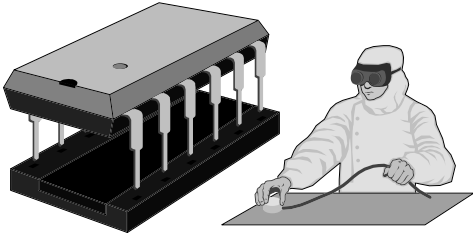


Cleanroom Software Engineering

Harlan Mills, 1987



Horizontal lines for notes.

Cleanroom Software Engineering

- **IC manufacturing metaphor**
 - Applied to software engineering
- **Defect identification**
 - Ascribed to process, not product
- **Correction**
 - Process fixed, product discarded

Horizontal lines for notes.

Cleanroom Process

- **Incremental life cycle**
- **Formal methods**
 - Specification and design
- **Develop without execution**
- **Independent testing**
 - Statistically based

Horizontal lines for notes.

Cleanroom Life Cycle

- **Developed in increments**
- **Assessed for reliability**
 - Test results fed back to development
- **Structured specification**
 - Decomposed into subspecifications

Cleanroom Process

- **Timeboxing**
 - Short interval scheduling
 - Prevent "feature creep"
 - Separate deliverables per interval
- **Functional verification**
 - "Prove" correct before release (test)

Formal Methods

- **Begin with formal spec**
- **Functional verification**
 - Mathematical reasoning
 - Sets and functions
- **Verifiability**
 - Programs written to assist verification

No Program Execution

- **Emphasis on verification**
 - And comprehensibility
- **Developers may not debug!**
 - Forbidden to execute their code!
- **Inspections, verification**
 - Individual and group

Cleanroom Testing

- **Goal is to assess quality**
 - Not to find defects!
- **Independent test team**
- **Test data from specifications**
- **If quality not OK, rework!**
 - Start the development process over

Formal Specifications

- **Set theory**
- **Predicate calculus**
- **Languages**
 - Z ("zed")
 - VDM (Vienna development method)
 - Larch

Z Notation

- **Schema**
 - State space and operations
 - Predicate constraints on variables
- **Operations**
 - Changes in state

An Introduction to Discrete Mathematics, Formal System Specification, and Z, D.C. Ince

Z Notation

- $S:P X$ S is a set of X's (F for finite)
- $x \in S$ x is a member of S
- $x \notin S$ x is not a member of S
- $S \subseteq T$ S is a subset of T
- $S \cup T$ Union of S and T
- $S \cap T$ Intersection of S and T
- $S \setminus T$ Difference of S and T
- \emptyset Empty set

Z Example

- **File system (like UNIX)**
- **Files are sequences of bytes**
- **File header status**
- **File ID allocation**
- **File operations return status**

Z Example: Declarations (1)

Clients, status, data unit
[UserNum, Report, ByteVal]

GuestNum : UserNum ← Guest user number

MaxFileSize : ℕ ← Maximum file size

From *Formal Specification and Documentation using Z*, Jonathan Bowen, University of Reading

Z Example: Declarations (2)

File data is sequence of bytes
Data == {s : seq ByteVal | #s ≤ MaxFileSize}

Size == 0..MaxFileSize
← Valid file size

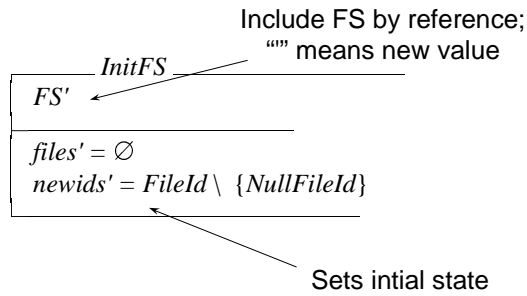
Z Example: Declarations (3)

Times represented as natural numbers
Time == ℕ

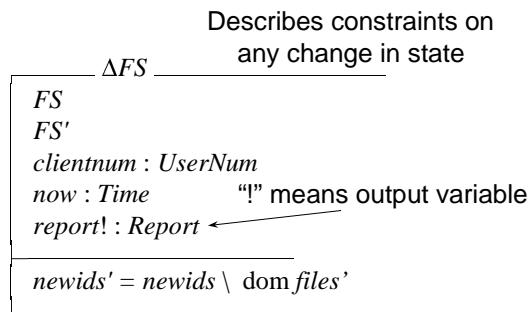
Define ordering relationship on times
<: Time ↔ Time

(_<_) ∈ total_order
← There is a total_order on time

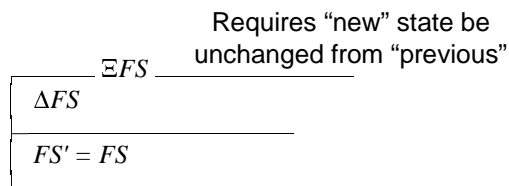
Z Example: Initial State



Z Example: Change of State



Z Example: Unchanged State



Can be referenced in another
 schema to specify that state will
 not be changed
