UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

Departamento de Informática
Universidad Técnica Federico Santa María

# Calidad de Software

## Ingeniería de Software

**Hernán Astudillo & Cristian Orellana**
*Departamento de Informática*
*Universidad Técnica Federico Santa María*

# Aseguramiento de Calidad de Software

- Objetivo de Ingeniería de Software
  - Producir consistentemente software de calidad
- Calidad engloba todo el proceso, y está determinada por factores directos e indirectos
- Calidad
  - Concepto complejo y multifacético
  - Diversas perspectivas

# Visiones de Calidad

- Visión trascendental
  - puede ser reconocida pero no definida
- Visión del usuario
  - grado de adecuación al propósito
- Visión del productor
  - conformidad con la especificación
- Visión del producto
  - ligada a características inherentes del mismo
- Visión basada en valor
  - ¿cuánto el cliente está dispuesto a pagar?

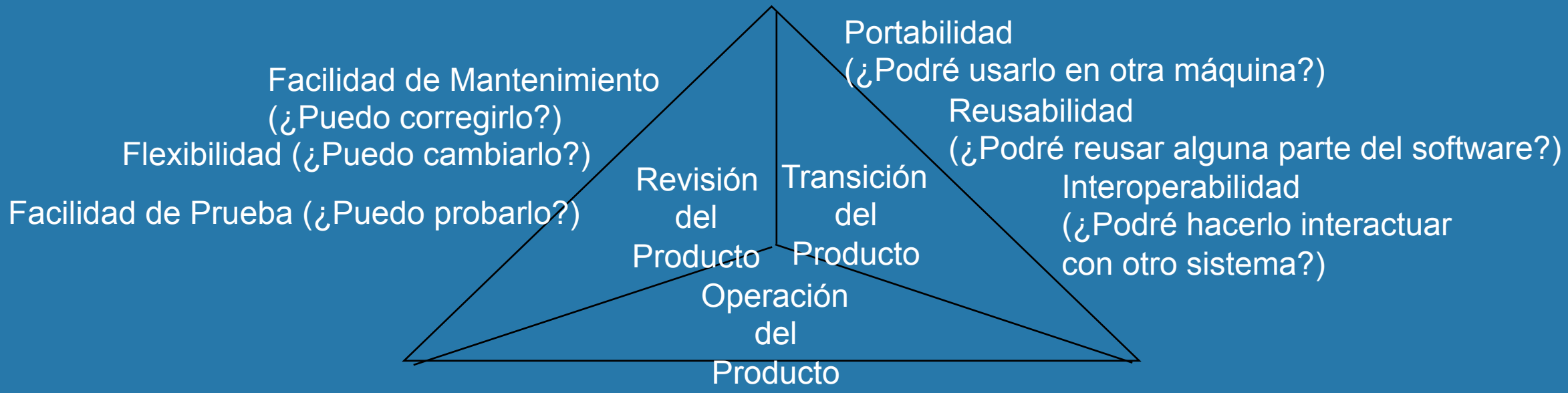
# Modelos de calidad de software

- Modelo de calidad
  - taxonomía de atributos de calidad y sus relaciones
- Atributo de calidad
  - caracterización específica o propiedad de un proceso o producto
    - puede ser medido u observado
    - incluyen las tradicionalmente llamadas “ilities”
  - Cada atributo de calidad puede tener varias métricas
- Cada métrica define
    - una medida o escala (cuantitativa o cualitativa)
    - un método o técnica para observar o medir el atributo

# Modelo de calidad – nota

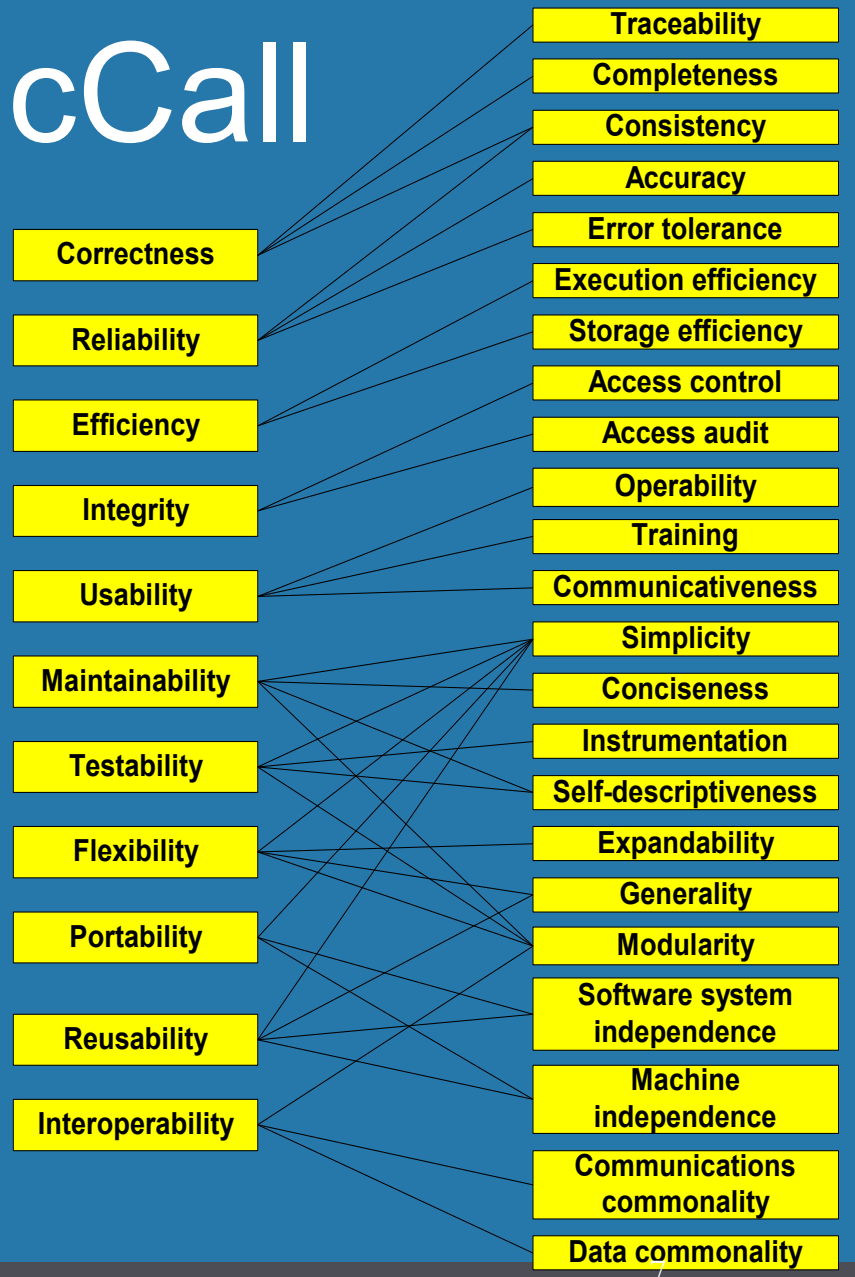
- Métricas internas
  - Medición de atributos del software por sí mismo
  - Aplicadas durante la construcción del sistema
  - Ej: desempeño de un subsistema, complejidad del código, tamaño
- Métricas externas
  - Medición de atributos visibles externamente
  - Requieren evaluar al software en su contexto
    - Aplicadas a un producto de software en ejecución
  - Ej: funcionalidad, confiabilidad, desempeño
- Métricas comunes
  - Tamaño
    - de módulos, del diseño mismo, del código
  - Complejidad
  - Modularidad
    - Cohesión
    - Acoplamiento

# Modelo de Calidad de McCall

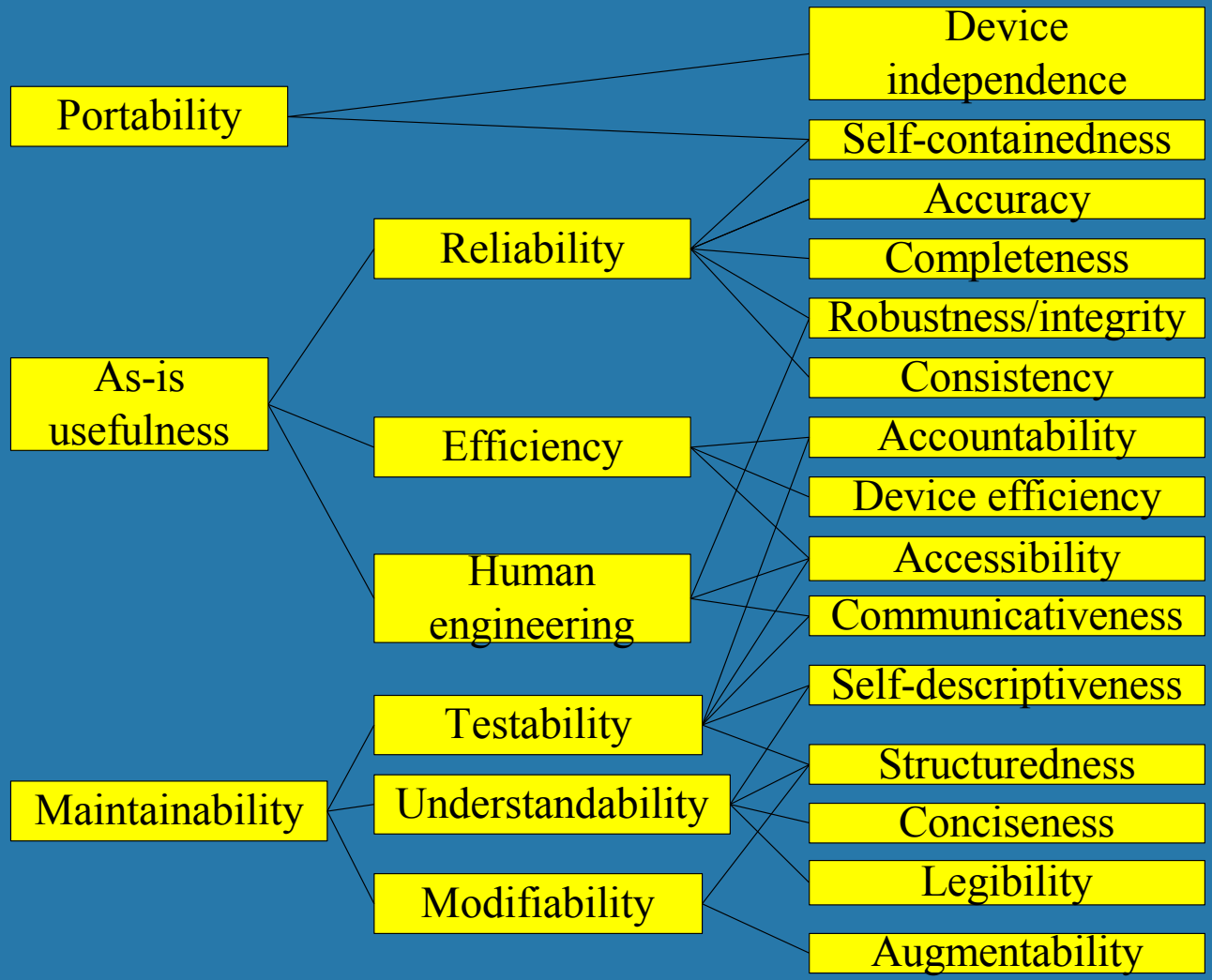



Corrección (¿Hace lo que quiero?)
Confiabilidad (¿Lo hace de forma confiable todo el tiempo?)
Eficiencia (¿Se ejecutará en mi hardware lo mejor que pueda?)
Integridad (¿Es seguro?)
Facilidad de uso (¿Está diseñado para ser usado?)

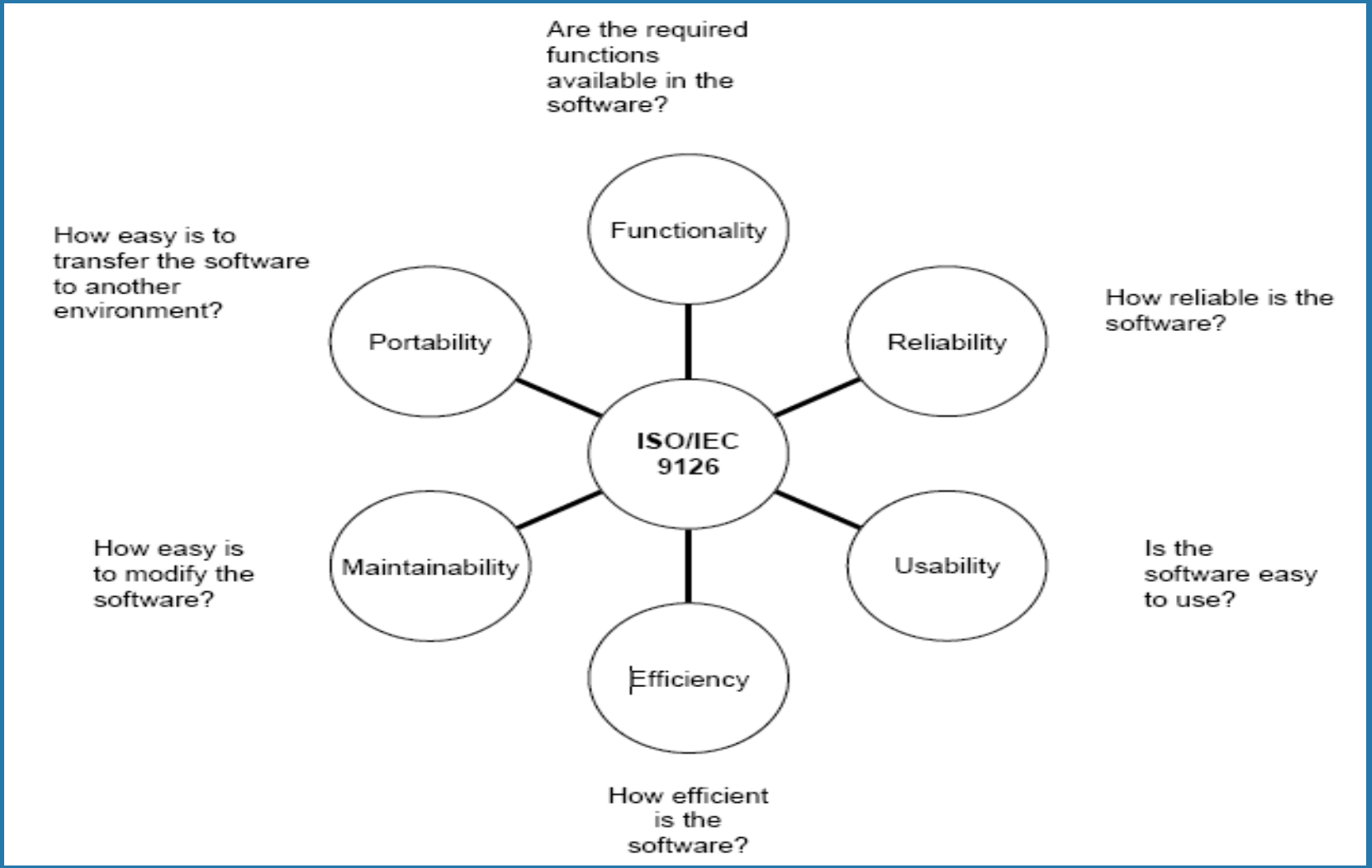
# Modelo de Calidad de McCall



# Modelo de Calidad de Boehm

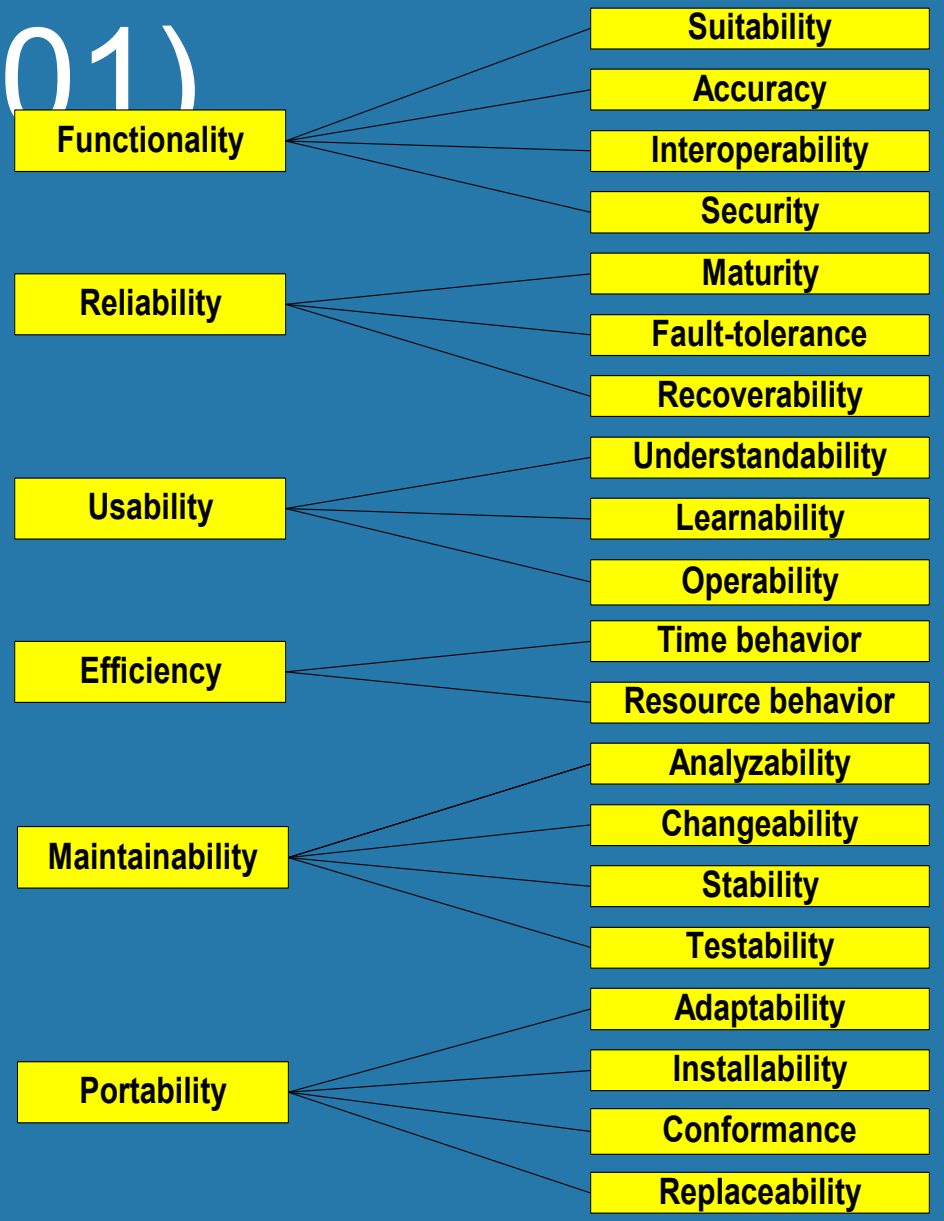


# ISO 9126 (1991 & 2001)



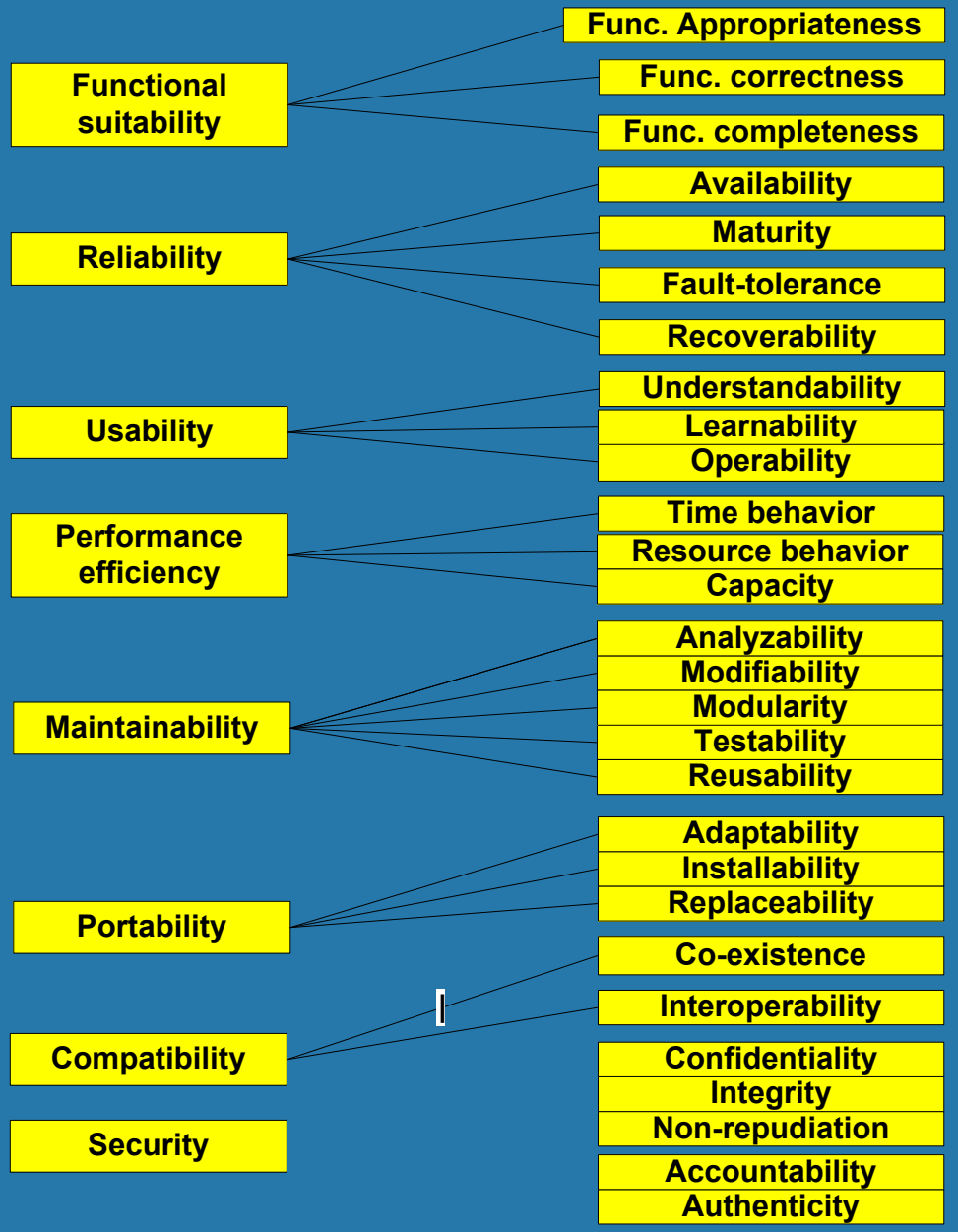
# ISO 9126 (1991 & 2001)

- 6 Características
  - p.ej. “portabilidad”
- 22 Sub-características
  - p.ej. “adaptabilidad”
- Atributos
  - entidad que se puede verificar o medir en producto de software
  - no definidos por el estándar: dependen del producto
  - p.ej. “plataformas”



# ISO 25010 (2011)

- 8 Características
- 31 Sub-características



# ISO 25010

- Sub-característica Conformidad (“conformance”)
  - se aplica a todas las características
  - P.ej. conformidad a la legislación referente a usabilidad y fiabilidad

# Atributos de Calidad (Microsoft)

- Availability
- Conceptual Integrity
- Interoperability
- Maintainability
- Manageability
- Performance
- Reliability
- Reusability
- Scalability
- Security
- Supportability
- Testability
- User Experience / Usability

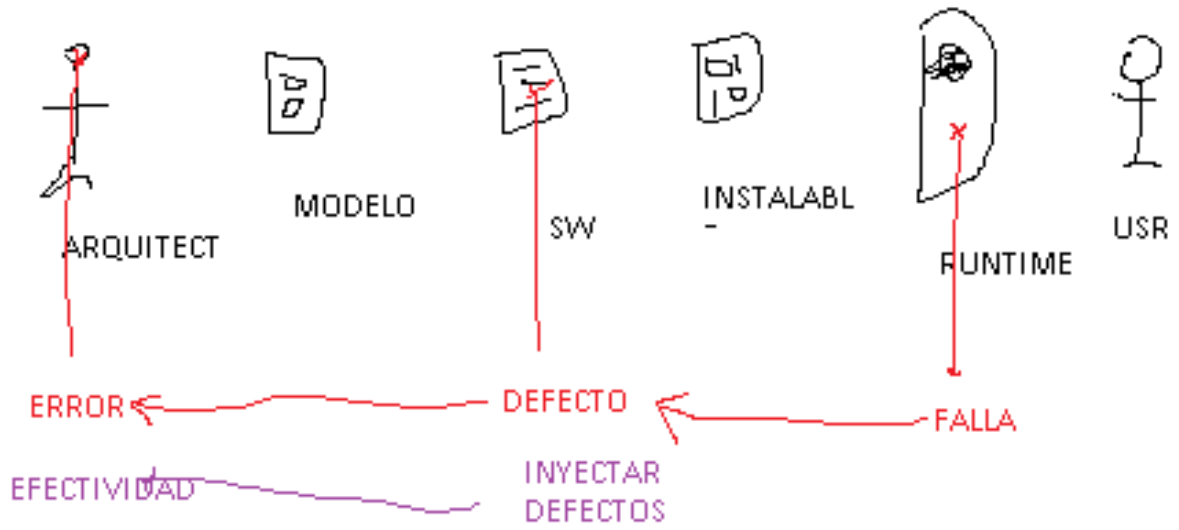
# Problema: ¡A nadie le importa!

- A nadie le importa confiabilidad, eficiencia, portabilidad y mantenibilidad
  - A nadie le importan TODOS ellos
- Las personas (stakeholders, lectores) sólo se preocupan de algunas categorías
  - Tendría más sentido definir calidad basada en el lector

# Ejercicio

- Modelo de calidad para SIGA

# Modelo de Calidad "5 niveles"

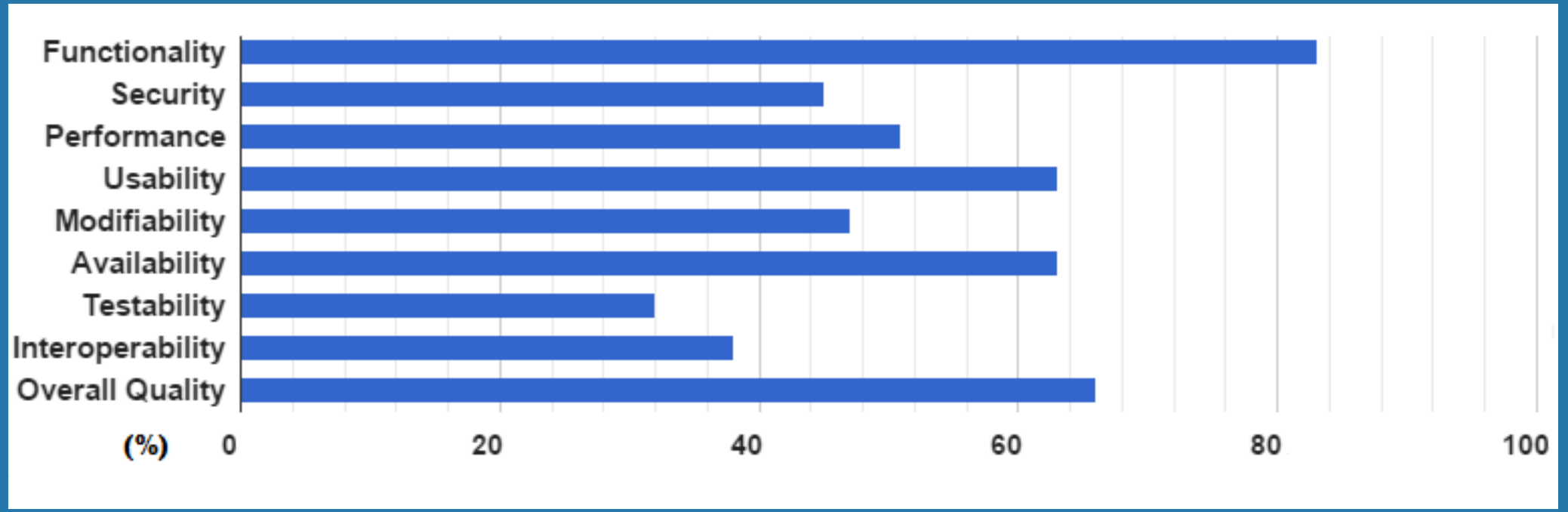

ARQUITECT
MODELO
SW
INSTALABL
-
RUNTIME
USR
ERROR
DEFECTO
FALLA
EFECTIVIDAD
INYECTAR
DEFECTOS

# Atributos de calidad en la práctica

- [Tumyrkin et al., 2016]
- Diseño experimental-> Encuesta a profesionales
- Encuesta compuesta por 19 preguntas
- 103 personas de 37 países respondieron la encuesta

# Atributos de calidad en la práctica

- Nivel de satisfacción de los clientes en sus proyectos de software en función de atributos de calidad



# Atributos de calidad en la práctica

- Nivel de satisfacción de los clientes con respecto a atributos de calidad para diferentes tipos de industrias

# Aseguramiento de Calidad de Software

# Aseguramiento de Calidad de Software

- Software Quality Assurance (SQA)
  - Acciones sistemáticas y planificadas requeridas para asegurar la calidad de software
- Grupo de SQA
  - Objetivos: planificar, desarrollar y controlar el proceso de verificación y validación
  - Actividades: aplicación de métodos, revisiones e inspecciones, testing, estándares, control de cambios, mediciones, registro

# Verificación y Validación

- Verificación -- ¿estamos construyendo el producto correctamente?
- Validación -- ¿estamos construyendo el producto correcto?

# Revisiones de Software [1/3]

- Idea clave: someter a chequeo uno o más productos de trabajo
  - Actúan como filtro
- Descubrimiento temprano de defectos
  - Gran impacto en los costos de testing y mantención
  - Defectos de software tienen efecto de amplificación
  - Permiten evitar “microcascadas”
- Tres tipos (D. Galin):
  - Revisiones formales de diseño
    - Formal: sólo producto de trabajo aprobado puede continuar a siguientes fases o tareas del desarrollo
  - Revisiones por pares
    - Inspecciones
    - Caminatas de código/texto
  - Revisiones por expertos

# Revisiones de Software [2/3]

- Objetivos
  - Detectar errores en la lógica, función o implementación
  - Verificar satisfacción de requerimientos
  - Asegurar cumplimiento de estándares
  - Fomentar uniformidad
  - Hacer proyectos más manejables
- Herramientas relacionadas
  - Issue trackers (ej: Jira, Mantis Bug Tracker, Taiga,…)
  - Code reviewers (ej: Upsource,…)
  - Discusión colaborativa (ej: Confluence, Taiga…)

# Revisiones de Software [3/3]

- Guía de Acción
  - Definir tamaño, conformación y duración
  - Revisar producto, no productor
  - Establecer agenda
  - Limitar debates y rebates
  - Enunciar problemas, no resolverlos
  - Llevar registro
  - Limitar tamaño del grupo
  - Exigir preparación previa
  - Definir checklists
  - Asignar recursos
  - Entrenar a los revisores
  - Revisar las revisiones

# Inspecciones Fagan

- Desarrolladas en 1972, por Michael Fagan
- Beneficios: prevención y reducción de defectos, y por ende de costos
- Proceso de inspección en 6 etapas

# Inspecciones Fagan

- Etapas
  - Planeación: preparación logística (materiales, disponibilidades, fechas)
  - Overview: conocimiento general, asignación de roles (autor, lector, tester, moderador)
  - Preparación: estudio del material a ser inspeccionado
  - Inspección: búsqueda de defectos
  - Retrabajo: hacer las correcciones
  - Iteración: verificación de lo realizado, y de la no introducción de efectos secundarios

# Otras formas de revisión

- Talleres de patrones de diseño
- Lectura dirigida de casos de uso

UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

Departamento de Informática
Universidad Técnica Federico Santa María

# Calidad de Software

## Ingeniería de Software

**Hernán Astudillo & Cristian Orellana**
*Departamento de Informática*
*Universidad Técnica Federico Santa María*