

Papers Fundacionales

7 Mitos acerca de los Métodos Formales

Foguelman - Modrow - Tilli

DC - UBA

30 de junio de 2013

Introducción

Mito 1: Pueden garantizar que el software es perfecto

- Límite en las demostraciones (mundo real vs modelo)

Mito 1: Pueden garantizar que el software es perfecto

- Límite en las demostraciones (mundo real vs modelo)
- Error en la especificación

Mito 1: Pueden garantizar que el software es perfecto

- Límite en las demostraciones (mundo real vs modelo)
- Error en la especificación
- + Demostración de correctitud: propiedades globales, y relación entre programa y especificación

Mito 1: Pueden garantizar que el software es perfecto

- Límite en las demostraciones (mundo real vs modelo)
- Error en la especificación
- + Demostración de correctitud: propiedades globales, y relación entre programa y especificación
- + Encontrar errores

Mito 1: Pueden garantizar que el software es perfecto

- Límite en las demostraciones (mundo real vs modelo)
- Error en la especificación
- + Demostración de correctitud: propiedades globales, y relación entre programa y especificación
- + Encontrar errores

Hecho: Los métodos formales son muy útiles para encontrar errores de manera temprana y pueden eliminar casi completamente algunas clases de errores

Mito 2: Son útiles sólo para probar correctitud

- + Construcción de la especificación y detección temprana de errores

Mito 2: Son útiles sólo para probar correctitud

- + Construcción de la especificación y detección temprana de errores
- + Demostración de propiedades de la especificación

Mito 2: Son útiles sólo para probar correctitud

- + Construcción de la especificación y detección temprana de errores
- + Demostración de propiedades de la especificación
- + Implementación semiautomática e iterativa

Mito 2: Son útiles sólo para probar correctitud

- + Construcción de la especificación y detección temprana de errores
- + Demostración de propiedades de la especificación
- + Implementación semiautomática e iterativa
- + Prueba correctitud

Mito 2: Son útiles sólo para probar correctitud

- + Construcción de la especificación y detección temprana de errores
- + Demostración de propiedades de la especificación
- + Implementación semiautomática e iterativa
- + Prueba correctitud

Hecho: Trabajan generalmente haciendo que pienses mucho sobre el sistema que pretendes construir

Mito 3: Sólo se benefician los sistemas altamente críticos

- + Se benefician muchos sistemas: críticos, replicados, embebidos en hardware, de alta calidad

Mito 3: Sólo se benefician los sistemas altamente críticos

- + Se benefician muchos sistemas: críticos, replicados, embebidos en hardware, de alta calidad
- + Objetividad, mantenimiento, facilidad de construcción, visibilidad

Mito 3: Sólo se benefician los sistemas altamente críticos

- + Se benefician muchos sistemas: críticos, replicados, embebidos en hardware, de alta calidad
- + Objetividad, mantenimiento, facilidad de construcción, visibilidad
- + Monitoreo del desarrollo

Mito 3: Sólo se benefician los sistemas altamente críticos

- + Se benefician muchos sistemas: críticos, replicados, embebidos en hardware, de alta calidad
- + Objetividad, mantenimiento, facilidad de construcción, visibilidad
- + Monitoreo del desarrollo
- + Distinta granularidad en la formalidad

Mito 3: Sólo se benefician los sistemas altamente críticos

- + Se benefician muchos sistemas: críticos, replicados, embebidos en hardware, de alta calidad
- + Objetividad, mantenimiento, facilidad de construcción, visibilidad
- + Monitoreo del desarrollo
- + Distinta granularidad en la formalidad

Hecho: Son útiles para casi cualquier tipo de aplicación

Mito 4: Involucran matemática compleja

+ Principalmente lógica y teoría de conjuntos

Mito 4: Involucran matemática compleja

- + Principalmente lógica y teoría de conjuntos
- + Conocimientos requeridos similares a los de lenguajes de programación

Mito 4: Involucran matemática compleja

- + Principalmente lógica y teoría de conjuntos
- + Conocimientos requeridos similares a los de lenguajes de programación
- + En general especificación más corta que implementación

Mito 4: Involucran matemática compleja

- + Principalmente lógica y teoría de conjuntos
- + Conocimientos requeridos similares a los de lenguajes de programación
- + En general especificación más corta que implementación
 - Dificultad en modelar el mundo real

Mito 4: Involucran matemática compleja

- + Principalmente lógica y teoría de conjuntos
- + Conocimientos requeridos similares a los de lenguajes de programación
- + En general especificación más corta que implementación
 - Dificultad en modelar el mundo real
 - Muy abstracto o muy específico

Mito 4: Involucran matemática compleja

- + Principalmente lógica y teoría de conjuntos
- + Conocimientos requeridos similares a los de lenguajes de programación
- + En general especificación más corta que implementación
 - Dificultad en modelar el mundo real
 - Muy abstracto o muy específico

Hecho: Se basan en especificaciones matemáticas que son más fáciles de entender que un programa

Mito 5: Incrementan los costos de desarrollo

Hecho: Pueden decrementar los costos de desarrollo

Mito 6: Son incomprensibles para los clientes

Hecho: Pueden ayudar a los clientes a entender qué están comprando

Mito 7: Nadie los usa para proyectos reales

Hecho: Son utilizados con éxito proyectos útiles de la industria