## Patrocinadores

Ministerio de Educación y Ciencia · Zaragoza Ayuntamiento · Ministerio de Industria, Turismo y Comercio · Universidad de Zaragoza · Sun · Telefónica · Expo Zaragoza 2008 · Gobierno de Aragón, Departamento de Ciencia, Tecnología y Universidad

## Entidades Organizadoras

- AdaSpain.
- Asociación de Enseñantes Universitarios de la Informática (AENUI).
- Asociación de Técnicos Informáticos (ATI).
- Asociación Española para la Inteligencia Artificial (AEPIA).
- Asociación para la Interacción Persona-Ordenador (AIPO).
- Asociación para el Desarrollo de la Informática Educativa (ADIE).
- Ayuntamiento de Zaragoza.
- Capítulo Español de la IEEE Computational Intelligence Society.
- Comité Español de Automática (CEA).
- Conferencia de Decanos y Directores de Informática (CODDI) de las Universidades Españolas.
- Departamento de Informática e Ingeniería de Sistemas de la Universidad de Zaragoza.
- European Society for Fuzzy Logia and Technology (EUSFLAT).
- Federación de Asociaciones de Ingenieros en Informática (AI2).
- W3C España (World Wide Web Consortium).
- Programa Nacional de Tecnologías Informáticas - Dirección General de Investigación. Ministerio de Educación y Ciencia.
- Red Española de Metaheurísticas.
- Red Española de Minería de Datos y Aprendizaje.
- Sección Española de la European Association for Computer Graphics (EUROGRAPHICS).
- Sociedad de Arquitectura y Tecnología de Computadores (SARTECO).
- Sociedad de Ingeniería del Software y Tecnologías de Desarrollo del Software (SISTEDES).
- Universidad de Zaragoza.

ISBN: 978-84-9732-607-0

CEDI 2007 II Simposio sobre Seguridad Informática | SSI'07 |

THOMSON

# CEDI 2007

II CONGRESO ESPAÑOL DE INFORMÁTICA

ZARAGOZA SPAIN

AUDITORIO PALACIO DE CONGRESOS
11 AL 14 DE SEPTIEMBRE DE 2007

# II Simposio sobre Seguridad Informática

| SSI'07 |

**EDITORES**
Benjamín Ramos Álvarez y Arturo Ribagorda Garnacho


# ACTAS DEL
# II SIMPOSIO SOBRE SEGURIDAD INFORMÁTICA [SSI'2007]

**EDITORES**

Benjamín Ramos Álvarez
Arturo Ribagorda Garnacho

**SIMPOSIO ORGANIZADO POR**

Grupo de Seguridad de las Tecnologías de la Información (SeTI)
Universidad Carlos III de Madrid

Grupo de Tecnologías de las Comunicaciones (GTC)
Universidad de Zaragoza

**ENTIDADES COLABORADORAS**





**ACTAS DEL II SIMPOSIO SOBRE SEGURIDAD INFORMÁTICA (SSI'07)**


Magallanes, 25; 28015 Madrid, ESPAÑA
Teléfono 91 4463350
Fax: 91 4456218
clientes@paraninfo.es

ISBN: 978-84-9732-607-0
Depósito legal: M-

Maquetación: Los Editores
Coordinación del proyecto: @LIBROTEX
Portada: Estudio Dixi
Impresión y encuademación: FER Fotocomposición, S. A.

IMPRESO EN ESPAÑA-PRINTED IN SPAIN

**Presidente**

Arturo Ribagorda Garnacho
Universidad Carlos III de Madrid

**Vicepresidente**

Benjamín Ramos Álvarez
Universidad Carlos III de Madrid

**Secretario**

José Luis Salazar Riaño
Universidad de Zaragoza

**Comité de programa**

| | |
|---|---|
| Abascal Fuentes, Policarpo | Universidad de Oviedo |
| Álvarez Marañón, Gonzalo | CSIC |
| Areitio Bertolín, Javier | Universidad de Deusto |
| Borrell Viader, Joan | Universidad Autónoma de Barcelona |
| Caballero Gil, Pino | Universidad de La Laguna |
| Cabello, Adán | Universidad de Sevilla |
| Curty Alonso, Marcos | Universidad de Zaragoza |
| Dávila Muro, Jorge | Universidad Politécnica de Madrid |
| Domingo-Ferrer, Josep | Universidad Rovira i Virgili |
| Estévez Tapiador, Juan | Universidad Carlos III de Madrid |
| Fernández-Medina Patón, Eduardo | Universidad de Castilla La Mancha |
| Ferrer Gomila, Josep Lluis | Universidad Illes Balears |
| Fúster Sabater, Amparo | CSIC |
| García Teodoro, Pedro | Universidad de Granada |
| Gómez Eskarmeta, Antonio | Universidad de Murcia |
| González-Tablas Ferreres, Ana Isabel | Universidad Carlos III de Madrid |
| González Jiménez, Santos | Universidad de Oviedo |
| González Vasco, María Isabel | Universidad Rey Juan carlos |
| Gutiérrez Gutiérrez, Jaime | Universidad de Cantabria |
| Hernández Castro, Julio César | Universidad Carlos III de Madrid |
| Hernández Encinas, Luis | CSIC |
| Hernández Goya, Candelaria | Universidad de La Laguna |

| | |
|---|---|
| Herrera Joancomartí, Jordi | UOC |
| Huguet Rotger, Llorenç | Universidad Illes Balears |
| López Muñoz, Javier | Universidad de Málaga |
| Malagón Poyato, Chelo | CSIC-RedIris |
| Mañas Argemí, José Antonio | Universidad Politécnica de Madrid |
| Martín del Rey, Ángel | Universidad de Salamanca |
| Melús Moreno, José Luis | Universidad Politécnica de Cataluña |
| Miret Biosca, Josep María | Universidad de Lleida |
| Munuera Gómez, Carlos | Universidad de Valladolid |
| Orfila Díaz-Pabón, Agustín | Universidad Carlos III de Madrid |
| Ortega García, Javier | Universidad Autónoma de Madrid |
| Padró Laimon, Carles | Universidad Politécnica de Cataluña |
| Peinado Domínguez, Alberto | Universidad de Málaga |
| Pérez González, Fernando | Universidad de Vigo |
| Ramió Aguirre, Jorge | Universidad Politécnica de Madrid |
| Ramos Álvarez, Benjamín | Universidad Carlos III de Madrid |
| Ribagorda Garnacho, Arturo | Universidad Carlos III de Madrid |
| Rifá Coma, Josep | Universidad Autónoma de Barcelona |
| Robles Martínez, Sergi | Universidad Autónoma de Barcelona |
| Salazar Riaño, José Luis | Universidad de Zaragoza |
| Sánchez Reíllo, Raúl | Universidad Carlos III de Madrid |
| Sempere Luna, José María | Universidad Politécnica de Valencia |
| Soriano Ibáñez, Miquel | Universidad Politécnica de Cataluña |
| Tena Ayuso, Juan | Universidad de Valladolid |
| Villar Santos, Jorge | Universidad Politécnica de Cataluña |

# Presentación

Hace dos años se celebró en Granada el primer CEDI (Congreso Español de Informática), lo que supuso un hito en nuestro país para las reuniones académicas de las numerosas materias que hoy en día engloba esta macrodisciplina que denominamos informática. Así pues, ésta fue la primera vez que gracias al esfuerzo coordinado de un gran número de profesores e investigadores –atinadamente dirigidos por los organizadores–, se logró reunir en un mismo escenario y en un breve lapso de tiempo a la práctica totalidad de los académicos que nos dedicamos a la informática.

Naturalmente, la seguridad de la información –una de las más pujantes disciplinas de la informática–, no podía estar ausente de este acontecimiento, y por ello celebramos en dicha ocasión el Simposio sobre Seguridad Informática. Esta participación fue una decisión fácil de tomar por parte de los que nos dedicamos a esta disciplina, pues aunque desde el ya lejano 1988 convocábamos una reunión bienal (de nombre Reunión Española de Criptología y Seguridad de la Información, más conocida como RECSI), no podíamos dejar pasar la oportunidad de acogernos al paraguas de CEDI y sumarnos a un Congreso que llamaba a todos nuestros compañeros y amigos de otras disciplinas hermanas.

Además, se da la circunstancia de que CEDI, aun siendo de periodicidad bienal, como RECSI, se reúne los años impares, mientras que la última lo hace los pares, por lo que aquél, aparte de su intrínseco interés, nos ofrecía a los dedicados a la seguridad la posibilidad de seguir manteniendo el contacto entre años impares.

Dado que la iniciativa fue un éxito, y en el 2005 el Simposio sobre Seguridad Informática congregó a un número importante de participantes, parecía indudable que se debía mantener la cita este año en Zaragoza, como así se hizo en su momento, con el resultado que este libro de actas muestra y que los lectores deben de juzgar.

Por lo que atañe a la seguridad de la información, cabe decir que es uno de los campos que ha experimentado en los últimos diez años un crecimiento más vertiginoso, principalmente en todo el llamado primer mundo.

En nuestro país, son tres las principales causas de este hecho. Por un lado, la rápida expansión de Internet, que en poco tiempo ha alcanzado todos los rincones de nuestra sociedad, convirtiéndose en una instrumento ineludible

para las empresas, sin el cual no ya su actividad, sino incluso su presencia entre sus clientes se veía seriamente comprometida. Igualmente, para las Administraciones Públicas Internet supone satisfacer los principios constitucionales de "eficacia, descentralización y coordinación", que, entre otros, deben guiar sus actuaciones, así como atender las demandas de los ciudadanos que requieren servicios públicos ágiles y accesibles.

En segundo lugar, la promulgación en el año 1992 de la Ley Orgánica de Regulación del Tratamiento Automatizado de Datos de Carácter Personal (LORTAD) – derogada en 1999 por la Ley Orgánica de Protección de Datos de Carácter Personal (LOPD)–, y sobre todo la publicación en 1999 del Real Decreto 994/1999 (Reglamento de medidas de seguridad de los ficheros automatizados que contengan datos de carácter personal), supuso un revulsivo para empresas y Administraciones Públicas obligadas, so pena de ser sancionadas por la Agencia Española de Protección de Datos, a adoptar mecanismos y procedimientos de seguridad para proteger los datos personales que manejaban.

En tercer lugar, la presión de los usuarios informáticos, nada proclives a abandonar procedimientos ancestrales de relación con organismos públicos y privados, salvo que se les ofreciesen las mismas garantías de seguridad que se les supone (a menudo con más fe que raciocinio) a los procedimientos que se pretendían sustituir.

Por todo ello, lo que a finales de los ochenta era un pequeño grupo de académicos interesados casi en exclusiva por la criptología, ha devenido en un importante número de investigadores y docentes que, sin abandonar dicho campo, trabajan igualmente en los numerosos temas que figuran entre los de interés de este Simposio, y cuyas aportaciones se presentan en este libro.

Esperamos que estas actas y la reunión del próximo septiembre sirvan para potenciar aún más el interés y desarrollo de la seguridad, sin cuyo concurso difícilmente avanzaremos en la sociedad de la información que estamos conformando y de la que tanto esperamos para el progreso de la humanidad.

# ÍNDICE

## Criptografía



## Autenticación y Biometría

## Sistemas de detección y protección ante intrusos



## Redes P2P y MANET

## Gestión de la Seguridad



## Protocolos y aplicaciones de seguridad

# Puntos de Vista para Patrones de Arquitectura de Seguridad

David G. Rosado[1], Carlos Gutierrez[2], Eduardo Fernández-Medina[1], Mario Piattini[1]
(1) Grupo ALARCOS, Departamento de Tecnologías y Sistemas de Información
Centro Mixto de Investigación y Desarrollo de Software UCLM-Indra
Universidad de Castilla-La Mancha. Paseo de la Universidad, 4 – 13071 Ciudad Real, España
{David.GRosado, Eduardo.Fdez-Medina}@uclm.es
(2) Correos Telecom, Conde de Peñalver, 19 bis 6ª pl. 28006 Madrid, España
carlos.gutierrez@correos.es

## Resumen

La importancia de la arquitectura software está creciendo y aumentando en el campo de la ingeniería del software, en particular en el área de desarrollo de sistemas, debido a que proporciona un control sobre el sistema global, extrayendo los detalles que se creen más importantes y permite centrarse en las interacciones relevantes que se llevan a cabo en el sistema. Junto con la importancia de la arquitectura software, también existe un especial interés en los patrones de seguridad ya que, desde las primeras etapas del ciclo de vida de desarrollo software, proporcionan técnicas para solucionar o detectar los posibles problemas de seguridad que pueden aparecer. En este artículo se pretende concienciar a la comunidad de la importancia de utilizar patrones de seguridad, y cómo esos patrones carecen de una descripción y documentación completa y estructurada para facilitar el entendimiento y comprender la funcionalidad de los patrones para un uso más eficiente en el desarrollo software. Para ello, se pretende dar un modelo de puntos de vista para describir patrones arquitectónicos de seguridad conforme al estándar IEEE 1471-2000 y desarrollaremos un ejemplo que demuestra cómo describir un punto de vista de seguridad siguiendo la plantilla de puntos de vista definida en IEEE 1471 añadiéndole aspectos de seguridad.

## 1. Introducción

En un típico entorno de desarrollo de aplicación, los arquitectos y desarrolladores comparten experiencias similares. Ellos destacan las aplicaciones de negocio en un marco de tiempo altamente comprimido, haciendo aplicaciones que trabajen, comprobando la funcionalidad en todos los niveles, asegurando que conocen el rendimiento o niveles de servicio esperado y envolviendo las aplicaciones con una presentación atractiva para el cliente y una documentación de usuario. Garantizar la seguridad de una aplicación en todos los niveles, normalmente se ha considerado en las últimas fases del proceso de desarrollo [17].

Durante décadas, la comunidad de seguridad ha emprendido una investigación detallada dentro de las áreas específicas de seguridad, mientras se ignora, en gran parte, el proceso de diseño. Los aspectos de seguridad no pueden ser añadidos "ciegamente" dentro de un sistema, pero el desarrollo del sistema global debe tomar en cuenta los aspectos de seguridad. El resultado de un sistema de seguridad bien diseñado debe ser una arquitectura que garantiza aspectos específicos de seguridad tales como la integridad, anonimato y disponibilidad [1].

La Arquitectura software ha emergido como una importante sub-disciplina de la ingeniería del software, particularmente en el área del desarrollo de sistemas. La arquitectura nos proporciona un control intelectual sobre un sistema complejo permitiendo que nos centremos en los componentes esenciales y sus interacciones, más que en detalles extraños [2].

Como hemos visto una y otra vez, la arquitectura software para un sistema juega un papel central en el desarrollo del sistema así como también en la organización que la produce. La arquitectura sirve como el modelo tanto para el sistema como para el proyecto que lo desarrolla. Define las asignaciones de trabajo que deben ser

cumplidas por los equipos de diseño e implementación y es el portador primario de las calidades del sistema tales como rendimiento, capacidad para ser modificada y seguridad; ninguno de los cuales pueden ser alcanzada sin una visón arquitectónica unificada. La arquitectura es un artefacto para que el análisis temprano garantice que la propuesta de diseño producirá un sistema aceptable. En conclusión, la arquitectura es el pegamento conceptual que mantiene cada fase del proyecto unida para todos sus muchos implicados [3]. La arquitectura debe ser documentada y descrita para comunicar cómo alcanzar esas propiedades [2].

Recientemente, ha habido un creciente interés en identificar patrones de seguridad en sistemas software desde que proporcionan técnicas para considerar, detectar y solucionar cuestiones de seguridad desde el inicio de su ciclo de vida de desarrollo [5, 15, 16, 18]. Los patrones de seguridad trabajan conjuntamente para formar una colección de contramedidas de seguridad coordinadas del tal modo que tratan con la seguridad de la aplicación, de la red y del servidor.

Este artículo describe cómo los patrones arquitectónicos de seguridad carecen de una documentación bien estructurada comprensiva y completa que cubren la información esencial de su estructura lógica, tiempo de despliegue, comportamiento en tiempo de ejecución, configuración, restricciones, elementos, etc. En consecuencia, propondremos una forma alternativa para describir patrones arquitectónicos de seguridad desde los distintos puntos de vista y vistas, y así podemos añadir nueva y mayor información a la descripción del patrón. Los puntos de vista y vistas usados son los recomendados por ANSI/IEEE 1471-2000 [9].

El resto del artículo está organizado como sigue: en la sección 2, discutiremos la importancia de las arquitecturas software; En la sección 3, definiremos los patrones de seguridad y lo que son los patrones arquitectónicos de seguridad; En la sección 4, describiremos la plantilla de puntos de vista definida por el Standard IEEE 1471-2000, definiendo un punto de vista concreto, el punto de vista del subsistema de diseño de seguridad. Finalmente propondremos nuestras conclusiones y el trabajo futuro.

## 2. Arquitectura Software

La Arquitectura software ha emergido como una importante sub-disciplina de la ingeniería del software, particularmente en el área del desarrollo de sistemas. Hay muchas definiciones de arquitectura software [3, 8], pero lo que estas definiciones tienen en común es su énfasis en la arquitectura como una descripción de un sistema, como una suma de pequeñas partes y cómo esas partes se relacionan y cooperan unas con otras para desempeñar el trabajo del sistema. La arquitectura nos aporta el control sobre un sistema complejo permitiéndonos sustituir lo complejo por un conjunto de piezas que interactúan, cada una de las cuales es substancialmente más simple que el conjunto [2]. Estas definiciones pueden ser extendidas al entorno de la seguridad, definiendo una arquitectura de seguridad como una descripción de un sistema compuesto por distintas partes y/o elementos de seguridad que cooperan y se relacionan para desarrollar un sistema seguro.

Desde una perspectiva técnica, hay fundamentalmente tres razones de la importancia de la arquitectura software: i) *Comunicación entre stakeholders:* La arquitectura software representa una abstracción común de un sistema que la mayoría de los *stakeholders* del sistema pueden usar como base para una comprensión mutua, negociación, consenso y comunicación; ii) *Decisiones tempranas de diseño:* La arquitectura software manifiesta las decisiones tempranas de diseño sobre un sistema, y estas tempranas vinculaciones llevan el peso lejos de la proporción a su gravedad individual con respecto al desarrollo restante del sistema, su despliegue, y su vida de mantenimiento; y iii) Abstracción transferible de un sistema: La arquitectura constituye un pequeño y comprensible modelo de cómo un sistema está estructurado y cómo sus elementos trabajan juntos, y cómo este modelo es transferible a través de sistemas

En una arquitectura de software se describen los detalles de diseño de una colección de componentes y sus interconexiones, que conforman una vista abstracta a alto nivel del sistema que se está diseñando, y donde se consideran los requisitos identificados en la fase de análisis de requisitos del sistema. Actualmente,

en la comunidad de arquitecturas de software existe una gran variedad de elementos arquitectónicos que simplifican las tareas de diseño en la construcción de una arquitectura. Estos elementos arquitectónicos se conocen con el nombre de “estilos arquitectónicos".

### 2.1. Estilos arquitectónicos

Un estilo de arquitectura en software consta de características claves y reglas que se combinan para preservar la integridad.

Un estilo de arquitectura software esta determinado por lo siguiente:

1. Conjunto de tipos de componentes (repositorio de datos, un proceso, un procedimiento) que lleva a cabo alguna función en tiempo de ejecución.
2. Capas topológicas de esos componentes indicando sus interrelaciones en tiempo de ejecución.
3. Conjunto de restricciones semánticas (por ejemplo, en un repositorio de datos no está permitido cambiar los valores almacenados en él).
4. Conjunto de conectores (por ejemplo, llamada a subrutina, llamada a procedimiento remoto, flujo de datos, sockets) que median la comunicación, coordinación o cooperación entre componentes.

Los estilos se pueden dividir en varias familias, entre ellas los estilos centrados en datos (repositorios, las bases de datos, las arquitecturas basadas en hipertextos y las arquitecturas de pizarra), los estilos de flujo de datos (tubería-filtros y las de proceso secuencial en lote), los estilos de llamada y retorno (arquitecturas de programa principal y subrutina, los sistemas basados en llamadas a procedimientos remotos, los sistemas orientados a objeto y los sistemas jerárquicos en capas), los estilos de código móvil (arquitectura de máquinas virtuales)y los estilos peer-to-peer (arquitecturas basada en eventos, orientadas a servicios, basadas en recursos).

## 3. Patrones de seguridad

Los patrones de seguridad proporcionan técnicas para identificar y solucionar cuestiones de seguridad, trabajan juntos para formar una colección de mejores prácticas (o soportar una estrategia de seguridad) y se dirigen al servidor, a la red y a la seguridad de la aplicación. Los beneficios de usar patrones de seguridad son: pueden ser revisados e implementados en cualquier momento para mejorar el diseño de un sistema; los menos expertos se pueden beneficiar de la experiencia de los más influyentes en patrones de seguridad; proporciona un lenguaje común para argumentar, probar y desarrollar; puede ser fácilmente registrado, clasificado y refactorizado; proporciona reutilidad, se puede repetir y documentar prácticas seguras; no definen estilos codificados, lenguajes de programación o proveedores [4].

Un patrón arquitectural es una descripción de tipos de elementos y relaciones junto con un conjunto de restricciones de cómo pueden ser usados. Es una abstracción de alto nivel. La elección del patrón arquitectural a ser usado es una decisión de diseño fundamental en el desarrollo de un sistema software. Determina la estructura del sistema y obliga las elecciones disponibles de diseño para los diversos subsistemas. Es, en general, independiente del lenguaje de implementación a ser usado. Un patrón puede ser pensado como un conjunto de restricciones en una arquitectura, como tipos de elementos y sus patrones de interacción, y esas restricciones definen un conjunto o familia de arquitecturas que las satisfacen.

Un patrón de diseño proporciona un esquema para refinar los subsistemas o componentes de un sistema software, o las relaciones entre ellos. Describe una estructura comúnmente recurrente de componentes de comunicación que solucionan un problema de diseño general dentro de un contexto particular. Un patrón de diseño es una abstracción de medio nivel.

Las estrategias de diseño determinan que tácticas de aplicación o patrones de diseño deben ser usados para escenarios y restricciones particulares de seguridad de la aplicación. Los patrones de diseño de seguridad son una abstracción de problemas de negocio que dirigen una variedad de requisitos de seguridad y proporcionan una solución a los conocidos problemas de seguridad relacionados. Pueden ser patrones arquitecturales que describen cómo un problema de seguridad puede ser resuelto arquitecturalmente, o pueden ser estrategias de

diseño defensivas sobre cuyo código seguro puede ser construido más adelante [17].

## 4. Puntos de vista para los patrones de seguridad

Los patrones de seguridad y las arquitecturas deben ser descritos para comunicar a los interesados (ingenieros, proveedores y desarrolladores de seguridad, directores, examinadores, mantenimiento, etc.) cómo alcanzar la seguridad en el sistema, sirviendo como vehículo de comunicación entre los interesados y como base para el análisis del sistema de seguridad.

Definimos patrones arquitectónicos de seguridad desde diferentes perspectivas, dependiendo de los distintos interesados potenciales que vayan a usarlo, viendo cada uno de ellos las características, funcionalidades, restricciones, comportamiento, etc. sobre el patrón que más le interesa y pueda servirle para la incorporación de dicho patrón o arquitectura en el sistema global, dejando la información no relevante a un lado.

Estamos definiendo una librería de puntos de vista de seguridad que nos permite describir patrones arquitectónicos de seguridad siguiendo el Standard IEEE 1471-2000. Por definición, estos puntos de vistas son reutilizables para cualquier sistema software, así podemos ampliar dicha propuesta para describir patrones de seguridad, arquitecturas de seguridad, arquitecturas software, patrones de diseño, etc., basadas en nuestra librería de puntos de vista.

Existe un número de catálogos de puntos de vista, pero hemos observado que ninguno de ellos considera los aspectos de seguridad, sólo se aplican al desarrollo de arquitecturas de los sistemas de información y no son aplicables en el contexto de la seguridad. Por este motivo, hemos considerado un conjunto de puntos de vista para los arquitectos de seguridad, para los ingenieros de seguridad, que aumenta y extiende el modelo "4+1", basándonos en los trabajos de Philippe Kruchten [12], Nick Rozanski y Woods [14]. Nuestro catálogo de puntos de vista contiene siete puntos de vista de seguridad: Lógico, Proceso, Desarrollo, Físico, Despliegue, Operacional y Casos de Mal uso y casos de uso de seguridad, que podemos ver en la Figura 1.

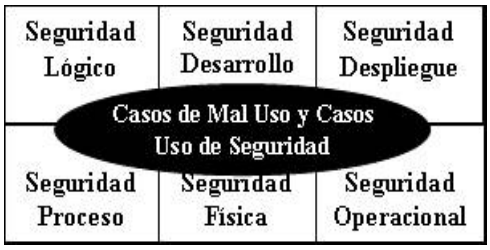


Figura 1. Propuesta de Puntos de Vista de Seguridad

El punto de vista de seguridad *Lógico* describe los objetos o modelos de objetos dentro de la arquitectura de seguridad que da soporte a los requisitos de seguridad del comportamiento. El punto de vista de seguridad de *Proceso* describe la arquitectura de seguridad como una red lógica de procesos seguros comunicándose. Este punto de vista asigna a un objeto del modelo de objetos a una hebra de ejecución y captura los aspectos de concurrencia y sincronización del diseño de seguridad. El punto de vista *Físico* de seguridad enlaza software con hardware y elementos de red, y refleja el aspecto distribuido de la arquitectura software de seguridad. El punto de vista de seguridad de *Desarrollo* se centra en la organización estática del software en el entorno de desarrollo de seguridad y trata con cuestiones del manejo de la configuración, tareas del desarrollo de seguridad, responsabilidades de seguridad y contramedidas. El punto de vista de seguridad de *Despliegue* describe el entorno de seguridad dentro del cual el sistema será desplegado, incluyendo el hecho de capturar las dependencias que el sistema tiene en el entorno de ejecución. El propósito del punto de vista de seguridad *Operacional* es identificar las estrategias de seguridad del sistema para dirigir las inquietudes operacionales de los interesados en el sistema e identificar soluciones que las dirijan. El punto de vista de los *Casos de Mal uso y Casos de uso* de Seguridad define una secuencia completa de acciones, desencadenadas por un "mal" actor (un actor para el cual el sistema no debe dar soporte funcional alguno). Los casos de mal uso representan escenarios de ataques a las vulnerabilidades identificadas sobre ciertos

recursos. Los casos de uso de seguridad definen el concepto de casos de uso de seguridad como herramienta para el análisis de los requisitos de seguridad de un sistema software. Un caso de uso de seguridad define los requisitos de seguridad que cierto sistema debería garantizar protegiéndose así mismo de las amenazas de seguridad relevantes.

### 4.1. Plantilla para definir puntos de vista

El Standard ANSI/IEEE 1471-2000 [9] proporciona guías para elegir el mejor conjunto de vistas y puntos de vista para documentar y describir arquitecturas, satisfaciendo a la comunidad de interesados. Para describir los puntos de vista y vistas, IEEE 1471 define un conjunto de elementos o secciones (plantilla) que se compone de los siguientes elementos: i) Resumen; ii) Interesados y preocupaciones resueltas; iii) Elementos, Relaciones, Propiedades y Restricciones; iv) Lenguaje(s) para Modelar/Representar Vistas Conformes; v) Técnicas de Análisis/Evaluación y Criterio de Consistencia/Completitud; vi) Origen del Punto de Vista.

Esta plantilla sirve para describir arquitecturas o patrones arquitectónicos tanto con aspectos de seguridad como sin ellos, es una plantilla genérica para cualquier tipo de arquitectura o patrón, es decir, es reutilizable para cualquier sistema software. Nosotros, para ampliar dicha plantilla adaptándola al contexto de la seguridad y haciéndola específica de seguridad, le añadimos nuevas secciones relacionadas con la seguridad, creando de esta forma una plantilla de puntos de vista y vista para describir arquitecturas de seguridad; las nuevas secciones de esta plantilla, en el contexto de la seguridad, son las siguientes:

1. Elementos a incluir en la política de seguridad en base al punto de vista de seguridad. Consideramos que la política de seguridad completa de un patrón de seguridad es la agregación de las políticas de seguridad definidas para cada punto de vista de seguridad.
2. Métricas de seguridad a tener en cuenta en ese punto de vista.
3. Procedimientos de seguridad a tener en cuenta desde el punto de vista considerado; por ejemplo, desde el punto de vista físico, procedimientos para restaurar el nodo físico en el que corre cierto servicio de seguridad definido por el patrón o desde el punto de vista lógico, como llevar a cabo el intercambio de claves off-line entre las partes.
4. Buenas prácticas: por ejemplo desde el punto de vista del desarrollador, técnicas de programación que prevengan código inseguro, o desde el punto de vista físico, topologías de redes seguras.

## 5. Ejemplo de punto de vista: Subsistema de Diseño de Seguridad

La vista lógica de seguridad incluye un paquete de vistas con la información de descomposición en subsistemas de diseño de la seguridad. La información de una vista se distribuye entre los 'stakeholders' interesados en esa vista (y designados por el punto de vista con la que es conforme) en forma de paquetes de vista. Un paquete de vistas para los desarrolladores y gestor de proyectos (con el objeto de que pueda asignar cada subsistema a los equipos de trabajo) puede incluir este punto de descomposición en subsistemas de seguridad lógicos. De esta forma, podemos definir, siguiendo la plantilla definida por el Standard IEEE 1471 vista anteriormente, este punto de vista como sigue:

- **Resumen**: Este punto de vista muestra la descomposición modular, y el uso entre sistemas, del sistema software. Cada módulo se interpreta como un subsistema a desarrollar, por tanto es una entidad en tiempo de construcción, que puede interactuar con otros subsistemas para completar su funcionalidad. Se parte de un módulo definido en una Vista Contextual y se muestra su descomposición en subsistemas. La descomposición sigue hasta que cada módulo, o subsistema, es asignable a un equipo o responsable de desarrollo único. A este nivel, los módulos se denominan subsistemas de diseño. Además, y en cada paso de descomposición, se debe presentar cómo estos subsistemas interactúan entre sí para completar las preocupaciones de los interesados.

- **Interesados y preocupaciones resueltas**: Las aplicaciones de seguridad serán desarrolladas al menos por tres diferentes interesados: i) desarrolladores de la aplicación software que se centran en la lógica de negocio; ii) Proveedores de Seguridad que se centran en el diseño e implementación de marcos de trabajo reutilizables de la seguridad lógica; iii) Ingenieros de Seguridad que implementan la política de seguridad para una aplicación particular y se centra en cómo el sistema es implementado desde la perspectiva de la seguridad, y cómo la seguridad afecta a las propiedades del sistema.
  También están implicados:
  1. Gestor de proyectos, que deben definir la asignación del trabajo, formar equipos, y formular planes de proyectos, presupuestos y horarios.
  2. Mantenedores, cuya tarea es modificar los elementos software.
  3. Probadores e Integradores, que utilizan los módulos como su unidad de trabajo;

- **Elementos, Relaciones, Propiedades y Restricciones**: En este apartado se definen los módulos, como unidades de implementación, y su descomposición en módulos más pequeños así como dependencia de uso que existe entre sí. Las relaciones entre los módulos puede tener la semántica 'es parte de' o 'utiliza' asociada. El último nivel de subsistemas, aquellos denominado como subsistemas de diseño, definidos en las vistas conformes a este punto de vista deben:
  1. Ser el conjunto de productos de trabajo de diseño asignables a los diferentes equipos de desarrollo.
  2. Los subsistemas se correlacionarán con los directorios de construcción que serán desarrollados, probados y entregados por los equipos de desarrollo respectivos.
  3. Siguiendo los principios de modularidad, los subsistemas debe exhibir alta cohesión y bajo acoplamiento.
  4. Estos subsistemas serán las entidades de más bajo nivel para las cuales el equipo de arquitectos software necesitarán definir las interfaces. Las partes internas de cada subsistema deberá ser diseña por el Equipo de Diseño que deberá especificar la colaboración entre los elementos contenidos en el subsistema que implementa la interfaz declarada por éste.
  5. Se pueden asignar múltiples subsistemas a un mismo equipo de desarrollo, pero cada subsistema será desarrollado, probado y versionado de manera independiente.
  6. Cada subsistema se puede considerar como un sistema a diseñar por el equipo de diseño al que ha sido asignado.

- **Lenguaje(s) para Modelar/Representar Vistas Conformes**: El lenguaje de representación a utilizar será UML y extensiones para los aspectos de seguridad tales como *UMLSec* [10, 11] y *SecureUML* [13]. Cada módulo o subsistema se representará como un paquete UML estereotipado con la palabra reservada <<subsystem>>. Las relaciones de uso se mostrarán como relaciones de dependencia UML que incluyen el estereotipo <<uses>> y las de descomposición mediante anidamiento de paquetes UML. Las interfaces que implementan cada sistema se modelan como interfaces UML y el nombre de los servicios a incluir en cada interfaz se corresponde con los nombres de los casos de uso definidos en el nivel de abstracción de "*Meta Resumen*" [6] para cada subsistema. Los subsistemas de diseño incluidas en las vistas conformes con este punto de vista declararán una realización de una (o más) interfaces cuyos métodos se corresponden con los casos de uso en el nivel de abstracción "*Meta de Usuario*" especificados en el modelo de casos de uso de ese subsistema de diseño.

- **Técnicas de Análisis/Evaluación y Criterio de Consistencia/Completitud**: Reuniones de revisión con los diferentes grupos de desarrollo de forma que comprendan el contexto del subsistema que van a desarrollar (de qué sistema proviene) así como las interacciones con otros subsistemas de diseño. Algunos métodos de análisis y evaluación son descritos por Wassermann [5] y Jürjens [7].

- **Origen del Punto de Vista**: Punto de Vista de Subsistema de Diseño [8].

Estamos trabajando en ampliar dicha descripción del punto de vista del subsistema de diseño de seguridad añadiéndole los aspectos de seguridad que comentamos anteriormente (métricas, procedimientos de seguridad, etc.), ampliando dicha plantilla para describir y documentar patrones de seguridad.

## 6. Conclusiones

Los arquitectos toman decisiones de diseño muy temprano en el ciclo de vida del proyecto. Muchos de ellas son difíciles, si no imposibles, validar y probar partes del sistema que están actualmente construidas. Debido a la dificultad de validar decisiones de diseño muy temprano, los arquitectos confían sensiblemente en los métodos probados y ensayados para solucionar ciertas clases de problemas. Esto es uno de los grandes valores de los patrones arquitectónicos. Permiten a los arquitectos reducir el riesgo con diseños apropiados con atributos de ingeniería conocidos.

Los patrones de seguridad ayudan a no perder de vista los requisitos no-funcionales de seguridad al inicio del diseño. En las aplicaciones críticas de seguridad es extremadamente importante evitar errores, ya que se debe garantizar la seguridad de dichas aplicaciones y otorgar un alto nivel de seguridad a todas las operaciones e interacciones que se hagan en la aplicación. Por tanto, el uso de los patrones de seguridad es importante para desarrollar un sistema seguro.

Es importante describir una arquitectura software porque, en primer lugar, sirve para introducir y presentar a la gente al sistema; en segundo lugar, sirve como vehículo para comunicarse entre los interesados, y finalmente, es usada como una base para el análisis de sistemas. Además, una arquitectura documentada y bien descrita es fundamental para comprender sus principales características, su funcionalidad, sus componentes y conexiones, su comportamiento, etc. Será importante describir y definir las principales características de los patrones arquitectónicos para que los interesados sean capaces de usar y analizar el patrón al tiempo de integrarlo tanto en el diseño de la aplicación como en el diseño de la arquitectura global.

En este artículo, hemos añadido algunas nuevas secciones a la plantilla existente usada para describir patrones arquitectónicos de seguridad desde distintos puntos de vistas, añadiendo aspectos de seguridad. La adopción de IEEE 1471 y la próxima liberación de UML 2.0, UMLSec [11] y SecureUML [13] deberían ayudar a mejorar las futuras prácticas de la arquitectura software de seguridad.

Nuestra intención no es sólo definir patrones arquitectónicos de seguridad por medio de plantilla de vistas y puntos de vista, sino también recomendar ANSI/IEEE 1471-2000 [9], que proporciona guías para elegir el mejor conjunto de vistas a documentar. Hemos definido un catálogo de puntos de vista y hemos añadido y estamos añadiendo nuevos elementos o secciones a la plantilla de puntos de vista de IEEE 1471-2000. Crearemos un catálogo completo de vistas y puntos de vista para patrones arquitectónicos de seguridad conforme al Standard IEEE 1471.

## Agradecimientos

Esta artículo ha sido desarrollado en el contexto de los proyectos DIMENSIONS (PBC-05-012-2) y MISTICO (PBC-06-0082) financiados parcialmente por FEDER y por la "Consejería de Ciencia y Tecnología de la Junta de Comunidades de Castilla-La Mancha" (España), RETISTRUST (TIN2006-26885-E) y ESFINGE (TIN2006-15175-CO5-05) del "Ministerio de Educación y Ciencia" (España).

## Referencias

[1] Artelsmair, C. and R. Wagner. *Towards a Security Engineering Process*. in The 7th World Multiconference on Systemics, Cybernetics and Informatics. 2003. Orlando, Florida, USA.

[2] Bachmann, F., L. Bass, J. Carriere, P. Clements, D. Garlan, J. Ivers, R. Little, and R. Nord, *Software Architecture Documentation in Practice: Documenting Architectural Layers CMU/SEI-2000-SR-004*. 2000. p. 46

[3] Bass, L., P. Clements, and R. Kazman, eds. *Software Architecture in Practice*. 2003, Addison-Wesley.

[4] Berry, C.A., J. Carnell, M.B. Juric, M.M. Kunnumpurath, N. Nashi, and S. Romanosky, *Chapter 5: Patterns Applied to Manage Security*, in *J2EE Design Patterns Applied*. 2002.

[5] Cheng, B.H.C., S. Konrad, L.A. Campbell, and R. Wassermann. *Using Security Patterns to Model and Analyze Security Requirements*. 2003. Monterey Bay, CA, USA. p. 13-22

[6] CockBurn, A., *Writing Effective Use Cases*. 2000: Addison-Wesley Professional. 270.

[7] Deubler, M., J. GrÃ¼nbauer, J. JÃ¼rjens, and G. Wimmel. *Sound Development of Secure Service-based Systems*. 2004. New York City, USA: ACM Press.

[8] Garlan, J. and R. Anthony, *Large-Scale Software Architecture*. 2002: John Wiley & Sons 278.

[9] IEEE, *Recommended Practice for Architectural Description of Software-Intensive Systems (IEEE Std 1471-2000)*. . 2000, Institute of Electrical and Electronics Engineers: New York, NY. p. 29

[10] Jurjens, J. *Towards Secure Systems Development with UMLsec*. in International Conference of Fundamental Approaches to Software Engineering (FASE/ETAPS). 2001. Genoa, Italy: Springer-Verlag.

[11] Jurjens, J. *UMLsec: Extending UML for Secure Systems Development*. in 5th International Conference on the Unified Modeling Language (UML). 2002. Dresden, Germany: Springer.

[12] Kruchten, P., *Architectural Blueprints - The "4+1" View Model of Software Architecture.* IEEE Software, 1995. **12**(6): p. 42-50.

[13] Lodderstedt, T., D. Basin, and J.r. Doser. *SecureUML: A UML-Based Modeling Language for Model-Driven Security*. 2002. Dresden, Germany: Springer. p. 426--441

[14] Rozanski, N. and E.i. Woods, *Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives*. 2005: Addison Wesley Professional. 576.

[15] Schumacher, M., E.B. Fernandez, D. Hybertson, and F. Buschmann, *Security Patterns*. 2005: John Wiley & Sons. 512.

[16] Schumacher, M. and U. Roedig. *Security Engineering with Patterns*. in 8th Conference on Patterns Lnaguages of Programs, PLoP 2001. 2001. Monticello, Illinois, USA.

[17] Steel, C., R. Nagappan, and R. Lai, *Core Security Patterns*. 2005: Prentice Hall PTR. 1088.

[18] Yoder, J. and J. Barcalow. *Architectural Patterns for Enabling Application Security*. 1997. Monticello, Illinois, USA.