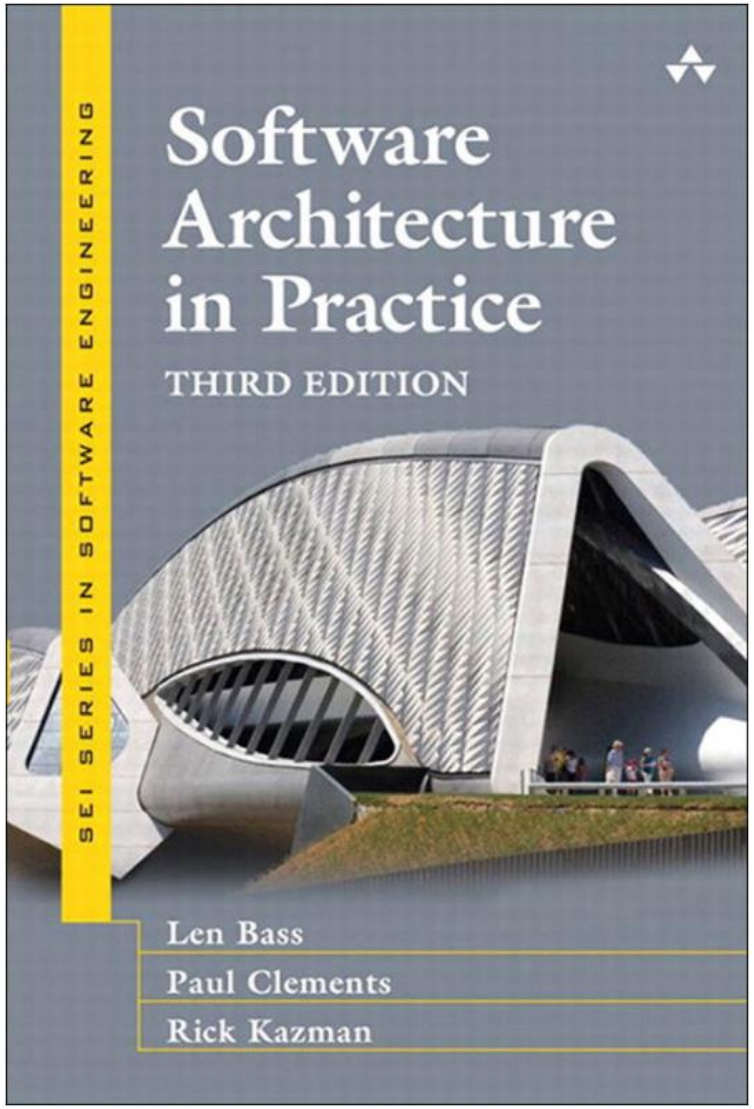
Agenda

Atributos de Calidad en las Aplicaciones
Servicios Web
SOAP
REST
REST vs SOAP
Práctica Grupal
Práctica en Equipo

Atributos de Calidad en las Aplicaciones



Bibliografía Sugerida

Software Architecture in Practice 3rd Edition
Len Bass, Paul Clements, Rick Kazman
Addison-Wesley Professional
2013

Parte de la Certificación “SEI Software Architecture Professional Certificate”
http://www.sei.cmu.edu/training/certificates/architecture/professional.cfm

Atributos de Calidad en las Aplicaciones

Atributo de Calidad: Propiedad medible o sujeta a pruebas, de un sistema, que es utilizada para determinar el grado de satisfacción de las necesidades de los stakeholders en dicho sistema.

- Disponibilidad
- Tolerancia al Cambio
- Desempeño
- Seguridad
- Usabilidad
- Integridad Física del Usuario (Safety)

Disponibilidad

Funcionalidad del sistema que se encuentra lista para llevar a cabo su propósito justo en el momento en que el usuario la requiere.

Disponibilidad + Recuperación = Confiabilidad

Ping
Monitoreo de Sw y Hw
Redundancia de Sw y Hw
Detección de Excepciones (Timeouts)
DRP (Disaster Recovery Plan)

Tolerancia al Cambio

En cualquier tipo de software siempre está presente el cambio.

La mayor cantidad de cambios pueden ocurrir después de que el sistema es terminado y entregado al cliente:

- Nuevos requisitos.
- Corrección de defectos.
- Incrementar la seguridad.
- Mejorar el desempeño.
- Mejorar la usabilidad.
- Utilizar nuevas tecnologías, plataformas, protocolos, etc.

Encapsular funcionalidad.
Ocultar las implementaciones y utilizar interfaces.
Restringir dependencias entre los módulos.
Re-factorización / Abstraer servicios comunes: Si dos módulos proveen servicios similares, implementarlos una sola vez de manera general, de tal forma que solo se tenga un punto de cambio.

Desempeño

¿En qué tiempo me entregará el sistema el resultado que requiero? ¿Es el mejor tiempo en base al ambiente en el que se encuentra el sistema (limitaciones de hw, sw, base de datos, red, etc.)?

Elementos a considerar:

- Concurrencia: Ejecución de operaciones en paralelo.
- Latencia: Tiempo entre la llegada de una solicitud y la entrega de la respuesta.
- Rendimiento: Cantidad de transacciones que el sistema puede procesar en un tiempo determinado.

Incrementar recursos: Procesadores más rápidos, RAM, redes más rápidas, HDD más rápido, migrar de tecnología de desarrollo, etc.
Incrementar la eficiencia: ¿Se puede mejorar ‘X’ algoritmo, ‘Y’ módulo? ¿Se puede mejorar el diseño de la base de datos?
Uso de cachés.
Calendarización de procesamientos masivos de alto consumo de recursos, como procesador, disco duro y red.

Seguridad

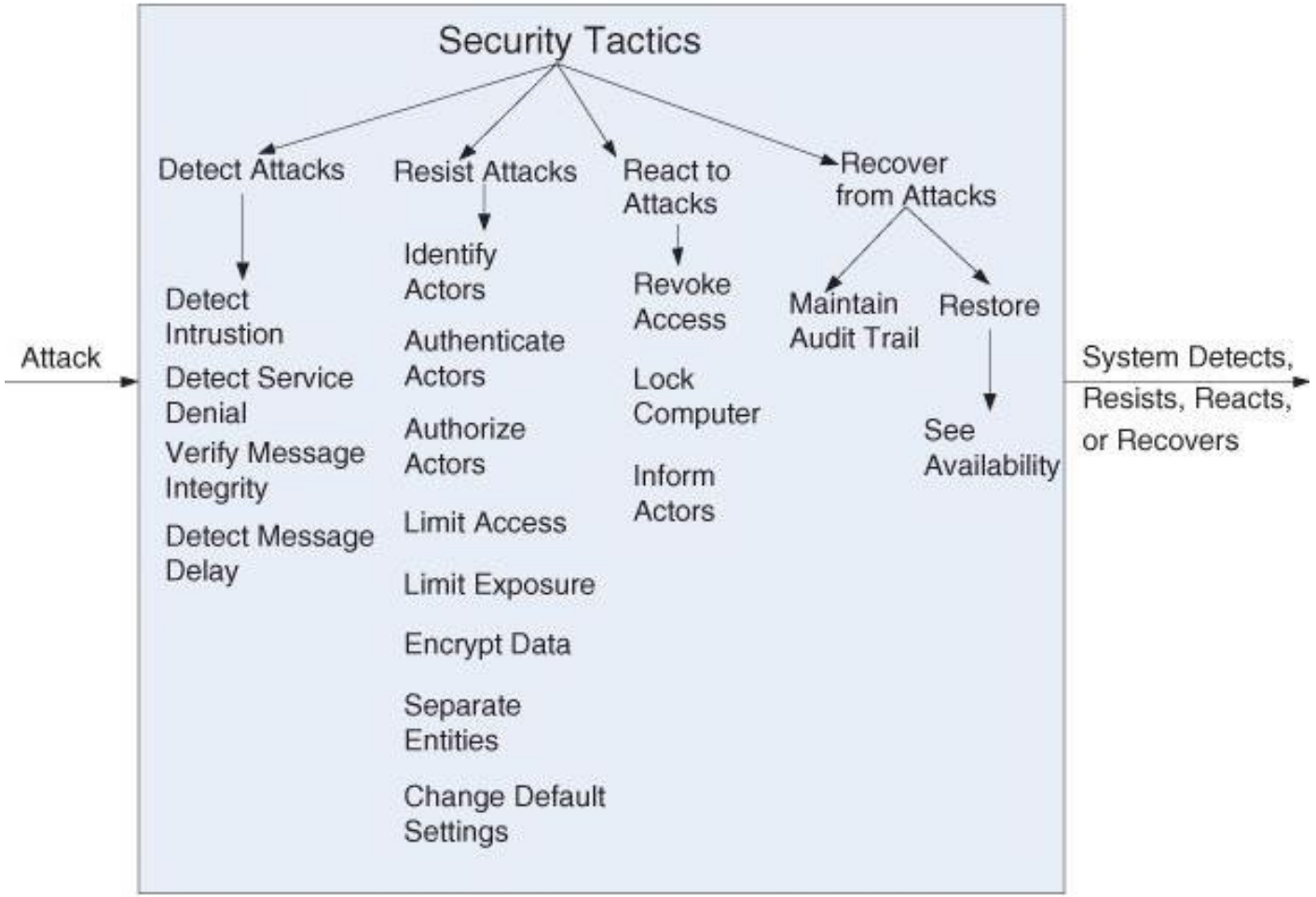
Medida de la capacidad de un sistema de proteger datos e información de accesos no autorizados, mientras, que al mismo tiempo, provee accesos a personas y sistemas que están autorizados.

- El triángulo de la seguridad (CIA):
- Confidencialidad: Los servicios y datos están protegidos contra accesos no autorizados.
- Integridad: Los servicios y datos están protegidos contra manipulaciones no autorizadas.
- Disponibilidad (Availability): Los servicios y datos podrán ser accedidos cuando se requiera.

- Autenticación: Verificar que eres quien dices ser.
- No repudio: La entidad que envía un mensaje no puede negar que envió el mensaje, y la entidad que recibió el mensaje no puede negar que recibió el mensaje (logs de las aplicaciones protegidos).
- Autorización: Privilegios de un usuario para llevar a cabo sus funciones.

Mecanismos de detección de intrusos: Listas de acceso, alertas.
Detección de ataques de negación de servicio, monitoreando el tráfico de la red con determinadas reglas.
Cambiar configuraciones por default.
Verificar integridad de la información: Uso de algoritmos Hash.
Limitar accesos.
Cifrado de información.
Llevar a cabo auditorías a los mecanismos de seguridad.

Seguridad


Security Tactics
Detect Attacks
Resist Attacks
React to Attacks
Recover from Attacks
Detect Intrustion
Detect Service Denial
Verify Message Integrity
Detect Message Delay
Identify Actors
Authenticate Actors
Authorize Actors
Limit Access
Limit Exposure
Encrypt Data
Separate Entities
Change Default Settings
Revoke Access
Lock Computer
Inform Actors
Maintain Audit Trail
Restore
See Availability
Attack
System Detects, Resists, Reacts, or Recovers

Usabilidad

Qué tan fácil es para el usuario llevar a cabo las funciones del sistema, y la manera en que el sistema da soporte al usuario. Mejora la percepción de la calidad del software.

- ¿De qué manera contribuye el sistema en facilitar el aprendizaje de su uso?
- ¿De qué manera contribuye el sistema en hacer más eficiente las tareas del usuario?
- ¿Qué puede hacer el sistema para minimizar el impacto de un error del usuario?

Cancelar una operación.
Deshacer una operación.
Pausar / resumir procesamientos que toman mucho tiempo.
Homogenización de pantallas.
Llevar a cabo las tareas con la menor cantidad de clics, taps, y comandos.
Tener disponible mecanismos de ayuda en todo momento.

Integridad Física del Usuario

A medida que el software comienza a controlar cada vez más dispositivos que utilizamos en nuestra vida diaria, éste atributo de calidad toma más importancia.

Habilidad del software para prevenir que entre en un estado que ocasione daños, lesiones, o la muerte.

- Control de ventiladores de una mina.
- Datos que asisten el despegue y aterrizaje de los aviones.
- Autos que se manejan solos.
- Casas inteligentes (instalaciones eléctricas y de gas).

- IOT: Internet de las Cosas

Servicios Web

Servicios Web

- Mecanismo de comunicación entre sistemas que se encuentran implementados con diferentes plataformas.

- Una aplicación escrita en .NET y que corre en Windows puede comunicarse con una aplicación escrita en Java y que corre en Linux.

- Beneficios:

    - Reusabilidad: Construir una sola vez el procesamiento y consumo de información que puede ser utilizado para varias aplicaciones.

    - Interoperabilidad: Permite la interacción de sistemas sin importar sobre qué plataforma se construyeron y están siendo ejecutados.

    - Bajo Acoplamiento: Se pueden modificar servicios web sin impactar otros servicios o aplicaciones que no estén relacionados.

    - Mantenimiento: Se puede dar mantenimiento a un servicio web y liberar cambios en la lógica de negocio sin impactar a los servicios y aplicaciones que hagan uso de él, siempre y cuando no se altere su interfaz.

    - Facilidad de Integración: Permite intercambiar la información de las aplicaciones, invirtiendo poco tiempo para la construcción de los servicios y sus clientes.

SOAP

- Simple Object Access Protocol.
- Protocolo basado en XML que puede ser utilizado por una aplicación para operar con otros sistemas e intercambiar información.
- Se generan mensajes XML para la comunicación.
- Se utiliza principalmente el protocolo HTTP para la transmisión de los mensajes, pero también se pueden utilizar otros medios de transmisión.
- No impone los nombres de servicios ni nombres de los métodos de los servicios. Éstos se pueden llamar de cualquier forma, siguiendo las reglas para la declaración de objetos y métodos del lenguaje en el que se escribe el servicio web.
- El cliente y el servidor pueden estar escritos en cualquier lenguaje de programación y ser ejecutados en cualquier plataforma, siempre y cuando se apeguen al protocolo para procesar los mensajes XML generados.

SOAP

- Para que un cliente se pueda comunicar con el servicio web de un servidor, es necesario conocer los siguientes detalles técnicos sobre dicho servicio:

  - Ubicación del servicio web.
  - Métodos disponibles, la firma de éstos métodos y sus tipos de retorno.
  - Protocolo de comunicación.

- WSDL: Web Service Description Language. Es un archivo XML que describe los detalles técnicos de la implementación del servicio web: URI, puertos, nombres de los métodos, argumentos, tipos de datos, etc. Es generado por el proveedor del servicio y está disponible a los clientes que requieren consumirlo.

- Ejemplo de WSDL: http://localhost:8080/WebServiceDummy/DummyWS?wsdl

SOAP

- Creación de Web Services:
  - Bottom Up: Primero se escribe el código del web service, por ejemplo una clase de Java, y por medio de una herramienta se genera el WSDL del servicio.
  - Top Down: Primero se genera el WSDL del servicio, y por medio de una herramienta se genera el código del servicio web.

REST

- REpresentational State Transfer: Transferir el estado de la representación de un recurso. La representación del recurso puede ser un JSON.

- Arquitectura de servicios web para llevar a cabo operaciones CRUD sobre recursos, representadas por las operaciones HTTP POST, GET, PUT, y DELETE (y otros métodos HTTP), mediante el uso de URI.

- Cualquier cliente HTTP puede establecer comunicación con un servidor HTTP utilizando REST, sin que la configuración del cliente o del servidor sea compleja.

REST Vs SOAP

- SOAP es un protocolo basado en el intercambio de información por medio de XML. Define estándares de comunicación, que deben ser implementados por los clientes y los servidores.

- REST es una arquitectura estructurada alrededor del protocolo HTTP.

- Un mensaje de REST contiene menos caracteres que un mensaje de SOAP. Esto es importante para sistemas que tienen un alto volumen de intercambio de mensajes.

```
http://www.XYZdirectory.com/phonebook/UserInfo/99999
```

```
<?xml version="1.0"?>
<soap:Envelope xmlns:soap=http://www.w3.org/2001/
    12/soap-envelope
 soap:encodingStyle="http://www.w3.org/2001/12/
    soap-encoding">
  <soap:Body pb="http://www.XYZdirectory.com/
    phonebook">
    <pb:GetUserInfo>
       <pb:UserIdentifier>99999</pb:UserIdentifier>
    </pb:GetUserInfo>
  </soap:Body>
</soap:Envelope>
```

- Con las herramientas adecuadas, cada enfoque es relativamente sencillo de implementar.

REST Vs SOAP

Recomendación

- Se recomienda utilizar SOAP para aplicaciones empresariales que requieran un alto nivel de calidad de servicio. Se cuenta con las implementaciones del conjunto de protocolos WS-* (https://wiki.apache.org/ws/WebServiceSpecifications).

- Se recomienda utilizar REST para la implementación rápida y ágil de aplicaciones web con bajo acoplamiento, escalabilidad, y arquitectura simple. Es más fácil la creación de clientes que consuman servicios REST.

Práctica Grupal

Servicios Web REST con NodeJS

Práctica en Equipo

Servicios Web REST con NodeJS