

Tema 2

Métodos Formales

Introducción



- Propósito de los métodos formales
 - ◆ Introducir rigor en todas las fases de desarrollo del software
 - Evitar que se pasen por alto cuestiones críticas
 - ◆ Proporcionar un método estándar de trabajo
 - ◆ Constituir un base de coherencia entre muchas actividades relacionadas
- Lenguajes de programación
 - ◆ Sintaxis y semántica precisa para la fase implementación
 - ◆ Resto de fases → otras fuentes
 - Métodos formales → Nivel comparable de precisión

Deficiencias enfoques menos formales



■ Métodos convencionales

- ◆ Lenguaje natural y representaciones gráficas
 - Bien aplicados → software de elevada calidad
- ◆ Pueden sufrir de algunos problemas
 - Contradicciones
 - Ambigüedades
 - Vaguedad
 - Incompletitud
 - Niveles mezclados de abstracción
- ◆ Las revisiones técnicas formales pueden eliminar muchos de estos problemas.



Deficiencias enfoques menos formales



■ Contradicciones

- ◆ Conjunto de sentencias que difieren entre sí.
 - *El sistema debe supervisar todas las temperaturas en un reactor químico. [...] El sistema sólo debe revisar la temperaturas en un rango.*

■ Ambigüedades

- ◆ Planteamientos que se pueden interpretar de muchas maneras
 - *El operador de identidad consta del nombre del operador y la contraseña. La contraseña consta de 6 dígitos. Debe ser visualizada en la pantalla de seguridad*

Deficiencias enfoques menos formales



■ Vaguedad

- ◆ Lograr un elevado grado de precisión consistentemente es una tarea casi imposible
 - *“La interfaz con el sistema, empleada por los operadores del radar, debe ser amistosa para con el usuario.”*

■ Incompletitud

- ◆ No se especifica toda la información necesaria
 - *El sistema debe mantener el nivel del depósito. Debe almacenarse para los últimos 6 meses*
 - *La función PROMEDIO visualizará en un PC el nivel medio de agua para un sensor concreto entre dos fechas.*

Deficiencias enfoques menos formales



- Niveles mezclados de abstracción
 - ◆ Se entremezclan aleatoriamente sentencias muy abstractas con otras con un gran nivel de detalle
 - *El objetivo del sistema es hacer un seguimiento del stock de un almacén.*
 - *Cuando el encargado de venta escribe la orden venta será preciso comunicar el número de pedido, el número de identificación del material vendido, y el número de unidades solicitadas. El sistema responderá con una confirmación.*
 - ◆ Ambos tipos son importantes
 - La mezcla hace muy difícil analizar la arquitectura funcional del sistema.

¿Qué es un método formal?



- Definición (*Enciclopedia de la Ingeniería del Software*)
 - ◆ Un método es formal si posee una base matemática estable, que normalmente vendrá dada por un lenguaje formal de especificación.
 - Definir de manera precisa → consistencia, completitud, especificación, implementación, corrección
- Atractivo de las matemáticas



¿Por qué las matemáticas?



- Propiedades para el desarrollo de grandes sistemas
 - ◆ Permiten describir de manera exacta una situación física, un objeto o el resultado de una acción
 - Área de una curva \rightarrow integral
 - ◆ Transición suave entre las actividades de la Ing. Software
 - Especificaciones funcionales, diseño del sistema, código
 - ◆ Abstracción
 - Medio exacto
 - Desaparecen muchos de los problemas de los métodos convencionales
 - ◆ Elevado nivel de verificación
 - Prueba matemática para demostrar que un diseño encaja en una especificación o un código refleja un diseño

Modelos formales



- Lógica de primer orden
 - ◆ Especificar un sistema mediante estados y operaciones
 - Los datos y sus relaciones se describen detalladamente
 - Sus propiedades se expresan en lógica de primer orden
 - Teoría conjunto es la base de la semántica
- Algebraicos y de especificación ecuacional
 - ◆ Describen las estructuras de datos de forma abstracta
 - Nombre de los conjuntos de datos, funciones básicas y propiedades
 - Formas ecuacionales
 - ◆ Soporte de deducciones
 - Cálculo ecuacional y sistemas de reescritura

Modelos formales



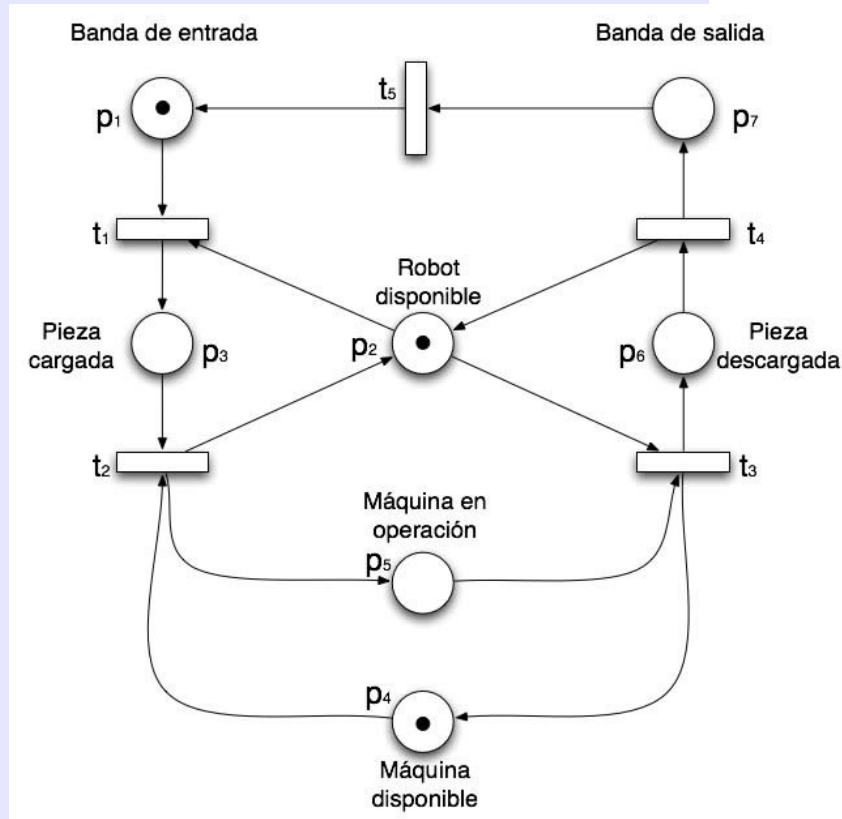
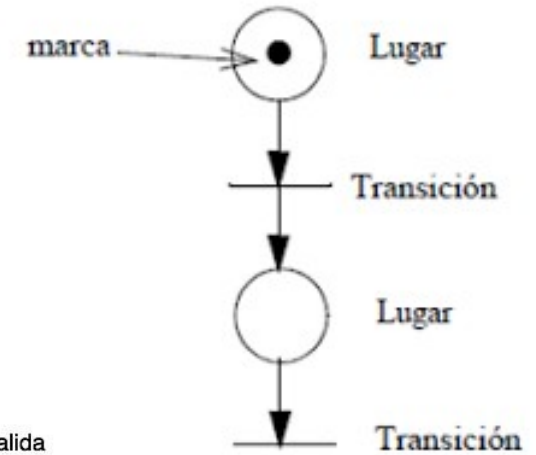
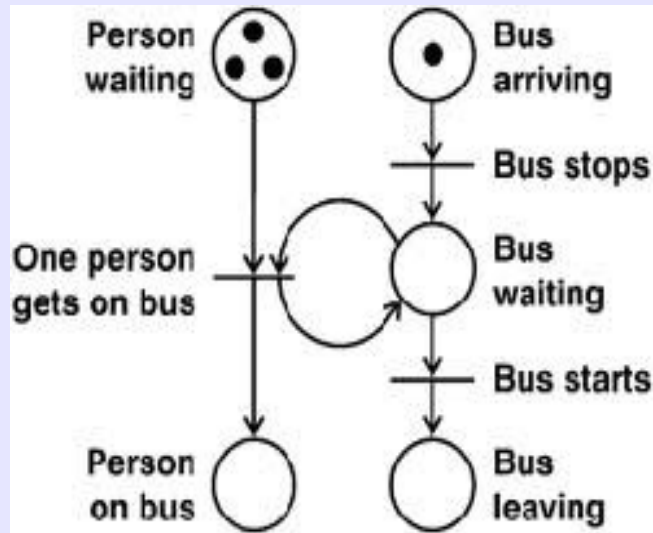
■ Redes de Petri

- ◆ Formalismo basado en autómatas para especificar/modelar comportamiento
 - Modelo formal basado en flujos de información
 - Permite expresar eventos concurrentes
- ◆ Establecen el concepto de estado mediante lugares que pueden contener marcas
- ◆ Describen como evolucionan los sistemas mediante un conjunto de transiciones (con pre y post condiciones)
 - Evolución → producción y consumo de marcas en varios puntos de la red

Modelos formales



■ Redes de Petri



Modelos formales



■ Lógica temporal

- ◆ Se utiliza para describir sistemas concurrentes y reactivos
 - Sistema reactivo → sistema que mantiene una continua interacción con su entorno respondiendo a estímulos externos y produciendo salidas en respuesta a los mismos
 - El orden de los eventos no es predecible
 - Su ejecución no tiene por qué terminar.
 - Ejemplo → interfaz hombre-máquina

Until event **stop** occurs, every occurrence of event **request** is eventually followed by an occurrence of event **response**”

(request → “eventually” response) “Until” stop
(request → \diamond response) \cup stop

Modelos formales



- Álgebra de procesos
 - ♦ Modelan la interacción entre procesos concurrentes
- Verificación de modelos
 - ♦ Trabajan mediante una búsqueda exhaustiva en los estados posibles de un modelo para encontrar errores en la especificación
- Prueba de teoremas
 - ♦ A partir de un conjunto de axiomas se trata de probar si la especificación, extendida con algunos teoremas, es válida.

Métodos Formales en el Proceso Desarrollo



- Especificación formal
 - ◆ Utilización de una notación matemática para proporcionar una descripción precisa de lo que debe hacer un programa
- Desarrollo formal
 - ◆ Se desarrollan programas de una forma tal que se asegura matemáticamente que satisfacen sus especificaciones formales
- Verificación formal
 - ◆ Se utilizan reglas precisas para demostrar matemáticamente que un programa satisface una especificación formal

¿Especificación formal?

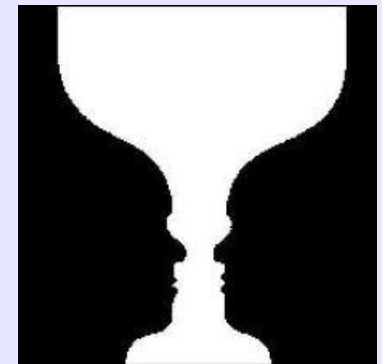
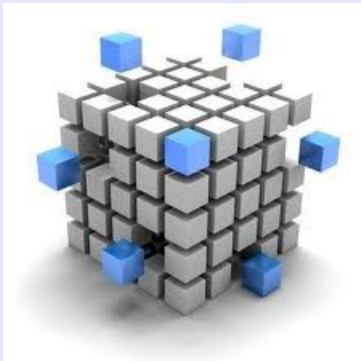


■ Características deseables especificación

- ◆ No ambigüedad
- ◆ Consistencia
- ◆ Completitud



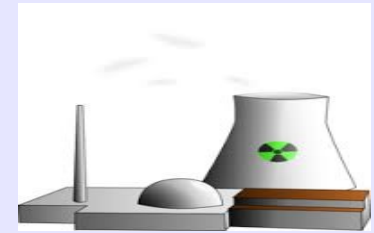
■ Utilizando métodos formales → + probabilidad de éxito



¿Verificación formal?



- Demostrar la corrección de un software es muy importante en algunas aplicaciones
 - ◆ Controladores de reactores nucleares
 - ◆ Sistemas de frenado de coches
 - ◆ Equipos médicos controlados por software
- La demostración de corrección formal
 - ◆ Manera de establecer la ausencia de fallos en el software
 - ◆ No es posible una validación exhaustiva mediante pruebas



Ejemplos aplicación M.F.



- Métodos formales en proyectos software: CICS IBM
 - ◆ Proyecto CICS (Customer Information Control System)
 - Diseñado por IBM
 - Ofrecer acceso a datos, comunicaciones, integridad y servicios de seguridad
 - Utilizado por las principales compañías del mundo
 - 30.000 licencias
 - Aplicaciones
 - Sistemas de compensación bancaria
 - Control de almacenes
 - Reservas aéreas
 - Sistemas ATM

Ejemplos aplicación M.F.



- Métodos formales en proyectos software: CICS IBM
 - ◆ Especificación Formal con Z de la API basada en módulos
 - Detecto errores cometidos en versiones anteriores
 - Nuevas orientaciones para el lenguaje Z
- Seguridad de un aeropuerto
 - ◆ Uso de UML y métodos formales (B y Focal) para modelar estándares que regulan la seguridad de un aeropuerto
 - UML para soportar la actividad de validación
 - Modelo Formal para su verificación

Ejemplos aplicación M.F.



- Chip de firmware sistema móvil FeliCa
 - ◆ Tecnología de tarjetas *contactless* usadas principalmente en monederos electrónicos
 - ◆ Usan el lenguaje VDM+
 - El método formal contribuyó a la calidad de los procesos y a la implementación de la seguridad de la comunicación
- Dispositivos de firma digital seguros
 - ◆ Se recurrió a la especificación formal para definir las políticas de seguridad de la aplicación de firmas de datos

Métodos formales vs Métodos convencionales



- Métodos formales difieren de los métodos tradicionales en la manera y tiempo de cada una de la fases del ciclo de vida del software
 - ♦ Mayor tiempo en la especificación y diseño
 - ♦ Menor tiempo en la verificación

Métodos formales vs Métodos convencionales



■ Ejemplo

- ◆ Sistema de seguridad que graba vídeo digital, controla automáticamente puertas, detecta intrusos y supervisa guardias
 - Se desarrolla un módulo mediante métodos formales (VDM) y métodos tradicionales

	Métodos Formales	Métodos convencionales
N.º de líneas	5500	12000
Cumplimiento requisitos	82%	62%
Tiempo total	28 semanas y media	28 semanas

Ventajas & Desventajas



■ Ventajas de los Métodos Formales

- ♦ Se comprende mejor el sistema.
- ♦ La comunicación con el cliente mejora ya que se dispone de una descripción clara y no ambigua de los requisitos del usuario.
- ♦ El sistema se describe de manera más precisa.
- ♦ El sistema se asegura matemáticamente que es correcto según las especificaciones.
- ♦ Mayor calidad en el software respecto al cumplimiento de las especificaciones.
- ♦ Mayor productividad.

Ventajas & Desventajas



- Los métodos formales no son la panacea
 - ◆ Un diseño formalmente verificado podría no funcionar
 - ◆ Dan una sensación falsa de seguridad



Métodos formales



Pruebas



Ventajas & Desventajas



- Desventajas de los Métodos Formales
 - ◆ El desarrollo de herramientas que apoyen la aplicación de métodos formales es complicado y los programas resultantes son incómodos para los usuarios.
 - ◆ Los investigadores por lo general no conocen la realidad industrial.
 - ◆ Es escasa la colaboración entre la industria y el mundo académico, que en ocasiones se muestra demasiado dogmático.
 - ◆ Se considera que la aplicación de métodos formales encarece los productos y ralentiza su desarrollo.

Siete mitos de los MF



- Garantizan que el software es perfecto
 - ◆ Lleva a expectativas irreales
- Se centran en demostrar corrección
 - ◆ Porque es su mayor uso
- Sólo son útiles para los sistemas críticos
 - ◆ Dificultad de aplicar los métodos formales
- Requieren matemáticos entrenados
 - ◆ Percepción de que las matemáticas son difíciles
- Aumentan el coste del desarrollo
 - ◆ Reducen los costes de mantenimiento

Siete mitos de los MF

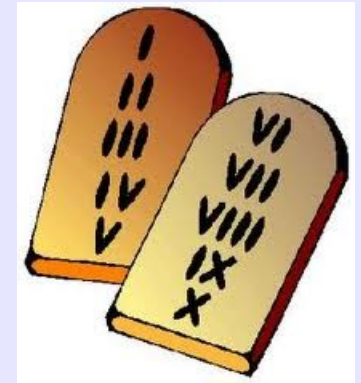


- Son incomprensibles para el usuario
 - ◆ Especificación formal llena de símbolos matemáticos → incomprensibles para quien no conozca la notación
- No se usan en grandes proyectos reales
 - ◆ Los MF se asocian con departamentos académicos y organizaciones de investigación
 - Son los únicos con capacidades para aplicarlos

Decálogo



- Elegirás la notación apropiada
 - ◆ Tener en cuenta: vocabularios, tipo de aplicación a especificar y la amplitud del lenguaje
- Formalizarás pero no en exceso
 - ◆ Componentes cruciales
 - ◆ Componentes cuyo fallo no se puede tolerar
- Estimarás costos
 - ◆ Poner en marcha los métodos formales
- Tendrás un experto en métodos formales a tu disposición
 - ◆ Entrenamiento y consultoría de seguimiento



Decálogo

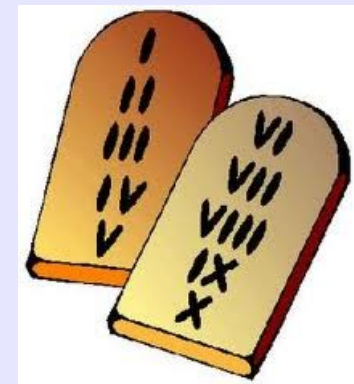


- No abandonarás los métodos tradicionales de desarrollo
 - ◆ Integración de los métodos tradicionales y formales
 - Resultados excelentes
- Documentarás suficiente
 - ◆ Comentario en lenguaje natural para reforzar la comprensión del sistema
- No comprometerás los estándares de calidad
 - ◆ No hay nada mágico- → aplicar otras actividades SQA
- No serás dogmático
 - ◆ Los métodos formales no garantizan la corrección

Decálogo



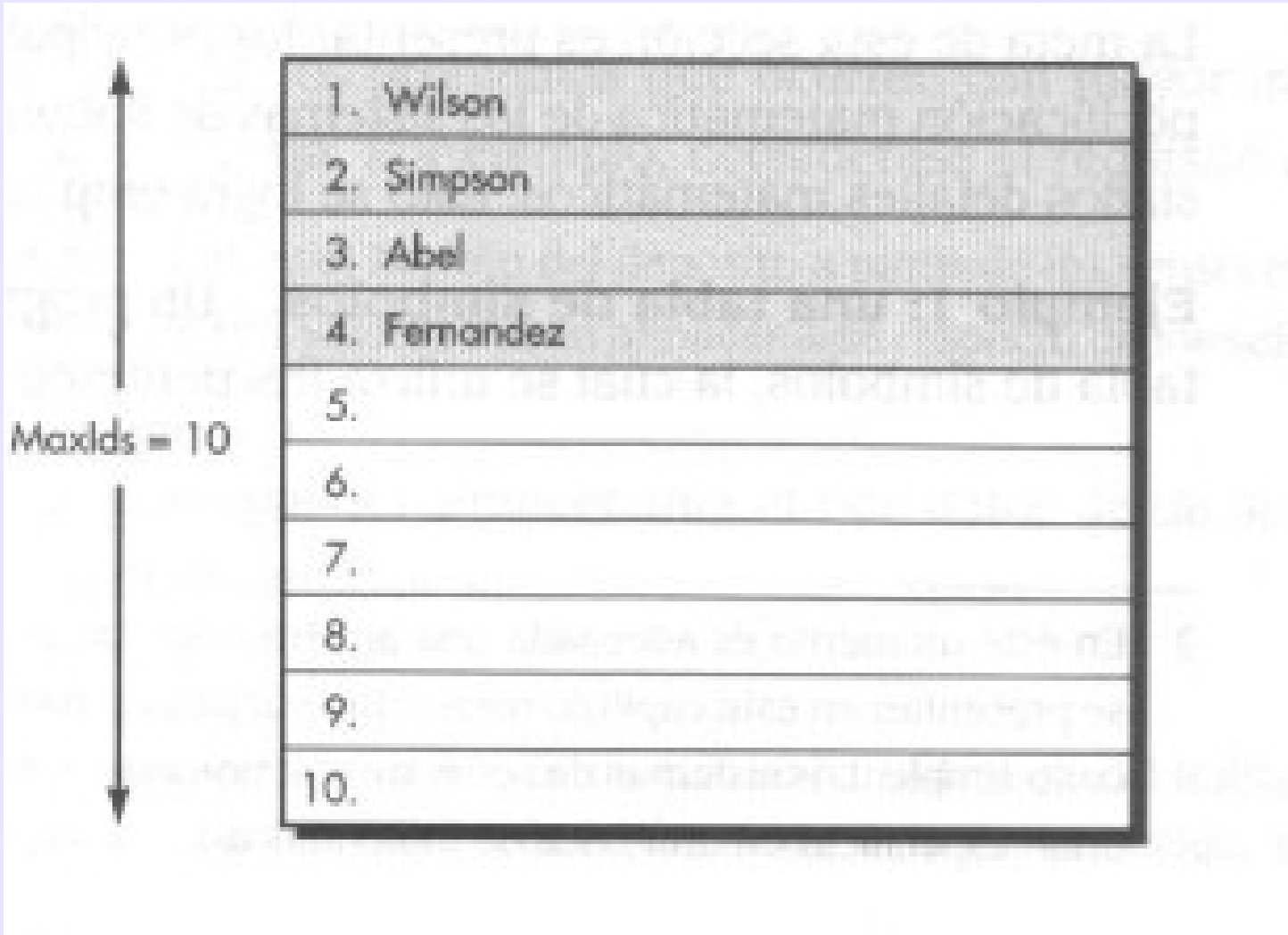
- Probarás, probarás y probarás de nuevo
 - ◆ Los métodos formales no absuelven de la necesidad de llevar a cabo pruebas exhaustivas y bien planeadas
- Reutilizarás cuanto puedas
 - ◆ Reducir los costos y aumentar la calidad



Matemáticas - especificación



■ Ejemplo1: tabla de símbolos



1. Wilson
2. Simpson
3. Abel
4. Fernandez
5.
6.
7.
8.
9.
10.

M&E → Conceptos



■ Invariante de datos

- ◆ Conjunto de condiciones verdaderas a lo largo de la ejecución de un sistema que contiene una colección de datos

■ Estado

- ◆ Datos almacenados en el sistema y que son alterados por éste

■ Operación

- ◆ Acción que ocurre dentro de un sistema y lee o escribe datos modificando el estado.
 - Dependerán del estado en qué esté el sistema

M&E → Conceptos

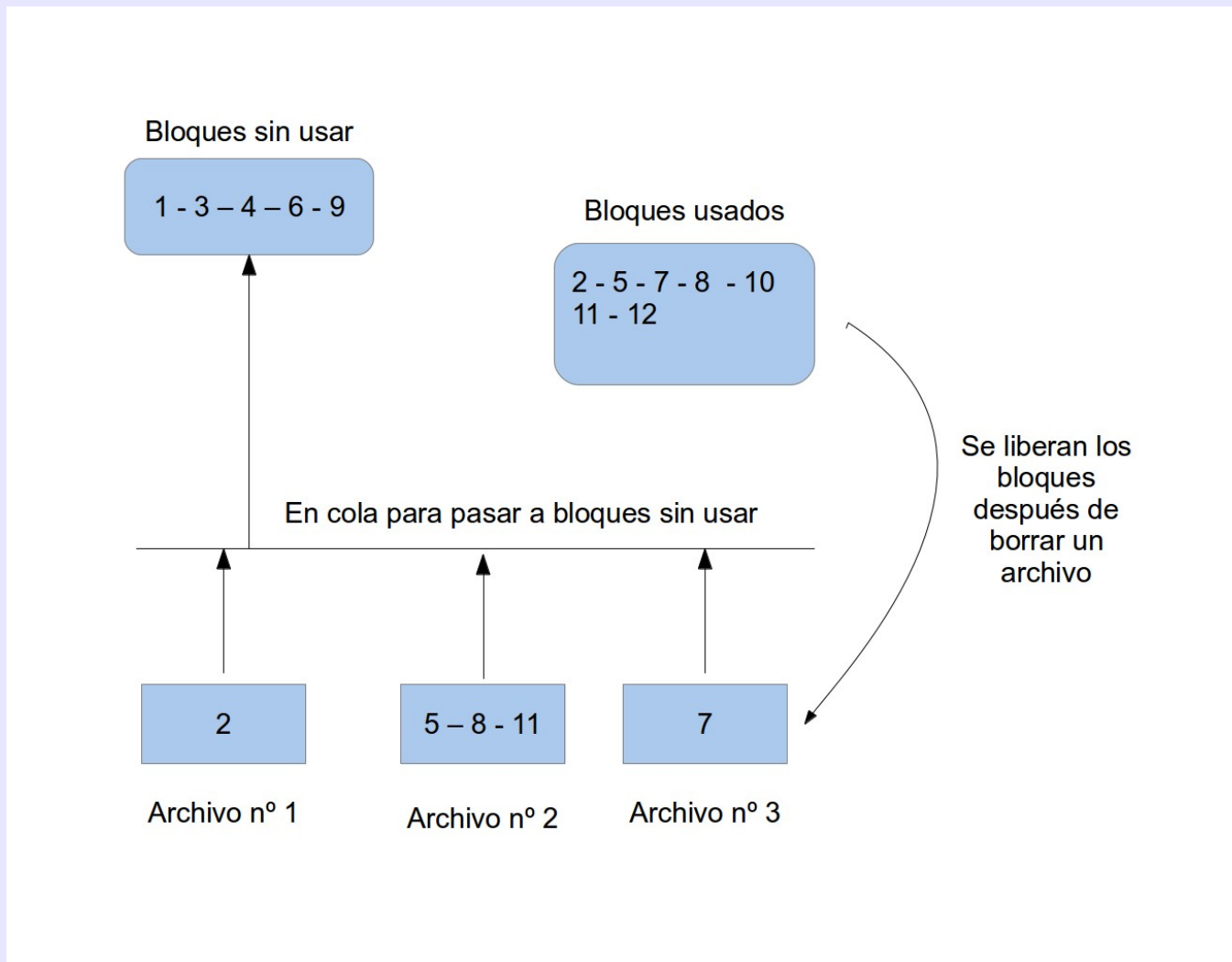


- Tres tipos de condiciones asociadas a las operaciones
 - ◆ Invariante
 - Lo que está garantizado que no cambiará
 - ◆ Precondición
 - Circunstancias en las cuales es válida una operación
 - Condición que ha de cumplirse ante de poder realizar la operación
 - ◆ Postcondición
 - Define lo que sucede cuando la operación ha finalizado su acción.
 - Normalmente, definiendo su efecto sobre el estado

M&E → Conceptos



■ Ejemplo 2: Gestor de bloques



M&E → Conceptos



- Ejemplo 2: Gestor de bloques
 - ♦ Estado
 - Colección de bloques libres
 - Colección de bloques usados
 - Cola de bloques borrados

M&E → Conceptos



■ Ejemplo 2: Gestor de bloques

- ◆ Invariante de datos (expresado en lenguaje natural)
 - No habrá ningún bloque que esté a la vez usado y sin usar
 - Todos los conjuntos de bloques almacenados en la cola serán subconjuntos de la colección de bloques (CB) utilizados en ese momento
 - Ningún número de bloque pertenecerá a dos o más elementos de la cola
 - La CB usados y sin usar será la colección total de bloques que configuran los archivos
 - La CB sin usar no tendrá números de bloques repetidos
 - La CB usados no tendrá números de bloques repetidos

M&E → Conceptos



■ Gestor de bloques

◆ Operaciones

- *Añadir ()* → una colección de bloques al final de la cola
 - Precondición → Los bloques deben de estar en la CB usados
 - Postcondición → CB está al final de la cola
- *Eliminar ()* → una colección de bloques usados del inicio de la cola y colocarlos en la CB sin usar
 - Precondición → La cola debe tener al menos un elemento
 - Postcondición → Los bloques deberán estar en la CB no usados
- *Verificar ()* → si la cola de bloques está vacía
 - Precondición → No posee.
 - Postcondición → Valor verdadero si la cola está vacío
Valor falso en caso contrario.

Preliminares matemáticos



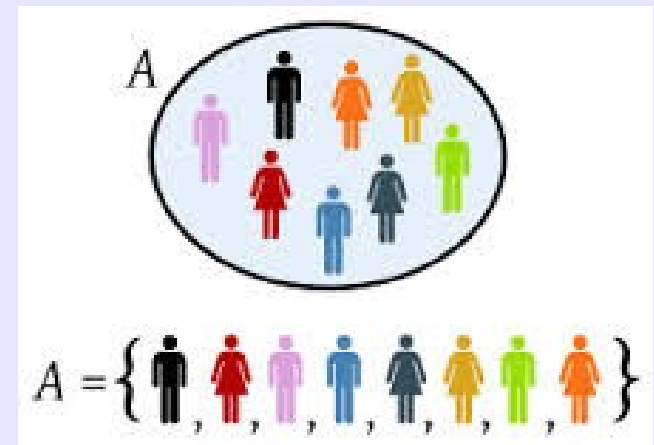
■ Conjuntos

- ◆ Piedra angular de los métodos formales
 - Colección de objetos o elementos no repetidos
 - Orden irrelevante
 - Operador $\# \rightarrow$ cardinalidad del conjunto

➤ $\#\{A,B,C,D\}=4$

◆ Definición

- Enumeración de sus elementos
 - $\{0,1,2\}$
- Especificación constructiva
 - $\{n:N \mid n < 3 \cdot n\}$





■ Especificación constructiva

- ◆ Forma general de especificar los miembros de un conjunto
 - Utilizando una expresión booleana
 - $\{n: \mathbb{N} \mid n < 3 \cdot n\}$
- ◆ Componentes
 - Signatura ($n: \mathbb{N}$)
 - Intervalo de valores que se considerarán cuando se forme el conjunto
 - Predicado ($n < 3$)
 - Expresión booleana que indica como se debe construir el conjunto
 - Término (n)
 - Indica la forma general de los elementos del conjunto

Preliminares matemáticos



■ Operadores de conjuntos

◆ Pertenencia de conjuntos (\in , \notin)

- $12 \in \{6, 1, 12, 22\} \rightarrow \text{True}$
- $5 \notin \{6, 1, 12, 22\} \rightarrow \text{True}$

◆ Inclusión (\subseteq , \subset)

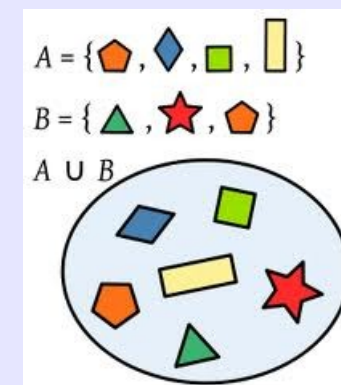
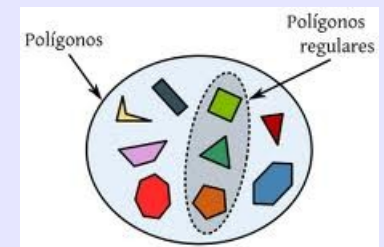
- $\{1, 2\} \subset \{4, 3, 1, 2\} \rightarrow \text{True}$
- $\{\text{HD1}, \text{LP4}, \text{RC5}\} \subset \{\text{HD1}, \text{RC2}, \text{HD3}, \text{LP1}, \text{LP4}\} \rightarrow \text{False}$

◆ Unión

- $\{1, 2, 3\} \cup \{2, 4, 6\} = \{1, 2, 3, 4, 6\}$

◆ Intersección

- $\{1, 2, 3\} \cap \{2, 4, 6\} = \{2\}$



Preliminares matemáticos



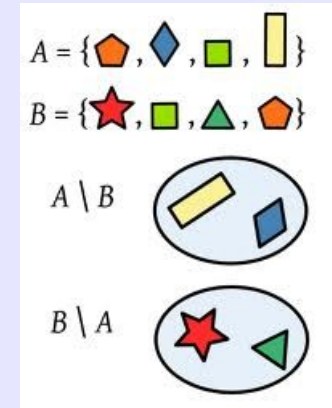
■ Operadores de conjuntos

◆ Diferencia de conjuntos

- $\{1, 2, 3, 4\} \setminus \{2, 4\} = \{1, 3\}$

◆ Producto cruzado

- $\{1, 2, 3\} \times \{4, 5\} = \{(1,4), (1,5), (2,4), (2,5), (3,4), (3,5)\}$



■ Conjuntos especiales

◆ Conjunto vacío (\emptyset)

- No tiene ningún elemento

◆ Conjunto potencia (P)

- Colección de todos los posibles subconjuntos
- $P\{1,2,3\} \rightarrow \{\emptyset, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}\}$

Preliminares matemáticos



■ Operadores lógicos

- ◆ Álgebra de expresiones verdaderas y falsas

Y	\wedge
O	\vee
No	\neg
Implica	\Rightarrow
Cuantificación Universal	\forall
Cuantificación Existencial	\exists

- ◆ La cuantificación universal es una forma de hacer una afirmación acerca de los elementos de un conjunto que es verdadera para todos los miembros del conjunto

$$\forall i, j: \mathbb{N} | i > j \Rightarrow i^2 > j$$

Preliminares matemáticos



■ Sucesiones

- ♦ Estructura matemática que modela el hecho de que sus elementos estén ordenados
- ♦ Conjunto de pares, cuyos elementos varían de 1 al elemento de mayor número
 - $\{(1, \text{Jones}), (2, \text{Wilson}), (3, \text{Shapiro}), (4, \text{Estévez})\}$
- ♦ Dominio e intervalo de una sucesión
 - $\langle \text{Jones}, \text{Wilson}, \text{Shapiro}, \text{Estavez} \rangle$
- ♦ Sucesión vacía
 - $\langle \rangle$
- ♦ Se admiten duplicados y el orden importa
 - $\langle \text{Jones}, \text{Wilson}, \text{Wilson} \rangle \neq \langle \text{Wilson}, \text{Jones} \rangle$

Preliminares matemáticos



■ Sucesiones

◆ Operadores

- Concatenación (^) →

- Añade al final de una sucesión otra
- $\langle 2, 3, 34, 1 \rangle \wedge \langle 12, 33, 34, 200 \rangle = \langle 2, 3, 34, 1, 12, 33, 34, 200 \rangle$

- Cabeza

- Extrae el primer elemento de una sucesión

- Cola

- Devuelve los últimos $n-1$ elementos en una sucesión de longitud n

- Ultimo

- Extrae el elemento final de una sucesión

- Frente

- Proporciona los primeros $n-1$ elementos en una sucesión de longitud n