Atributos de Calidad / Tácticas

Ing. Nicolás Passerini
Ing. Gustavo Andrés Brey
Gastón Coco

Agenda

#	Tema	Duración
1	Introducción	10 min
4	Atributos de Calidad / Tácticas	
4.1	Performance	15 min
4.2	Disponibilidad	15 min
4.3	Modifiability	15 min
4.4	Seguridad	10 min
4.5	Testability	10 min
4.6	Usabilidad	10 min
5	Restricciones de Arquitectura	15 min
6	Comentarios Finales	5 min

Agenda

#	Tema	Duración
1	Introducción	10 min
4	Atributos de Calidad / Tácticas	
4.1	Performance	15 min
4.2	Disponibilidad	15 min
4.3	Modifiability	15 min
4.4	Seguridad	10 min
4.5	Testability	10 min
4.6	Usabilidad	10 min
5	Restricciones de Arquitectura	15 min
6	Comentarios Finales	5 min.

Atributos de Calidad (Req. No Funcionales)

Son los aspectos del sistema, que en general no afectan directamente a la funcionalidad necesitada, sino que definen la calidad y las características que el sistema debe soportar.

- Relación con la arquitectura
  - La arquitectura posibilita o inhibe los atributos de calidad
  - No todos los atributos son responsabilidad de la arquitectura
  - Trade-off entre atributos

Escenarios

- **Problemas Comunes**
  - Definición poco sustentable ("el sistema debe ser modificable", todos son modificables)
  - Solapamiento de los atributos (problemas de seguridad afectan a la disponibilidad y a la usabilidad).

- Todo Atributo de Calidad posee al menos los siguientes elementos (primera aproximación a un escenario)
  - **Estimulo**
  - **Ambiente**
  - **Respuesta**

Escenarios (cont.)

Elemento	Descripción
Origen del Estímulo.	Cualquier actor que interactúa con el sistema.
Estimulo.	Es una condición que necesita ser considerada cuando arriba al sistema.
Ambiente.	Son las condiciones en la cual se encuentra el sistema en el momento que se recibe el estimulo.
Componentes.	Son los componentes del sistema que son afectados.
Respuesta.	La respuesta es la actividad que debe realizar el sistema.
Medida de la Respuesta.	Es un tipo de medida con al cual debe cumplir la respuesta para que el requerimiento pueda ser testeado.

Tácticas

Las **tácticas** son decisiones que tienen como objetivo lograr los atributos de calidad.

Las podemos agrupar por los requerimientos que atacan:

- Atributos de Calidad (Cualidades de ejecución)
- Restricciones de Arquitectura (Cualidades de evolución)

Conflictos Entre los Atributos de Calidad

- A veces las decisiones conflictúan entre sí y hay que elegir.
- Una táctica puede beneficiar a un atributo y perjudicar a otro.
- Otras veces entran en conflicto con los requerimientos del negocio
  - Costo
  - Time To Market
- Dos principios
  - Utilidad
  - Simplicidad
- Una buena arquitectura es aquella que combina tácticas que permiten satisfacer todas las necesidades del sistema.

Agenda

#	Tema	Duración
1	Introducción	10 min
4	Atributos de Calidad / Tácticas	
4.1	Performance	15 min
4.2	Disponibilidad	15 min
4.3	Modifiability	15 min
4.4	Seguridad	10 min
4.5	Testability	10 min
4.6	Usabilidad	10 min
5	Restricciones de Arquitectura	15 min
6	Comentarios Finales	5 min.

Atributos de Calidad – No Funcionales

- Performance (Rendimiento)
- Availability (Disponibilidad)
- Security (Seguridad)
- Testability (Comprobabilidad)
- Modifiability (Modificabilidad)
- Usability (Usabilidad)

Performance

"Es el grado en el cual el sistema o componente lleva a cabo una funcionalidad especifica dada una restricción de velocidad, precisión, etc. y el uso eficiente de los recursos"

- Patrones de Arribos (Eventos)
  - Puede ser periódico, probabilístico o esporádico.
  - Dependiendo lo que se mide no importa la cantidad de usuarios sino los pedidos o transacciones.
- Patrones de Respuesta
  - Latencia. Puede ser medido por el tiempo que tarde el sistema en responder
  - La cantidad de transacciones que el sistema puede responder
  - Números de Eventos no procesados y la cantidad de información perdida por que el sistema no puede responder

Performance

- Ej. Los usuarios inician 1.000 transacciones X por minuto (probabilidad) bajo condiciones normales, de 9 a 18:00hs, el sistema debe procesarlas (resultado en pantalla) en una latencia menor a 3 segundos.

Elemento	Descripción
Origen del Estímulo.	Usuarios. Definir el tipo de Usuario si es necesario.
Estimulo.	Inicio de Tx X. Probabilístico. 1.000 tx por minuto
Ambiente.	Procesamiento Normal. De 9 a 18. Carga Normal.
Componentes.	Todo el Sistema
Respuesta.	Transacción procesada
Medida de la Respuesta.	la latencia debe ser menor a 3 segundos

Tácticas Performance – Pasos previos

- Requerimientos
  - Evitar el "lo más rápido posible".
- ¿Qué se puede medir?
  - Consumo de recursos
    - Procesador, memoria, disco, red, etc.
  - Latencia
  - Disponibilidad, Contención, Dependencia en otras computaciones.
  - Escalabilidad
- Medición y profiling
  - Medición del consumo de los recursos durante la ejecución.
  - Se necesitan datos de prueba realistas (no necesariamente "reales")

Tácticas para lograr Performance

- Demanda
  - Incrementar la eficiencia computacional
  - Eliminar el overhead computacional
  - Bound Execution Times
  - Bound Queue Size
- Administración de recursos
  - Concurrencia
  - threads especializados por tarea
  - múltiples threads equivalentes
  - Tamaño de transacciones
  - Múltiples copias de los datos. Caching
  - Incrementar los Recursos disponibles
- Arbitraje de recursos
  - First-in / First-out (FIFO)
  - Fixed Priority
  - Dynamic priority
  - Static Scheduling

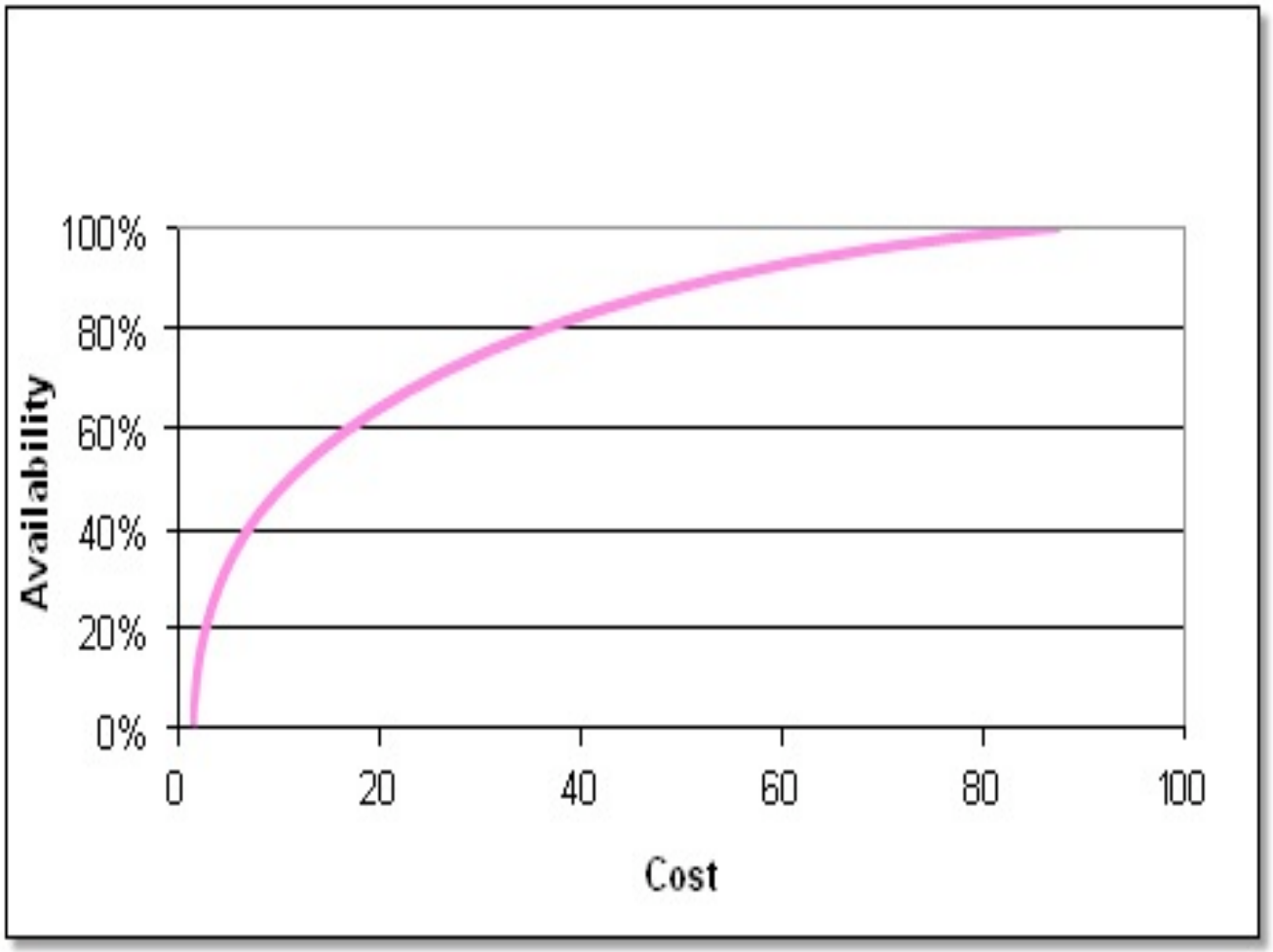
Disponibilidad (Availability)

"Es el grado de operabilidad de un sistema o componente"

- La disponibilidad esta relacionada con las fallas del sistema y las consecuencias asociadas.
- Es importante diferenciar entre desperfecto/error (**fault**) y falla (**failure**), el desperfecto puede provocar una falla, pero si no es controlado. Las fallas son observables, los desperfectos, en general, no.
- La disponibilidad se podria calcular "El sistema o componente debe cumplir con un 99.9% de disponibilidad", la manera de calcular dicho porcentaje es la siguiente:

$$\alpha = \frac{\text{mean time to failure}}{\text{mean time to failure} + \text{mean time to repair}}$$

Disponibilidad y Costo



Disponibilidad (Availability) – (cont.)

Dentro de Disponibilidad, generalmente agrupamos los siguientes atributos:

- Confiabilidad
- Recuperación de desastres
- Cortes (Programados y No Programados)
- Tolerancia a Fallos

Disponibilidad

- Ej. "Si el Middleware tiene alguna falla y no recibe pedidos, para transacciones que necesitan ser procesadas de manera asincrónicas se debe auditar en un archivo de log y se debe reintentar X veces con un intervalo de Y segundos"

Elemento	Descripción
Origen del Estímulo.	El Middleware está caído
Estimulo.	Transacciones asincrónicas a procesar
Ambiente.	En todo momento
Componentes.	Manejador de Transacciones e Interfaz con el Middleware
Respuesta.	Guardar en log y Reintentar
Medida de la Respuesta.	X veces con un intervalo de Y

Tácticas para lograr Disponibilidad

- Detección de fallas
  - Ping + Echo / Heartbeat
  - Exceptions
- Recuperación
  - Preparación y reparación
    - Cluster
    - Voting / Redundancia activa / pasiva / spare
    - Transparent Failover
    - Session replication
  - Reinserción
    - -Shadow / Resynchronization / Checkpoint
- Prevención
  - Removal from Service / Process Monitor
  - Transacciones
  - Tácticas de Integración
  - Colas
  - Replicación de datos
  - Dejarlo en un estado intermedio y después terminar.

Atributos de Calidad – Modifiability

"Está relacionado con costo del cambio. Los aspectos a tener en cuenta son: qué hay que cambiar, cuándo y quién."

- Se podrían agrupar en:
  - **Maintainability**
    - Relacionado con la resolución de problemas
  - **Extensibility**
    - Permitir extender el software con nuevas funcionalidades
  - **Restructuring**
    - Relacionado con la reorganización y alocaciones de módulos.
  - **Interoperability**
    - Relacionado con la integración con otros Sistemas

Atributos de Calidad – Modifiability

- Ej. "El arquitecto empresarial desea que el sistema publique algún servicio para ser consumido por otra aplicación (y no por pantalla) luego de que el sistema esté en su primera release de producción y el tiempo de desarrollo (analisis, design, code, test y deploy) no supere los 25 días desde la aprobación del Change Request."

Elemento	Descripción
Origen del Estímulo.	Arquitecto empresarial
Estimulo.	Desea agregar funcionalidad
Ambiente.	Etapa de Mantenimiento
Componentes.	Core Service e Interfaces
Respuesta.	Cambio de Requerimiento correctamente implementado
Medida de la Respuesta.	Que no supere los 25 días.

Tácticas para lograr Modifiability

- La mayoría de las tácticas pasan por "localizar" las modificaciones, evitando los "ripple effects"
- Para eso se trabaja sobre la calidad del diseño
  - Cohesión y acoplamiento
  - Coherencia semántica
  - Generalización
  - Anticiparse a los cambios
  - Limitar las opciones posibles

Efecto"ripple"(ondas)

Es la aparición de efectos inesperados en lugares posiblemente remotos del sistema después de hacer un cambio. Muestra un acoplamiento indeseado.

Tácticas para lograr Modifiability (cont.)

- Tipos de "ripples"
  - Sintaxis de datos o servicios
  - Semántica de datos o servicios
  - Secuencia
  - Identidad/ Ubicación/ Existencia
  - Calidad de servicio
  - Comportamiento
- Prevenir "Ripple efects"
  - Restringir los canales de comunicación
  - Encapsular información
  - Intermediarios (Data / Service / Identity / Location / Existence)
  - Mantener Interfaces Existentes
  - Agregar Interfaces, Adapters, Stubs
- Aplazar tiempo de enlace ("binding time")
  - Polimorfismo
  - Adherirse a protocolos definidos
  - Archivos de configuración
  - Dependency Injection
  - Registración en tiempo de ejecución

Seguridad

Es la medida sobre la habilidad del sistema para resistir usos no autorizados mientras sigue proveyendo sus servicios a los usuarios legítimos.

- El sistema puede ser atacado, buscando acceso y/o modificación de datos y/o servicios
  - Denial-of-Service
- No esta restringida a la protección contra ataques.

Seguridad

- Se puede caracterizar de la siguiente manera:
  - No Repudio
  - Aseguramiento
  - Confidencialidad
  - Integridad
  - Disponibilidad
  - Auditoria

Seguridad

- Ej. "Si el administrador de accesos (seguridad) realiza, agrega o quita algún rol a cualquier usuario, estos cambio deben ser reflejados de manera inmediata en la sesión del usuario a la cual se le modificaron los accesos"

Elemento	Descripción
Origen del Estímulo.	Administrador de Accesos
Estimulo.	Cambio de roles
Ambiente.	En cualquier momento
Componentes.	Todo el Sistema
Respuesta.	Reflejar los cambios
Medida de la Respuesta.	Inmediatamente

Tácticas para lograr Seguridad

- Resistencia a ataques
  - Autenticación de usuarios
  - Autorización de usuarios
  - Confidencialidad de los datos (encripción, SSL, VPN)
  - Integridad de los datos (redundancia: checksums, hash, firmas)
  - Limitar la exposición o el acceso (DMZ)
- Detección de ataques
- Recuperación
  - Recuperación, restauración
  - Identificación, Auditoría

Testability

"Es el grado de facilidad que tiene un sistema para probarse correcto"

- El 40% del costo de un proyecto es consumido por el testing
- Es importante tener en cuenta qué decisiones arquitectónicas pueden lograr bajar los tiempos de testeo, pero hay que tener en cuenta la complejidad y lo que impacta esa decisión en la construcción.
  - Automatización
  - Separación en capas para facilitar el testeo, etc.

Testability

- Ej. "Un testeador unitario que realiza el test de un componente X debe poder ejecutar los scripts y debe poder llegar a un nivel de completitud del 70% en 4 horas"

Elemento	Descripción
Origen del Estímulo.	Tester unitario
Estimulo.	Testear el componente
Ambiente.	En etapa de desarrollo y mantenimiento.
Componentes.	Componente X
Respuesta.	70% de completitud
Medida de la Respuesta.	4 horas

Tacticas para lograr Testability

- Tipos de Test
  - Unitario
  - Integración
  - Aceptación
  - Performance / Carga / Stress
  - Regresión

- Escenarios
  - Casos de test
  - Juegos de datos
  - Casos básicos
  - Casos extremos
  - Test Driven Development (contract first)

Tacticas para lograr Testability (cont.)

- Test Unitario
  - Modularidad
  - Separar interfaz de implementación
  - Mock Objects
  - Configuraciónes específicas para testing
  - Inversion Of Control / Dependency Injection
  - Modificación de la interfaz de para facilitar el testeo
- Regresión
  - Automatización de tests
  - Automatización de los juegos de datos
  - Manejo de las dependencias de ejecución
  - Automatización de la corrida, periódica o por eventos

Tacticas para lograr Testability (cont.)

- Integración
  - Manejo de las dependencias de ejecución
    - Transacciones individuales
    - Transacción global
- Aceptación
  - Automatización de Input / Output
    - Record / playback
      - Simulación de pedidos HTTP
      - Simulación de teclado/mouse
    - Scripts
- Performance
  - Monitoreo (procesador, memoria, red)
  - Simulación de tráfico usando robots

Tacticas para lograr Testability (cont.)

- Otras Tácticas
  - Debugging
    - Breakpoints
    - Avance paso a paso
    - Exploración del estado de variables
  - Logging
  - Manejo de excepciones y mensajes
  - Capacidad de modificar el programa en runtime
    - Lenguaje
    - Arquitectura
  - Similitud de entornos de desarrollo / test/ producción
  - Acceso a los entornos de test/ producción

Usabilidad

"Es el grado de facilidad del sistema para ser operado"

- Areas:
  - Facilidad de aprendizaje
  - Uso eficiente
  - Minimizar el impacto de los errores
  - Adaptabilidad
  - Incrementar la satisfacción del usuario

Usabilidad

Si bien este atributo de calidad es principalmente definido y especificado o en los casos de uso o en la especificación de la User Interface, existen operaciones y características que deben ser implementadas por la Arquitectura:

- proveer la posibilidad de cancelar operaciones que están siendo procesadas.
- deshacer una acción ejecutada y procesada
- otros

Usabilidad

- Ej. Los usuarios del modulo X, una vez que ejecutan cualquier acción, el sistema debe proveer posibilidad de cancelarlas durante su ejecución y quedar como antes de la ejecución de la misma

Elemento	Descripción
Origen del Estímulo.	Usuarios del módulo X
Estimulo.	Minimizar el impacto de error cancelando operaciones
Ambiente.	Procesamiento Normal. Carga Normal.
Componentes.	Módulo X
Respuesta.	Transacción cancelada
Medida de la Respuesta.	Sistema integro antes de la ejecución

Tácticas para lograr Usabilidad

- Familiaridad
  - Adecuación a estándares
  - Consistencia
  - Metáfora de interacción
  - Información de contexto
  - Cantidad de información, evitar información supérflua
- Adaptación al nivel de usuario
  - Menús para usuarios básicos
  - Comandos y shortcuts para usuarios avanzados
  - Accesibilidad
  - Aprendizaje, configurabilidad
- Velocidad

Agenda

#	Tema	Duración
1	Introducción	10 min
4	Atributos de Calidad / Tácticas	
4.1	Performance	15 min
4.2	Disponibilidad	15 min
4.3	Modifiability	15 min
4.4	Seguridad	10 min
4.5	Testability	10 min
4.6	Usabilidad	10 min
5	Restricciones de Arquitectura	15 min
6	Comentarios Finales	5 min.

Restricciones de Arquitectura

- Buildability
- Robustness

Tácticas para lograr Buildability

- Brindar al equipo un conjunto de herramientas que le resulten útiles y suficientes para construir la aplicación
- Claridad
  - Resolución de servicios genéricos
- Expresividad
  - Declaratividad
  - Lenguajes y abstracciones
- Consistencia
  - Reusabilidad
  - Motores de reglas
- Simplicidad

Tácticas para lograr Robustez

- Integridad conceptual
- Centralización de servicios
  - Manejo de transacciones
  - Manejo de situaciones excepcionales
  - Administración de recursos (e.g.: conexiones de base de datos)
- Proceso de desarrollo
  - Estructura de tests de arquitectura
  - Source Configuration Management
  - Políticas de uso del repositorio de código
  - Políticas de liberación de entregables
  - Automatización

Agenda

#	Tema	Duración
1	Introducción	10 min
4	Atributos de Calidad / Tácticas	
4.1	Performance	15 min
4.2	Disponibilidad	15 min
4.3	Modifiability	15 min
4.4	Seguridad	10 min
4.5	Testability	10 min
4.6	Usabilidad	10 min
5	Restricciones de Arquitectura	15 min
6	Comentarios Finales	5 min.

Comentarios Finales

- La clasificación planteada no quiere decir que sea estricta, todo lo contrario, es una buena manera de agrupar pero no siempre tiene que ser de esta manera.
- La clasificación es siempre buena tenerla desde la arquitectura, con el objetivo de atacar los problemas de manera más ordenada.
- Existen otros atributos de calidad, que también podrían clasificarse, como:
  - Portabilidad
  - Escalabilidad
  - Auditoria o Logging
- No importa tanto la terminología como la capacidad de comunicar conceptos. Si hay problemas de nomenclatura o clasificación, debemos ser capaces de entender el problema para luego resolverlo, tomando las decisiones arquitecturales correctas.

Referencia

- [SAinPractice] Len Bass, Paul Clements, Rick Kazman, Software Architecture in Practice, Second Edition. Addison Wesley, 2003, ISBN 0-321-15495-9.

- [TheArtSA] Stephen T. Albin , The Art of Software Architecture: Design Methods and Techniques, John Wiley & Sons, 2003, ISBN 0-471-22886-9.

- [POSA] Buschmann, F.,Meunier, R.,Rohnert, H.,Sommerlad, P.,Stal, M. 1996. Pattern-Oriented Software Architecture: A System of Patterns. Chichester: John Wiley and Sons, 1996, ISBN 0-471 95889-7

- Software Architecture in Practice, Second Edition. Len Bass, Paul Clements, Rick Kazman. Addison Wesley, 2003, ISBN 0-321- 15495-9.