Universidad Andrés Bello

Testing y Calidad

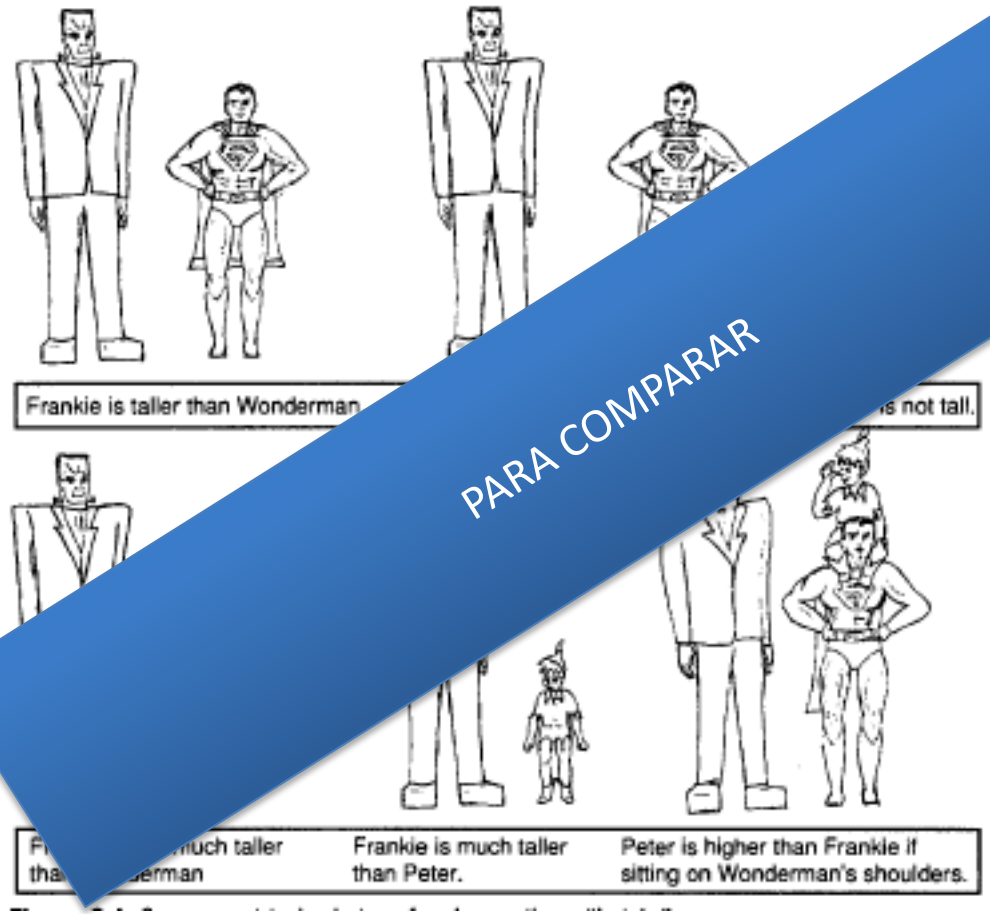
Capítulo 2 – parte 2

Calidad

Métricas de la calidad del software

Por qué medir?

Universidad Andrés Bello



PARA COMPARAR

Ejemplos de métricas

Universidad Andrés Bello

- Tamaño de documentos (medidos en páginas)
- Tamaño de código (medido en LOC)
- Tamaño de delta de código
- Tamaño total sobre el tamaño del código base
-

Ejemplo de Métricas de software: LOC

- Número de líneas liberadas de código fuente. :
  - Contar Todo elemento activo de código afectando las funcionalidades
  - No contar
    - titulos de programas, nombre de autores, historia de cambios, comentarios u otros.
    - Funciones de bibliotecta, dependencias externas
    - El contenido de header, include, import son contados una vez.

Universidad Andrés Bello

Sólo métricas objetivas?

Ejemplo metas subjetivas

Universidad Andrés Bello

- La opinión de los participantes durante la caminata respecto a lo útil de la actividad en una escala likert.

- Nos da una buena idea de la percepción que tienen los participantes acerca de la utilidad de las caminatas.

Dónde medir

Universidad Andrés Bello

- Del progreso
  - De la planificación y monitoreo de las pruebas
    - Tareas comenzadas vs completadas
  - Del desarrollo de la prueba
    - Número de procedimientos de prueba especificados y aprobados
    - Cubrimiento relevante en la especificación para requermientos, riesgos, procesos de negocio, estructuras …
  - De la ejecución y el reporte
    - Número de procedimientos ejecutados vs pasados
    - Número de procedimientos de pruebas ejecutados como pruebas de regresión
  - Del cierre
    - Tareas completadas

Qué medir

- Tiempo gastado en tareas específicas
- Costo por tiempo

Obtener mediciones de

Universidad Andrés Bello

- Número de elementos cubiertos por las pruebas ejecutadas
- Número de incidentes reportados
- Número de incidentes por tipo de clase: defectos, mejoras, malos entendidos
- Número de defectos reportados que ya han sido corregidos
- Número de incidentes cerrados
- Sentencias subjetivas acerca de la confianza de los diferentes stakeholders

Universidad Andrés Bello

No basta con sólo recolectar mediciones sino presentar el valor real que tiene para el proyecto y cómo apoyan nuestras decisiones

Pasos

Universidad Andrés Bello

- Primero acordar con tus stakeholders qué vas a medir
- Para eso debemos definir y acordar
  - Qué es lo que queremos saber, qué haremos con esos valores, qué monitorearemos y si lo controlaremos?
  - Cada cuanto mediremos?
  - Cuál es exactamente la escala que usaremos?
  - Si es la misma métrica pero sobre diferentes unidades o diferentes periodos de tiempo cómo se compararán?
  - PLAN: QUE, POR QUIEN, CUANDO, COMO y POR QUE

Inspecciones de programas

Universidad Andrés Bello

- Es la aproximación formalizada a las revisiones del documento
- Está pensado explícitamente para la detección de defectos (no su corrección)
- Los defectos pueden ser errores lógicos, anomalías en el código que pueden indicar una condición errónea (p.ej: una variable no iniciada).

Precondiciones de la inspección

- Debe haber una especificación precisa disponible.
- Los miembros del equipo deben estar familiarizados con los estándares de organización.
- Debe estar disponible un código sintácticamente correcto del sistema.
- Debería prepararse una lista de errores.
- La gestión debe aceptar que la inspección aumentará los costes pronto en el proceso de software.
- La gestión no debería utilizar inspecciones para la evaluación del personal, es decir, para encontrar quién comete errores.

Proceso de inspección

- Visión de conjunto del sistema presentado al equipo de inspección.
- Los códigos y documentos asociados se distribuyen al equipo de inspección por adelantado.
- La inspección tiene lugar y se anotan los errores encontrados.
- Se hacen modificaciones para reparar los errores descubiertos.
- Puede requerirse o no una reinspección.

Roles en el proceso de inspección

Autor o propietario	El programador o diseñador responsable de generar el programa o documento. Responsable de reparar los defectos descubiertos durante el proceso de inspección.
Inspector	Encuentra errores, omisiones e inconsistencias en los programas y documentos. También puede identificar cuestiones más generales que están fuera del ámbito del equipo de inspección.
Lector	Presenta el código o documento en una reunión de inspección
Secretario	Registra los resultados de la reunión de inspección
Presidente o moderador	Gestiona el proceso y facilita la inspección. Realiza un informe de los resultados del proceso para el moderador jefe.
Moderador jefe	Responsable de las mejoras del proceso de inspección, actualización de las listas de comprobación, estándares de desarrollo, etc.

Listas de inspección

Universidad Andrés Bello

- Debería utilizarse una lista de errores comunes para guiar la inspección.
- Las listas de errores dependen del lenguaje de programación y reflejan los errores característicos que es probable que aparezcan en el lenguaje.
- En general cuanto más «débil» sea la comprobación del tipo, más grande será la lista.
- Ejemplos: inicialización, nombramiento de constantes, terminación de bucles, límites de vectores, etc.

Comprobaciones de inspección 1

Defectos de datos	¿Se inicializan todas las variables antes de que se utilicen sus valores? ¿tienen nombre las constantes? ¿El límite superior de los vectores es igual al tamaño del vector o al tamaño 21? Si se utilizan cadenas de caracteres, ¿tienen un delimitador explícitamente asignado? ¿Existe alguna posibilidad de que el búfer se desborde?
Defectos de control	Para cada sentencia condicional, ¿es correcta la condición? ¿Se garantiza que termina cada bucle? ¿están puestas correctamente entre llaves las sentencias compuestas? En las sentencias case, ¿se tienen en cuenta todos los posibles casos? Si se requiere una sentencia break después de cada caso en las sentencias case ¿Se ha incluido?
Defectos de entrada /salida	¿Se utilizan todas las variables de entrada? ¿Se les asigna un valor a todas las variables de salida? ¿Pueden provocar corrupciones de datos las entradas no esperadas?

Comprobaciones de inspección 2

Defectos de interfaz	¿Las llamadas a funciones y a métodos tienen el número correcto de parámetros? ¿Concuerdan los tipos de parámetros reales y formales? ¿Están en el orden correcto los parámetros? Si los componentes acceden a memoria compartida, ¿tienen el mismo modelo de estructura de la memoria compartida?
Defectos de gestión de almacenamiento	¿Si una estructura enlazada se modifica, ¿se reasignan correctamente todos los enlaces? Si se utiliza almacenamiento dinámico ¿se asigna correctamente el espacio de memoria? Se desasigna explícitamente el espacio de memoria cuando ya no se necesita?
Defectos de manejo de expcepciones	¿Se tienen en cuenta todas las condiciones posibles de error?

Cifras de inspección

- 500 sentencias/hora durante la visión de conjunto.
- 125 sentencias de código fuente/hora durante la preparación individual.
- Pueden inspeccionarse de 90 a 125 sentencias/hora.
- Por lo anto la inspección es un proceso costoso.
- Inspeccionar 500 líneas cuesta el esfuerzo de unas 40 horas/hombre (unos 4146 € en cifras españolas).

Comprobaciones del análisis estático

Clase de defecto	Comprobación del análisis sintáctico
Defectos de datos	Variables utilizadas antes de su inicialización. Variables declaradas pero nunca utilizadas Variables asignadas dos veces pero nunca utilizadas entre asignaciones. Posibles violaciones de los límites de los vectores. Variables no declaradas
Defectos de control	Código no alcanzable. Saltos incondicionales en bucles.
Defectos de entrada/salida	Las variables salen dos veces sin intervenir ninguna asignación.
Defectos de interfaz	Inconsistencias en el tipo de parámetros. Inconsistencias en el número de parámetros. Los resultados de las funciones no se utilizan. Existen funciones y procedimientos a los que no se les llama
Defectos de gestión de almacenamiento	Punteros sin asignar. Aritmética de punteros.

Etapas del análisis estático

Universidad Andrés Bello

- Análisis del flujo de control. Comprueba los bucles con múltiples puntos de entrada o salida, encuentra códigos inalcanzables.
- Análisis de uso de los datos. Detecta variables no inicializadas, variables escritas dos veces sin que intervenga una asignación, variables que se declaran pero nunca se usan, etc.
- Análisis de interfaz. Comprueba la consistencia de una rutina, las declaraciones del procedimiento y su uso.

Etapas del análisis estático

Universidad Andrés Bello

- Análisis de flujo de información. Identifica las dependencias de las variables de salida. No detecta anomalías en sí pero resalta información para la inspección o revisión del código.
- Análisis de caminos. Identifica los caminos del programa y arregla las sentencias ejecutadas en el camino. Nuevamente, es potencialmente útil en el proceso de revisión.

Principales Objetivos

Universidad Andrés Bello

OBJ1: Facilitar el control de la gestión como la planificación y ejecución de las intervenciones necesarias

¿QUÉ MÉTRICAS NECESITAMOS?

Principales Objetivos

Universidad Andrés Bello

OBJ1: Facilitar el control de la gestión como la planificación y ejecución de las intervenciones necesarias

¿QUÉ MÉTRICAS NECESITAMOS?

- Desviación del rendimiento actual del planificado
- Desviación de la calendarización y presupuesto del planificado

Principales Objetivos

Universidad Andrés Bello

OBJ2: Identificar situaciones en que se requiere mejoras en el proceso de desarrollo o mantención en la forma de acciones preventivas o correctivas introducidas mediante la organización

¿QUÉ MÉTRICAS NECESITAMOS?

Principales Objetivos

Universidad Andrés Bello

OBJ2: Identificar situaciones en que se requiere mejoras en el proceso de desarrollo o mantención en la forma de acciones preventivas o correctivas introducidas mediante la organización

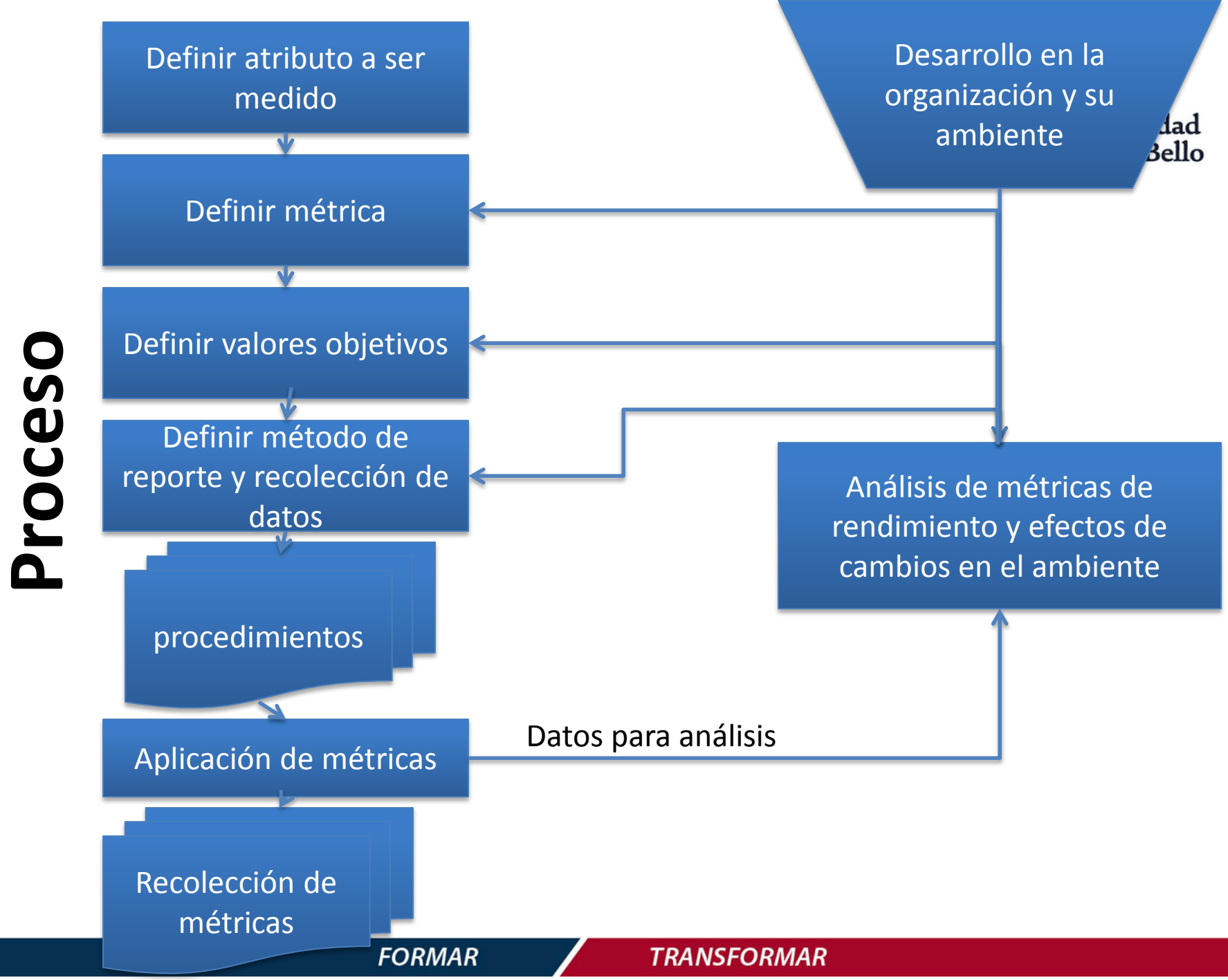
¿QUÉ MÉTRICAS NECESITAMOS?

- Acumulación de información de métricas respecto al rendimiento de equipos de trabajo, unidades, entre otros

Base de Comparación

Universidad Andrés Bello

- ✓ Estándares
- ✓ Objetivos
- ✓ Logros años pasados
- ✓ Logros de proyecto pasado
- ✓ Niveles promedio de calidad de otros equipos con herramientas de desarrollo similares
- ✓ Niveles promedio de calidad de la organización
- ✓ Prácticas industriales


Proceso
Definir atributo a ser medido
Definir métrica
Definir valores objetivos
Definir método de reporte y recolección de datos
procedimientos
Aplicación de métricas
Recolección de métricas
Desarrollo en la organización y su ambiente
Análisis de métricas de rendimiento y efectos de cambios en el ambiente
Datos para análisis

Métricas esenciales en un proyecto

Universidad Andrés Bello

Métricas de Tamaño

- ✓ KLOC

Covariancia óptima de Kullback-Leibler

Kilo de líneas de código

Mil líneas de código

Métricas esenciales en un proyecto

Universidad Andrés Bello

Métricas de Productividad de Software

- ✓ Productividad
- Productividad= Nº horas sprint/KLoc

Métricas esenciales en un proyecto

Universidad Andrés Bello

Métricas de Productividad de Software

Code	Name	Calculation formula
DevP	Development Productivity	$DevP = \frac{DevH}{KLOC}$
FDevP	Function point Development Productivity	$FDevP = \frac{DevH}{NFP}$
CRe	Code Reuse	$CRe = \frac{ReKLOC}{KLOC}$
DocRe	Documentation Reuse	$DocRe = \frac{ReDoc}{NDoc}$

Key:

- DevH = total working hours invested in the development of the software system.
- ReKLOC = number of thousands of reused lines of code.
- ReDoc = number of reused pages of documentation.
- NDoc = number of pages of documentation.

Métricas esenciales en un proyecto

Universidad Andrés Bello

Métrica de densidad de errores

- ✓ Densidad
- Densidad= Nº total de errores(defectos)/Kloc
- Nº total de errores(defectos)= número de errores de la fase + número de defectos de la fase

Métricas esenciales en un proyecto

Universidad Andrés Bello

Métrica de densidad de errores

Error severity class	Relative weight
Low severity	1
Medium severity	3
High severity	9

Error severity class *a*	Calculation of NCE (number of errors) *b*	Calculation of WCE Relative weight *c*	Calculation of WCE Weighted errors $D = b \times c$
Low severity	42	1	42
Medium severity	17	3	51
High severity	11	9	99
Total	70	—	192
NCE	70	—	—
WCE	—	—	192

CED Code Error Density $CED = \frac{NCE}{KLOC}$

WCED Weighted Code Error Density $WCED = \frac{WCE}{KLOC}$

Métricas esenciales en un proyecto

Universidad Andrés Bello

Métricas de Efectividad de Remoción de Defectos

- ✓ Efectividad
- Efectividad=número de errores/(número de errores de la fase + número de defectos de la fase)

Métricas esenciales en un proyecto

Universidad Andrés Bello

Métricas de Efectividad de Remoción de Defectos

Code	Name	Calculation formula
DERE	Development Errors Removal Effectiveness	$DERE = \frac{NDE}{NDE + NYF}$
DWERE	Development Weighted Errors Removal Effectiveness	$DWERE = \frac{WDE}{WDE + WYF}$

Key:

- NYF = number of software failures detected during a year of maintenance service.
- WYF = weighted number of software failures detected during a year of maintenance service.

Métricas esenciales en un proyecto: Ejemplo

Un proyecto que fue realizado en Scrum, conto con dos sprint de dos semanas cada uno, trabajando 20 horas a la semana con un equipo de 3 miembros.

En el primer sprint se realizaron 30 KLOC con una eficiencia desconocida. Sabiendo que el numero de errores que pasa a la siguiente etapa es de 120 y el número de defectos son 80.

En el segundo sprint se obtuvo una densidad de 0,05 manteniendo la eficiencia de 0,92 determinando que el numero de errores que pasa a la siguiente etapa es de 200

Determinar:

- ✓ Productividad de desarrollo de cada uno de los sprint
- ✓ Numero total de defectos del proyecto
- ✓ Promedio de la densidad del proyecto
- ✓ Promedio de la eficiencia del proyecto

1er sprint
30 kloc
errores siguiente etapa=120

Densidad=defectos/Kloc
Eficiencia=número de errores/(número de errores de la fase + número de defectos de la fase)
Productividad= Nºhoras sprint/Kloc
Numero total de errores (defectos) = 200
Densidad=200/30.000
Densidad= 0,006
Eficiencia= 120/200 = 0,6
Productividad = 80/30.000 = 0,0026 hrs/líneas código

2do sprint
densidad= 0,05
eficiencia=0,92
errores siguiente etapa=200

Numero total de errores (defectos) = 280
Densidad=defectos/kloc KLOC= 280/0,05 → **5 KLOC**
Densidad= 0,05
Productividad = 80/5600 = 0,014 hrs/líneas código

- ✓ **Productividad de desarrollo de cada uno de los sprint**
  1. **Sprint 0,0026 hrs/líneas código**
  2. **Sprint 0,014 hrs/líneas código**
- ✓ **Numero total de defectos del proyecto**
  1. **Sprint Numero total de errores = 200**
  2. **Sprint Numero total de errores = 280**

  Total de errores del proyecto 480
- ✓ **Promedio de la densidad del proyecto**

  1 Sprint Densidad= 0,006
  2 Sprint Densidad= 0,05
  Promedio densidad proyecto 0,028
- ✓ **Promedio de la eficiencia del proyecto**

  1 Sprint eficiencia= 0,6
  2 Sprint eficiencia= 0,92
  Promedio densidad proyecto 0,76

Métricas esenciales en un proyecto: Caso 1

Universidad Andrés Bello

Un proyecto que fue realizado en Scrum, conto con tres sprint de tres semanas cada uno, trabajando 40 horas a la semana con un equipo de 3 miembros.
En el primer sprint se realizaron 30 KLOC con una eficiencia desconocida. Sabiendo que el numero de errores que pasa a la siguiente etapa es de 210 y el número de defectos son 140.
En el segundo sprint se obtuvo una densidad de 0,03 manteniendo la eficiencia de 0,72 determinando que el numero de errores que pasa a la siguiente etapa es de 99.

Determinar:

- ✓ Productividad de desarrollo de cada uno de los sprint
- ✓ Numero total de defectos del proyecto
- ✓ Promedio de la densidad del proyecto
- ✓ Promedio de la eficiencia del proyecto

Métricas esenciales en un proyecto: Caso 2

Universidad Andrés Bello

Un proyecto que fue realizado en Scrum, conto con tres sprint de tres semanas cada uno, trabajando 40 horas a la semana con un equipo de 4 miembros.

En el primer sprint se desconoce el KLOC con una eficiencia de 0,4. Sabiendo que el numero de errores que pasa a la siguiente etapa es de 250 y el número de defectos son 199.

En el segundo sprint se obtuvo una densidad de 0,09 manteniendo la eficiencia de 0,4; determinando que el numero de errores que pasa a la siguiente etapa es de 180.

En el tercer sprint la densidad aumento a 0,1 en las 40 KLOC realizadas, la cantidad de errores al final del sprint fue de 500 y los defectos fueron de 121.

Determinar:

- ✓ Productividad de desarrollo de cada uno de los sprint
- ✓ Numero total de defectos del proyecto
- ✓ Promedio de la densidad del proyecto
- ✓ Promedio de la eficiencia del proyecto

Qué se Requiere para Implementar las Métricas de Calidad en una Organización

Universidad Andrés Bello

- ✓ Definición de métricas de calidad relevantes y adecuadas por equipo, por departamentos
- ✓ Aplicaciones regulares por la unidad
- ✓ Análisis estadísticos de datos recolectados respecto a las métricas
- ✓ Acciones subsecuentes:
  - Cambios en la organización y métodos del desarrollo y mantención
  - Cambios en las métricas o en sus formas de recolección de datos de métricas
  - Aplicación de datos y análisis de datos a las acciones correctivas de planificación de todas las unidades relevantes

Análisis Estadísticos de los Datos

Universidad Andrés Bello

- ✓ Análisis de los datos de métricas proveen oportunidades para comparar una serie de métricas de proyecto
- ✓ Comparación de métricas contra indicadores pre-definidos
- ✓ Comparación de efectividad en que la métrica alcanza su meta
- ✓ Responder preguntas

Análisis Estadísticos de los Datos

Universidad Andrés Bello

Responder Preguntas

- Hay diferencias significantes entre la calidad de servicio de los equipos desarrolladores?
- Las métricas apoyan el supuesto de que la aplicación de la nueva versión de una herramienta de desarrollo contribuye significativamente a la calidad del software?

Análisis Estadísticos de los Datos

Universidad Andrés Bello

- ✓ Estadística Descriptiva (ej. Promedio)
  - – Ilustra la información, visualizar tendencias
- ✓ Estadística Analítica
  - – Análisis de varianza, regresión …

Limitaciones de las Métricas

Universidad Andrés Bello

- ✓ Restricciones de presupuesto
- ✓ Factores humanos – empleados se oponen a ser evaluados
- ✓ Incertidumbre respecto a la validez de los datos
- ✓ Métricas de baja exhaustividad o baja validez
  - Parámetros usados en métricas de desarrollo tal como KLOC
- ✓ Porcentaje de código reutilizado
- ✓ Profesionalismo de las revisiones y pruebas
- ✓ Estilos de reportes de revisiones y pruebas

Limitaciones de las Métricas

Universidad Andrés Bello

Principales factores afectando los parámetros de proceso:

- ✓Estilos de programación → KLOC
- ✓Volúmen de comentarios en el código → KLOC
- ✓Complejidad del Software – módulos más complejos requieren más tiempo de desarrollo por línea de código en comparación con módulos simples
  - Módulos complejos también sufren de más defectos que los simples

Limitaciones de las Métricas

Universidad Andrés Bello

Principales factores afectando los parámetros de producto:

- ✓Calidad del sw instalado y su documentación
- ✓Número de instalaciones
- ✓Porcentaje de reuso del código
- ✓Estilo de programación y volumen de documentación

Métricas de Calidad del Proceso

Universidad Andrés Bello

- ✓ Métrica de densidad de errores
- ✓ Métricas de severidad de errores
- ✓ Métricas de complejidad ciclomática

Métricas de Calidad del Proceso
Métricas de Densidad de Errores

Universidad Andrés Bello

Qué involucra??

- Volumen de SW
  - Cómo mido el volumen del Software?
- Conteo de errores
  - Cómo cuento errores??

Preguntas