Software Engineering: Where have we been, where are we going?

WSA / SIRTI
Spokane, WA
November 14, 2000

Frederick T. Sheldon
Assistant Prof. of Computer Science
Washington State University

Agenda

- SE is process & product attributes
- Verification and validation
  - SLC expenditure profile
- The right process for your product
  - High quality SW with competitive cost & cycle time
- State of Software Eng. (...a miserable science?)
- Out with the Bathtub in with the Shower Model
  - *Lightweight* vs *heavyweight* Methodologies
- Where do we go from here?
  - Some thoughts on SE Education

Who am I?

- WSU School of EECS, CU, UTA
- NASA Langley and Ames Research Centers in various capacities
- Software Engineer
  - Avionics + diagnostics software development for the YF-22 and F-16 (Stores) at GD
  - Tornado aircraft nose radar at TI

Software Engineering

- ***Software engineering** is concerned with the theories, methods and tools for developing, managing and evolving software products*
  - **The economies of ALL developed nations are *dependent* on software**
  - **More and more systems are software controlled**

Software Costs

- **Software costs often *dominate* system costs (e.g., PC: SW Costs > HW Cost)**
- **Software costs more to maintain**
  - **Systems with long life: maintenance costs may be several times development costs**
  - **Roughly 60% development 40% testing (custom SW evolution costs often > development costs**
- **Software engineering is concerned with cost-effective software development**

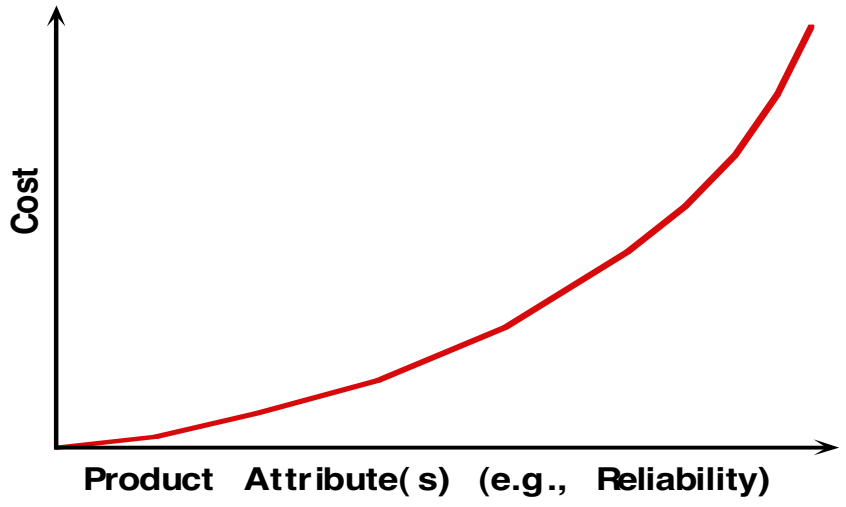
Software Products

- **Generic (COTS) products**
  - **Stand-alone systems produced by a development organization (sold open market to any customer)**
- **Bespoke (Custom) produces**
  - **Systems commissioned by a specific customer and developed specially by some contractor**
- **Fact: Most software expenditure is on generic products but... *most development effort is on bespoke systems***

Software Product Attributes

- **Maintainability, Dependability, Efficiency, Usability, Extensibility,….**
- **Some cases attribute(s) may dominate**
  - ***In safety-critical real-time systems, key attributes may be dependability and efficiency***
- **Costs tend to rise exponentially if very high levels of any one attribute are required**

Attribute vs Cost Relation



Def: Software Process

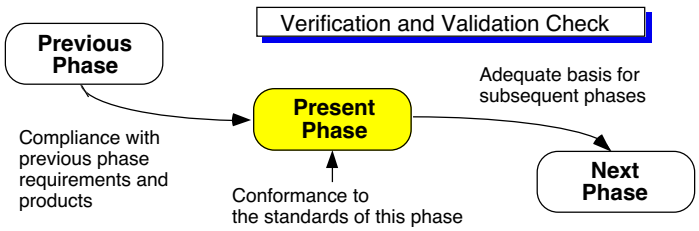
- **Structured set of activities required to develop a software system**
  - ***Specification***
  - ***Design***
  - ***Validation***
  - ***Evolution***
- **Activities vary depending on the organization and the type of system being developed**
- ***Must be explicitly modeled to be effectively managed***

How do processes work?

What is a software process...

Process consists of:	Activities	√ Things that are done
	Products	√ Inputs and Outputs
	Sequencing	√ Relationships among the activities and products

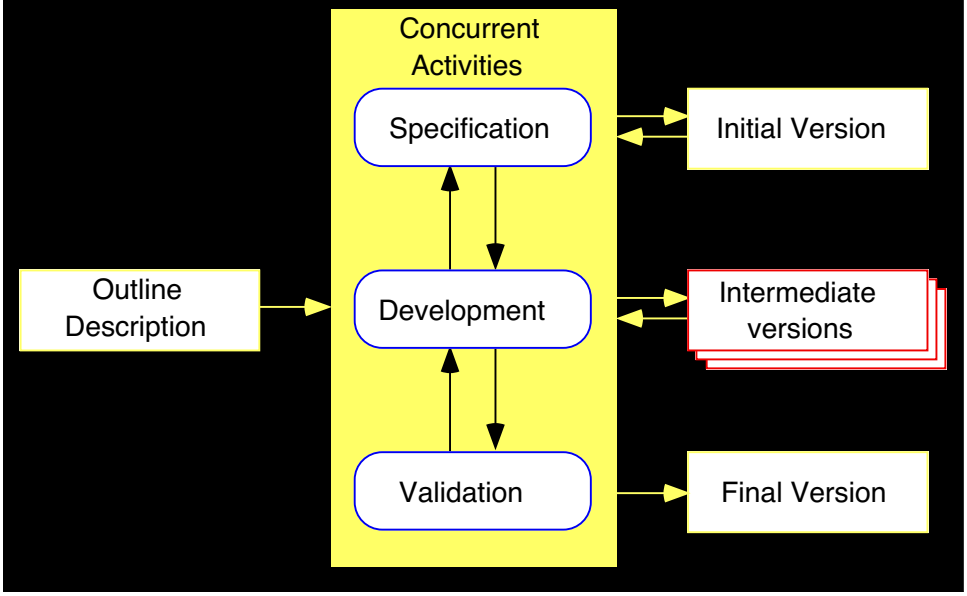
...and how does sequencing work?



Generic Process Models

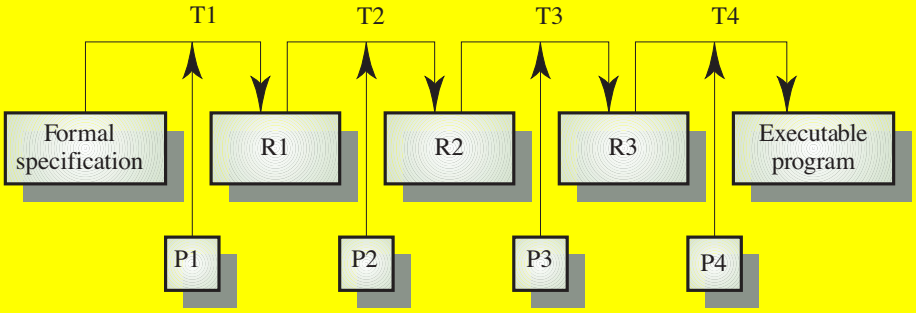
- **The waterfall model**
  - Separate and distinct phases of specification and development
- **Evolutionary development**
  - Specification and development are interleaved (e.g., Extreme Prgming)
- **Formal transformation**
  - Mathematical system model formally transformed to an implementation
- **Reuse-based development**
  - The system is assembled from existing components
- **Hybrid Methodologies**
  - Prototyping for high-risk specifications (e.g., Spiral Model)
  - A heavyweight methodology has many rules, practices, and documents.
  - A lightweight methodology few rules / practices → easy to follow.

Evolutionary development



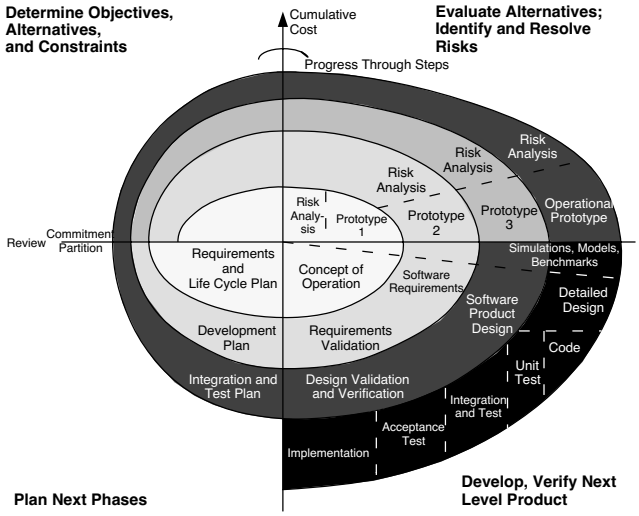
Transformational development

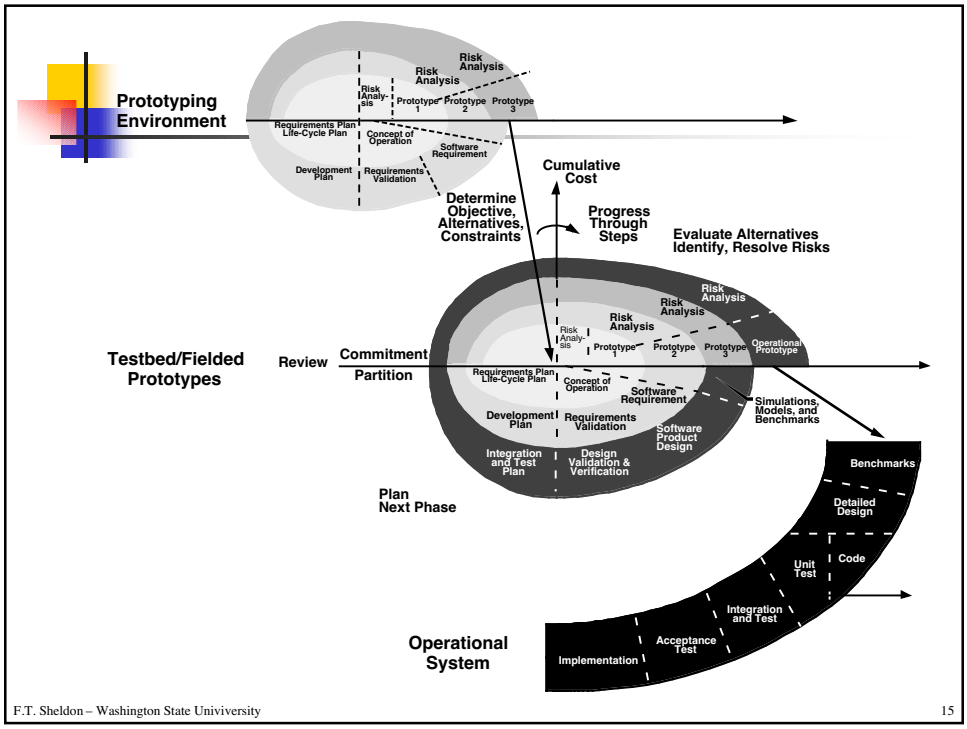
Formal Transformations ⇒



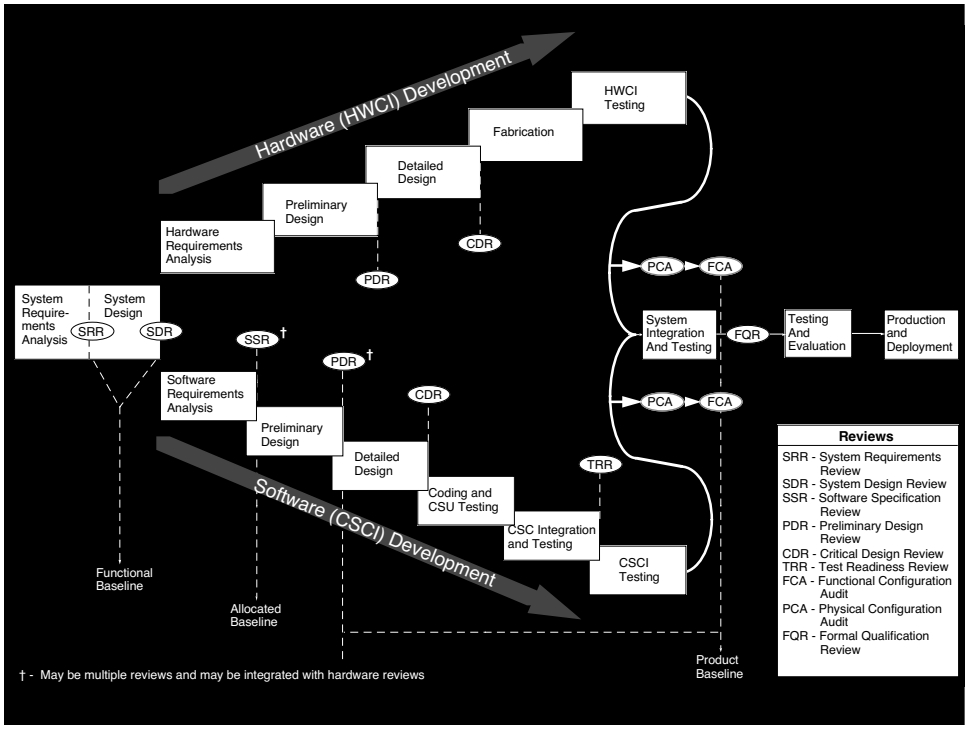
Proof of Transformation Correctness

Spiral Model – Boehm





F.T. Sheldon – Washington State Univiversity 15

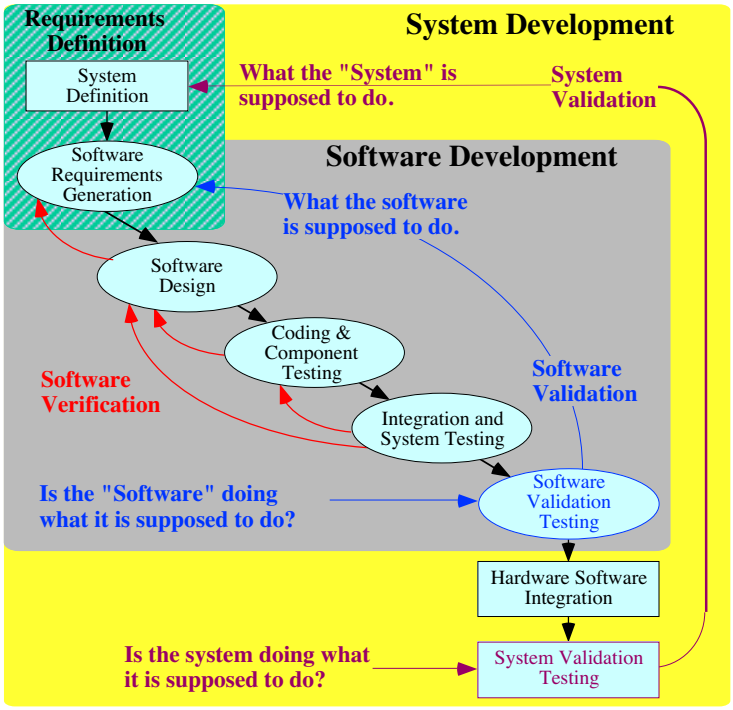


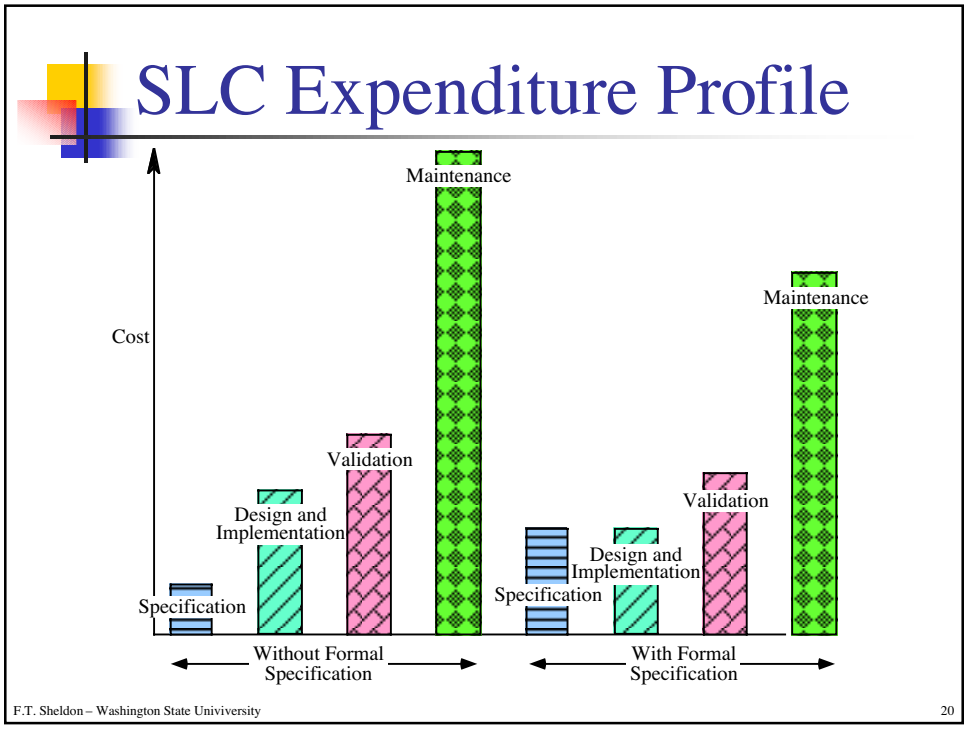
CMM

Level	Characteristics	Key Challenges	Result
5 Optimizing	• Focus on process improvement • Data gathering is automated and used to identify weakest process elements • Numerical evidence used to justify application of technology to critical tasks • Rigorous defect-cause analysis and detect prevention • Focus on process optimization to reduce errors	• Still human-intensive process • Maintain organization at optimizing level	Productivity & quality
4 Managed	(Quantitative) • Measured process: estimates/actuals, error-cause analysis • Minimum set of quality and productivity measurements established • Process database established & resources to analyze & maintain its data • Focus on technology management and insertion	• Changing technology • Problem analysis • Problem prevention	
3 Defined	(Quantitative) • Process defined and institutionalized • Software Engineering Process Group established to direct improvements • Focus on Software design skills, design tracking • Focus on various types of traceability	• Process measurement • Process analysis • Quantitative quality plans	
2 Repeatable	(Intuitive) • Process dependent on individuals • Established basic project controls with strength in doing similar work • Process faces major risk when presented with new challenges • Lacks orderly framework for improvement • Focus on collecting various types of "trend" data • Focus on management and tight project control	• Training • Technical practices (reviews, testing) • Process focus (standards, process groups)	
1 Initial	(Ad hoc/chaotic process) • No formal procedures, cost estimates, project plans • No management mechanism to ensure procedures are followed, tools not well integrated, and change control is lax • Senior management does not understand key issues	• Project management • Project planning • Configuration management • Software quality assurance	Risk

Verification and *Validation*

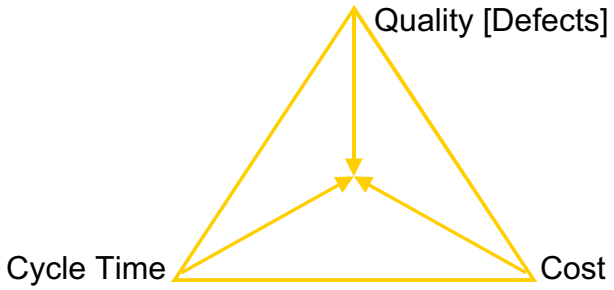
- **Verification** determines if the products of a given phase of the **SLC** fulfill the **requirements** established during the previous phase.
  - Formal proof of **program** correctness
  - Reviewing, inspecting, **testing**, checking, auditing, or otherwise establishing and documenting whether or not items, **processes**, services, or **documents** conform to specified **requirements** (ANSI/ASQC A3-1978).
- **Validation** checks ***if*** the program, as implemented, meets the expectations of the customer in such a way to ensure compliance with software requirements.


Requirements Definition
System Development
System Definition
What the "System" is supposed to do.
System Validation
Software Requirements Generation
Software Development
What the software is supposed to do.
Software Design
Coding & Component Testing
Software Verification
Software Validation
Integration and System Testing
Is the "Software" doing what it is supposed to do?
Software Validation Testing
Hardware Software Integration
Is the system doing what it is supposed to do?
System Validation Testing
19


SLC Expenditure Profile
Cost
Maintenance
Validation
Design and Implementation
Specification
Without Formal Specification
Maintenance
Validation
Design and Implementation
Specification
With Formal Specification
F.T. Sheldon – Washington State Univiversity
20

Right process for the product

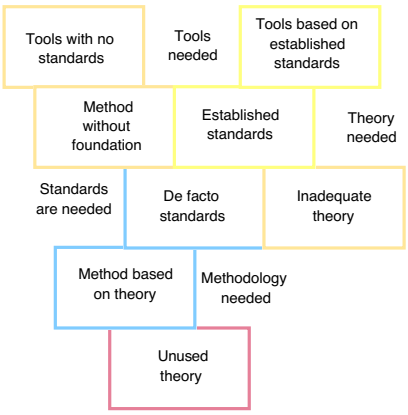
- High quality software with competitive cost and cycle time...



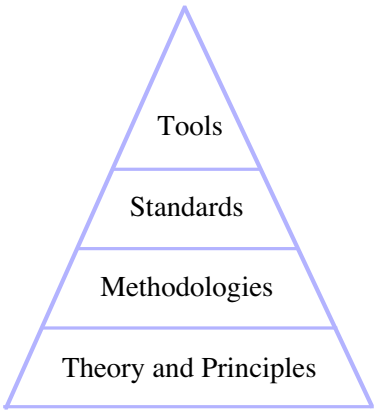
We must Shrink the Triangle

The State of SW Engring

Today



Goal



Thoughts on SE Education

- Software development is not mainstream in CS departments...
  - Typically one SE course is stuffed to overflowing with all the issues that arise in practice
  - Textbooks that are typically used for SE often devote only 5-10 pages to many topics that each should be the subject of a whole course

Opinion

- Some new standards require process related research, in part due to a *lack of a solid foundation* on which to move forward
- The issue of how to relate the impact of a system failure in its operating environment *back to characteristics about the process used to produce the software* for that system is not well known

What is needed in the Curriculum?

- Undergraduate
  - Sound CS fundamentals with SE support
- Graduate
  - Research in SE fundamentals
  - PhD programs in SE
  - *Masters programs for practitioners*
  - *Continuing education for practitioners*

UG Curriculum to support SE

- Fundamentals of CS
  - Algs, Discrete Math, Prgming, Systems, etc.
    - Beware of too much emphasis on current fashions
  - Networks, AI, CASE, OO,…
- Principles of Programming Langs.
  - More important than specific languages
- Experience on realistic team oriented projects
- Communication (oral+written), Ethics,..
- Some domain other than computing

Grad SE Curriculum

- Practical experience is strongly recommended
- Design and synthesis of software systems
- Management of SE Projects
- SW testing, and quality engineering
- Reuse and reengineering
- SW metrics and SE process
- The list can go on...

SE Research Agenda

- All topics on the previous page
- *Formal modeling and analysis of SW Reqs*.
- Automatic code generation from designs
- Automatic generation of ancillary products
  - Specifications, end-user documents, test code, test cases, maintenance support code, etc.
- Process modeling, analysis and enactment
- SW Arch., evolution, SW for critical systems, CHI, CM

Where do we go from here?

- SE is interdisciplinary –this goes against the grain of the academic establishment
- Its engineering (not science) –lives depend on software being created in a professional and competent manner
- SE is much more mature than it was 30 years ago… ***but a lot less mature than it needs to be!***
- This is where the jobs are and will be… *so this is where the are or will be…*