Castro, 2013). Se llama determinista al autómata que puede estar únicamente en un estado en un momento determinado. El desafío es transformar el caso de uso en un autómata finito determinista para validar su cohesión secuencial.

Partiendo de estas premisas, se formula la siguiente pregunta:

¿Es factible determinar un método, basado en simuladores de autómatas finitos deterministas, que verifique la consistencia de la secuencia de acciones, incluyendo variantes, definidas en un caso de uso?

La funcionalidad de SIAR que se detalla a continuación propone una alternativa viable para esta necesidad, que se realiza en 3 pasos:

4.1 Registración normalizada del requerimiento.
4.2 Transformación del Caso de Uso en una máquina de estados.
4.3 Validación de la consistencia secuencial de los “caminos” del Caso de Uso.

4.1 Registro normalizado del requerimiento

Como el modelo de requerimientos tiene por objetivo la captura de requerimientos funcionales (Jacobson y otros, 1992), la primera tarea que se emprendió, fue la construcción de SIAR para registrar en forma normalizada los casos de uso según el estándar UML.

El diseño de la interacción dentro de la prueba de usabilidad se divide en dos etapas. La primera identifica el menú, reconocimiento de íconos y manejo del cursor a través del evento click que activa cierta funcionalidad del sistema; mientras que la segunda trabaja directamente con la aplicación, en la que se muestra una serie de modelos tridimensionales de piezas prehispánicas, pudiéndose elegir alguno de ellos para realizar cuatro tareas básicas que son: acercar, alejar, rotar en eje X y rotar en eje Y.

4.2 Transformación del Caso de Uso en una máquina de estados

En segunda instancia se definió un conjunto de reglas de conversión del caso de uso en un grafo de estados para que sea la entrada a un simulador de autómata finito determinista.

Una vez completa la versión de un caso de uso, se genera un grafo de estados a partir de las siguientes definiciones:

- Todo caso de uso tiene al menos un curso de acción (“camino”) normal.
- Un caso de uso puede no tener ningún, o tener uno o tener varios cursos de acción alternativos.

- Cada paso de un curso de acción de un caso de uso responde a un estado y es un nodo de la máquina de estados.
- Todo caso de uso tiene un paso inicial, y es el primer paso de todos los cursos de acción. Este paso es el estado/nodo inicial.
- Todo caso de uso tiene un paso final por aceptación, y es el último paso del curso de acción normal. También puede ser el último paso de algún curso alternativo. Este paso es el estado/nodo final por aceptación.
- Un caso de uso puede no tener ningún, tener uno o tener varios finales por error, y son el último paso de un curso alternativo. Estos pasos son estados/nodos finales por error.
- Todo caso de uso pasa de un estado/nodo a otro por medio de una función de transición.
- Partiendo de un estado/nodo origen se puede continuar en un único estado/nodo destino, o en dos nodos/estados destino alternativos dependiendo de una condición de bifurcación.
- El grafo de estados asociado al caso de uso tiene un alfabeto de tres símbolos para indicar qué evento lo cambia de un estado/nodo a otro:
  - A = Por medio de una Acción determinada.
  - S = Cuando Si se cumple una condición que bifurca los cursos de acción.
  - N = Cuando No se cumple una condición que bifurca los cursos de acción.

Figura 3. *Bifurcación en la especificación de trazo fino de un Caso de Uso.*

Partiendo de un estado/nodo origen, en la función de transición puede estar asociado solamente uno de los símbolos: A, N o S. Con esto se cumple la condición necesaria de un autómata finito determinista. De esta manera, si la transición entre dos estados/nodos se da dentro de un mismo camino, se asocia el símbolo A. Si en cambio interviene una bifurcación (como se ejemplifica en la **Fig. 3**), la función de transición hacia el estado/nodo destino por cumplimiento de la condición de bifurcación, se asocia el símbolo S. Por el otro camino de la bifurcación, se asocia el símbolo N. Estas reglas de transformación se formalizan en una tabla de estados de casos de uso que se presenta en la **Fig. 4** a continuación:

Tabla De Estados Casos de Uso: 1 - 1 - 1 - 1 CONSULTAR CURSOS Versión: 6

Estado / Paso Origen	Estado / Paso Destino	Transición	Estado Final por Error	Tipo
1	2	A		S
2	3	A		S
3	4	A		S
4	4 - A	N		C
4 - A	4 - A - 1	A		S
4 - A - 1	4 - A - 2	A	SI	S
4	5	S		C
5	6	A		S
6	6 - A	S		C
6 - A	6 - A - 1	A		S
6	7	N		C

Figura 4. *Tabla de estados de un Caso de Uso. Transformación automática por SIAR.Tabla de estados de un Caso de Uso. Transformación automática por SIAR.*

Una vez generado el grafo de estados se expresa en protocolo XPDL, por ser el más adecuado para intercambiar modelos de procesos entre distintas herramientas. Este lenguaje “da soporte a la definición y a la importación/exportación de procesos, con el objetivo de que, aunque se modele un proceso en una aplicación, este modelo pueda ser usado por otras aplicaciones de modelado y/o por otras aplicaciones que trabajen en el entorno de ejecución” (Pérez, 2007).

```
<transicion
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="java:dominio.automatas.TransicionFinita">
    <simbolo>
        <simbolo>S</simbolo>
    </simbolo>
    <estado-inicio>
        <denominacion>4</denominacion>
    </estado-inicio>
    <estado-fin>
        <denominacion>5</denominacion>
    </estado-fin>
    <denominacion>f( 4, S) = 5</denominacion>
</transicion>
```

Figura 5. *Fragmento de archivo XPDL generado por SIAR.*

4.3 Validación de la consistencia secuencial de los "caminos" del Caso de Uso

Finalmente se ingresa el archivo XPDL (ver **Fig. 5**), que representa al caso de uso como grafo de estados (ver**Fig. 6**), al programa "Autómata Finito" que forma parte del conjunto de "Herramientas Prácticas para el Aprendizaje de Informática Teórica". Esta aplicación java "permite definir autómatas y gramáticas para la enseñanza y ejercitación de los alumnos de la asignatura Sintaxis y Semántica del Lenguaje que se dicta en la carrera de Ingeniería en Sistemas de Información de la Universidad Tecnológica Nacional Facultad Regional Córdoba" (U.T.N. F.R.C., 2009).

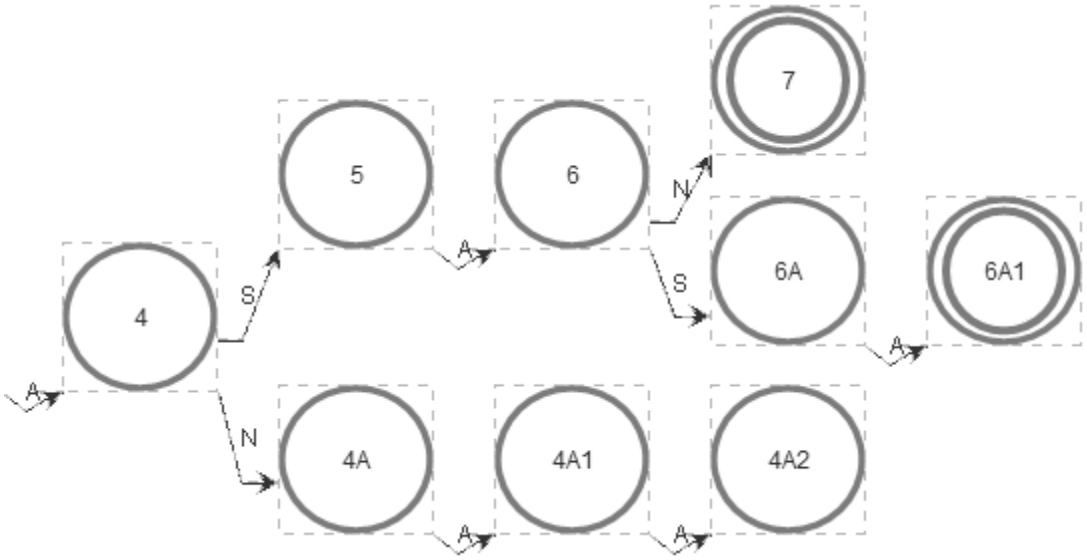


Figura 6. *Fragmento del grafo de estados en el simulador de autómatas finitos.*

El simulador posibilita probar el grafo de estados y comprobar si es aceptado o rechazado. En el segundo caso se puede visualizar el detalle para detectar la inconsistencia, la cual puede ser corregida en una nueva versión del caso de uso, generando un nuevo grafo de estados automáticamente. De esta manera se logra control y trazabilidad de los cambios en los diferentes “caminos” de los casos de uso. Vale recordar a F. P. Brooks cuando dijo: “la tarea más importante que el ingeniero de software hace para el cliente es la extracción iterativa y el refinamiento de los requerimientos del producto”.

4.4 Resultados

Se pone a consideración este método automatizado, implementado en el aplicativo SIAR que permite registrar, normalizar y controlar la trazabilidad de los casos de uso del Modelo Conceptual. Además transforma un caso de uso a formato XPDL para que un simulador de autómatas finitos pueda verificar la consistencia secuencial de los distintos caminos del caso de uso.

El aporte que realiza SIAR al proyecto en el que fue concebido, “Validación de Requerimientos a través de Modelos Conceptuales”, es el de constituirse como plataforma de software integradora de las aplicaciones que se utilizan en cada una de las líneas de investigación.

El presente desarrollo está fundamentado en una “Propuesta Metodológica para validación de Requerimientos Funcionales a través de Modelos Conceptuales” registrada en la República Argentina con Derecho de autor de producciones tecnológicas (rubro “Modelo de organización y/o gestión, Ciencia y cultura-Ciencia y tecnología”) según Expediente No.5229942. Esta metodología expone cómo se llega a identificar la necesidad de construir SIAR y la especificación de requerimientos funcionales de sus alcances, módulos y funcionalidades.

También “S.I.A.R. – Sistema Integral de Administración de Requerimientos” está registrado en la República Argentina con Derecho de autor de producciones tecnológicas (rubro “Máquina, equipo, instrumento y/o herramienta o su/s componente/s. Informática-software. Ciencia y cultura-Ciencia y tecnología”) según Expediente No.5229955.

En lo que respecta a la funcionalidad de análisis de consistencia, SIAR ofrece un método automatizado para validar la cohesión de un caso de uso desde el punto de vista de la transición de estados definidos intrínsecamente en los pasos de su especificación funcional. Permitiendo también enlazar este proyecto con un trabajo académico de la carrera Ingeniería en Sistemas de Información, el del Grupo de Herramientas Didácticas de Informática Teórica que contribuyó con el simulador de autómatas finitos.

Finalmente se tiene previsto hacer una transferencia de la metodología y el software a partir del año 2016 a emprendimientos tecnológicos de estudiantes próximos a graduarse y tutorados por el programa “Mentoreo de Emprendedores” de la Carrera.

5. Conclusiones y Trabajo Futuro

En el presente trabajo se ha analizado y comprendido el procedimiento y el modelo de datos asistido por una aplicación web denominada SIAR que gestiona los requerimientos funcionales y la trazabilidad de un sistema de información según los lineamientos de UML. Haciendo posible también enlazar este proyecto con un trabajo académico de la carrera Ingeniería en Sistemas de Información, el del Grupo de Herramientas Didácticas de Informática Teórica que contribuyó con el

simulador de autómatas finitos, que permitió implementar un método automatizado para validar la cohesión de un caso de uso desde el punto de vista de la transición de estados definidos intrínsecamente en los pasos de su especificación funcional.

Como lo señalaron Taurant y Marciszack: “Actualmente existen en el mercado una gran variedad de herramientas y metodologías que permiten la gestión de requerimientos de software a lo largo del ciclo de vida de desarrollo. Independientemente de la herramienta o metodología utilizada, la creación y mantención de un gran número de modelos y artefactos es realizada por el analista en forma manual, generando con gran frecuencia inconsistencias entre los modelos generados, impactando en la trazabilidad de los requerimientos.”

Es por esto que se propuso con SIAR, el desarrollo de una herramienta que permita al analista gestionar los requerimientos en forma asistida y parcialmente automatizada, por medio de la generación de modelos conceptuales como representaciones de máquinas abstractas, considerando que se alcanzaron estos objetivos parcialmente.

Para continuar trabajando con SIAR se propone extender la trazabilidad de los modelos de requerimientos de manera tal que permita medir el impacto de los cambios desde el Modelo Conceptual hasta el Modelo de Sistema de Información agregando técnicas de validación y funcionalidades de gestión de proyectos de software. Se está indagando la incorporación de “Patrones”, según el concepto de la Ingeniería de Software, en la validación de Modelos Conceptuales como el próximo paso a seguir para ampliar el alcance de esta herramienta.

Referencias

Jacobson, Ivar y otros (1992). Object Oriented Software Engineering. A Use Case Driven Approach. Addison Wesley.

Marciszack, Marcelo, Pérez,Ramiro, Castro, Claudia Castro (2013). Validación de Requerimientos a través de Modelos Conceptuales – Modelos y Transformaciones. WICC 2013.

Pérez, J. D. (2007). Notaciones y lenguajes de procesos. Una visión global. Tesis de Doctorado Universidad de Sevilla.

Rumbaugh, J., Jacobson,I., Booch,G. (1999). The Unified Modelling Language Reference. Addisson Wesley.

U.T.N. F.R.C. (2009). Proyecto Construcción de Herramientas Didácticas para la enseñanza y ejercitación práctica en laboratorio de Informática Teórica en las Carreras con Informática. Manual de Usuario – Grupo de Herramientas Didácticas.

Bibliografía adicional

Brooks, Frederik P. (1987). No Silver Bullet. Essence and Accidents in Software Engineering. IEEE Computer.

Chakraborty, Samarjit (2003). Formal Languages and Automata Theory-Regular Expressions and Finite Automata-. Computer Engineering and Networks Laboratory Swiss Federal Institute of Technology (ETH) Zurich.

Davis, A. (1993). Software requeriments. Object, functions and states. Pretince Hall International Inc.

Leonardi, C., Leite, J.C.S., Rossi, Gustavo (2001). Una estrategia de Modelado Conceptual de Objetos, basada en Modelos de requisitos en lenguaje natural. Tesis de Maestría Universidad Nacional de la Plata.

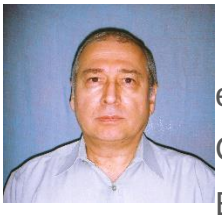
Sommerville, I. (2011). Software Enginnering, Computing Departament, Lancaster University, John Willey&Sons Ltd. 9a Edición en español. Editorial Pearson.

Notas biográficas:

Oscar Carlos Medina es Ingeniero en Sistemas de Información, egresado de la Universidad Tecnológica Nacional – Facultad Regional Córdoba (UTN – FRC), Argentina. En esta Universidad realizó los cursos de posgrado “Introducción a la Investigación, el Desarrollo y la Innovación”, “Formulación de Proyectos de I+D+i” y “Gestión de Proyectos bajo estándar PMI”. Es miembro de la carrera del Investigador Científico Categoría “E” UTN y forma parte de GIDTSI, Grupo de Investigación, Desarrollo y Transferencia de Sistemas de Información, dentro del proyecto de investigación EIUTNCO0003604 “Implementación de patrones en la validación de modelos conceptuales”, donde además prepara su postulación de Tesis de Doctorado. Tiene registrado títulos de propiedad intelectual en coautoría de “S.I.A.R. – Sistema Integral de Administración de Requerimientos” - Expediente No.5229955 y “Propuesta Metodológica para validación de Requerimientos Funcionales a través de Modelos Conceptuales” - Expediente No.5229942. Ha publicado diversos artículos técnicos y ha disertado en congresos universitarios sobre temas relacionados con la ingeniería de software, gobierno electrónico y firma digital. Actualmente es Docente auxiliar de 1ª interino, Coordinador del programa de “Mentoreo de Emprendedores ISI” del Departamento Ingeniería en Sistemas de Información de UTN - FRC. También participa en proyectos de consultoría a Gobierno y empresas privadas, integrando el plantel de CIDS, Centro de Investigación y Desarrollo de Sistemas de UTN – FRC.

Marcelo Martín Marciszack es Ingeniero en Sistemas de Información egresado de la Universidad Tecnológica Nacional, Facultad Regional Córdoba de Argentina, Magister en Ingeniería de Software graduado de la Universidad Nacional de la Plata de Argentina y Doctor por la Universidad de Vigo de España, en el programa de Doctorado: Ingeniería de Software basada en componentes reutilizables Aplicaciones interfaces Hombre-Máquina, donde se le concedió la nota máxima y Cum Laude. Es profesor titular Ordinario de la cátedra de Paradigmas de Programación de la Universidad Tecnológica Nacional, Facultad Regional Córdoba,

realizando también actividades docentes en la carrera de posgrado. En la actualidad se desempeña como Vice-Decano de la Facultad Regional Córdoba de la Universidad Tecnológica Nacional e Integrante del Comité Académico de la Maestría en Ingeniería en Sistemas de Información UTN – FRC. Ha sido Director de Departamento Ingeniería en Sistemas de Información UTN-FRC desde Diciembre 2005 hasta Diciembre de 2013. Es Director del Grupo de UTN de Investigación, Desarrollo y Transferencia de Sistemas de Información (GIDTSI) de la Facultad Regional Córdoba, con dependencia funcional de la Secretaría de Ciencia, Tecnología y Posgrado de la Universidad Tecnológica Nacional. Es Miembro de la carrera del Investigador Científico Categoría “B” Univ. Tecnológica Nacional y de la Carrera Investigador programa de Incentivos Categoría III – Octubre 2011. En la actualidad es Director del Proyecto de Investigación y Desarrollo “Implementación de Patrones en la Validación de Modelos Conceptuales”, período: 01 de Enero de 2015 al 31 de Diciembre de 2017 y codirector del proyecto “Metodología para determinar la exactitud de una respuesta, escrita en forma textual, a un interrogante sobre un tema específico, aplicando herramientas informáticas”, período: 01 de Enero de 2015 al 31 de Diciembre de 2016. Ha realizado innumerables publicaciones en Congresos de la especialidad y divulgación de actividades docentes. Ha participado en numerosos proyectos de transferencias dese 1997 hasta la fecha, ha participado como jurado en concursos docentes y actividades de investigación y en lo que respecta a la formación de recursos ha dirigido tesis de maestrías, dirección de pasantes en Proyectos de desarrollo y dirección de Becarios Alumnos y Graduados en Proyectos de Investigación.



Mario A. Groppo Groppo es Ingeniero en Sistemas de Información egresado de la Universidad Tecnológica Nacional, Facultad Regional Córdoba de Argentina y Doctor por la Universidad de Vigo de España, en el programa de Doctorado: Ingeniería de Software basada en componentes reutilizables Aplicaciones interfaces Hombre-Máquina, donde se le concedió la nota máxima y Cum Laude. Es profesor Asociado de la cátedra de Comunicaciones de la Universidad Tecnológica Nacional, Facultad

Regional Córdoba, realizando también actividades docentes en la carrera de posgrado. En la actualidad se desempeña como Director de la Maestría en Ingeniería en Sistemas de Información, posgrado de la Facultad Regional Córdoba de la Universidad Tecnológica Nacional. Es integrante del Grupo de UTN de Investigación, Desarrollo y Transferencia de Sistemas de Información (GIDTSI) de la Facultad Regional Córdoba, con dependencia funcional de la Secretaría de Ciencia, Tecnología y Posgrado de la Universidad Tecnológica Nacional. Es Miembro de la carrera del Investigador Científico Categoría “D” Univ. Tecnológica Nacional y de la Carrera Investigador programa de Incentivos Categoría IV. En la actualidad es codirector del Proyecto de Investigación y Desarrollo “Implementación de Patrones en la Validación de Modelos Conceptuales”, período: 01 de Enero de 2015 al 31 de Diciembre de 2017 y Director del proyecto “Metodología para determinar la exactitud de una respuesta, escrita en forma textual, a un interrogante sobre un tema específico, aplicando herramientas informáticas”, período: 01 de Enero de 2015 al 31 de Diciembre de 2016. Ha realizado publicaciones en Congresos de la especialidad y de divulgación de actividades docentes. Ha participado como organizador y jurado en congresos internacionales, Doctoral Symposium y actividades de investigación. En lo que respecta a la formación de recursos ha dirigido tesis de maestrías, dirección de pasantes en Proyectos de desarrollo y dirección de Becarios Alumnos y Graduados en Proyectos de Investigación. En la industria informática desarrolló su carrera en la empresa UNISYS.