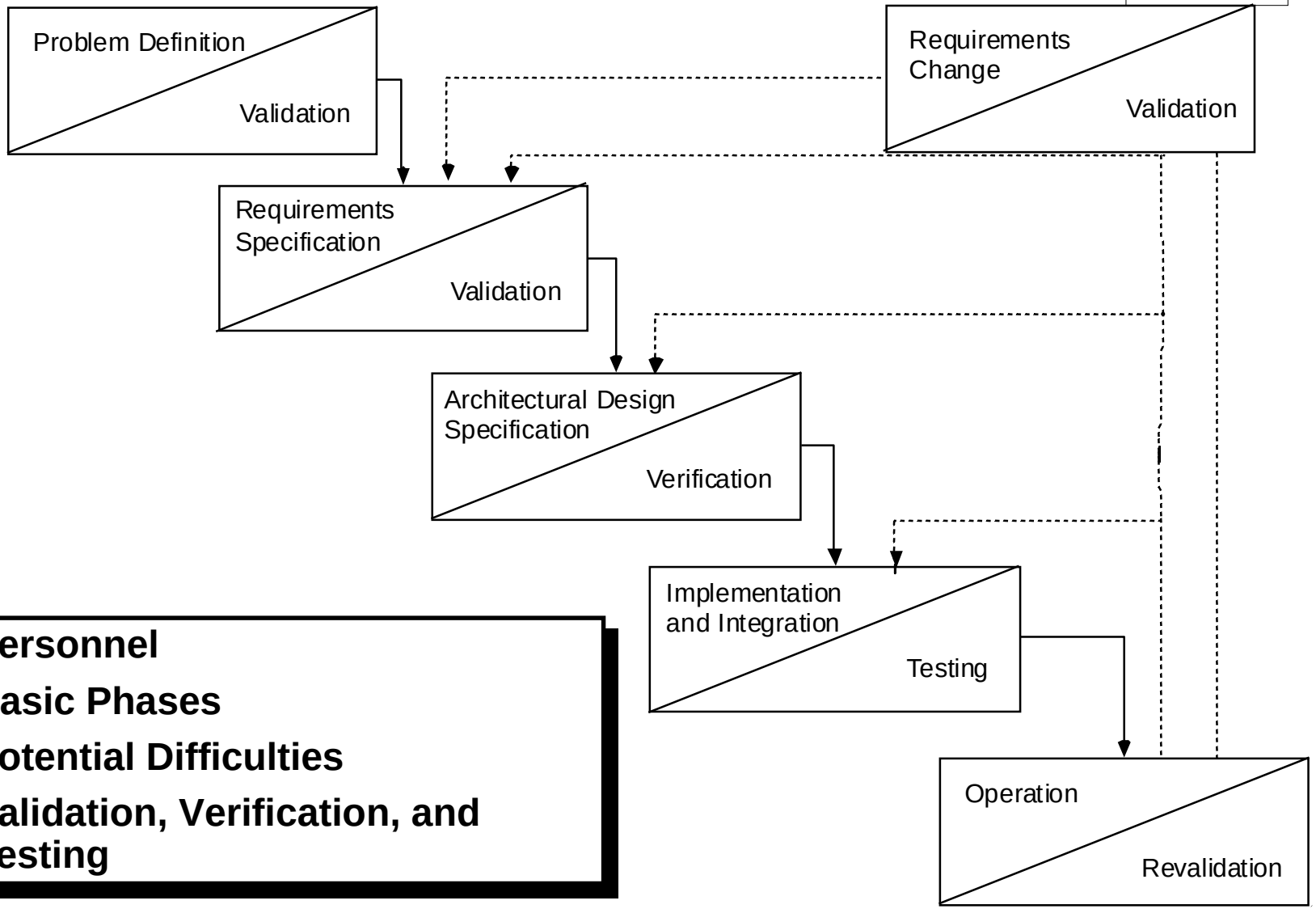
Software Production and Lifecycle Models



- **Personnel**
- **Basic Phases**
- **Potential Difficulties**
- **Validation, Verification, and Testing**

Problems: Essence and Accidents

- Software is conceptual (intangible)
- Essence: difficulties inherent in the intrinsic nature of software
- Accidents: difficulties encountered today, but not inherent in software production
- Accidents are amenable to research breakthroughs
- Essence constitutes those problems that are unsolvable
  - complexity
  - conformity
  - changeability
  - invisibility

No Silver Bullet !

Software Production Personnel

- Client – individual or organization that want product to be developed
- Developer(s) – (members of) organization producing product
- User – person on whose behalf client has commissioned developer, person(s) who will utilize software in operation
- internal software development: client = developer
- contract software development: client ° developer

QUALITY PRODUCTS THROUGH PROCESS

- **Quality Software Products developed through**
  - **systematic software processes**
  - **with explicit product quality requirements**
- **Effective testing and analysis must be included**
  - **incremental analysis activities**
  - **to complement synthesis activities**
- **Powerful tools and processes are essential to assure effectiveness**

- **Process Models**
- **Processes**

What is a Process?

- Device for producing a product (getting job done)
- Indirect nature
  - Process description (program) created to describe wide class of instances
- Humans create process descriptions (models or programs) to solve classes of problems
- Software Processes:

  devices for creating and evolving software products

The Lifecycle Approach

- Phasing promotes manageability and provides organization
- Reviews assure ultimate satisfaction of requirements
- Intermediate products promote visibility and assure continuity between phases

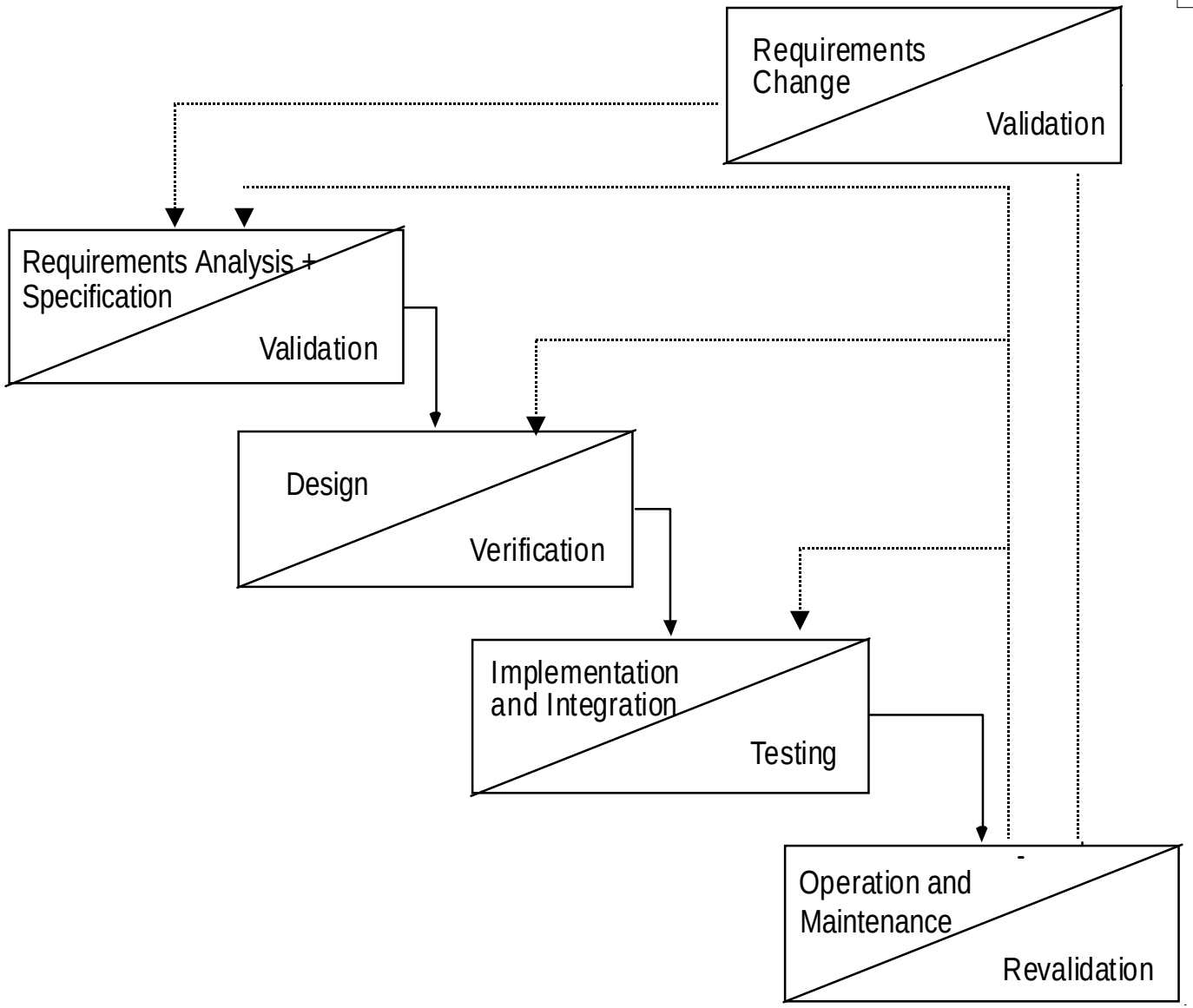
Major Components of a Lifecycle Model:

- Phases
- Reviews
- Intermediate Products

Intermediate Software Products

- **Objectives:**
  - **Demarcate end of phases**
  - **Enable effective reviews**
  - **Specify requirements for next phase**
- **Form:**
  - **Rigorous**
  - **Machine processible**
- **Content:**
  - **Specifications**
  - **Tests**
  - **Documentation**

Phases of a SW Lifecycle Model



Requirements Analysis and Specification

- **Problem Definition —> Requirements Specification**
  - **determine exactly what client (and user) wants and process constraints**
  - **develop a contract with client**
  - **what task the product is to do**
- **Difficulties**
  - **client asks for wrong product**
  - **client is computer/software illiterate**
  - **specifications may be ambiguous, inconsistent, incomplete**
- **Validation**
  - **extensive specification reviews to check that requirements specification satisfies client needs**
  - **look for ambiguity, consistency, incompleteness**
  - **check for feasibility, testability**
  - **develop system/acceptance test plan**

Design

- **Requirements Specification —> Design**
  - **develop architectural design (system structure): decompose software into modules with module interfaces**
  - **develop detailed design (module specifications): select algorithms and data structures**
  - **maintain record of design decisions and traceability**
  - **how the product is to do its task**
- **Difficulties**
  - **miscommunication between module designers**
  - **design may be inconsistent, incomplete, ambiguous**
- **Verification**
  - **extensive design reviews (inspections with checklists) to determine that design conforms to requirements**
  - **check module interactions**
  - **develop integration test plan**

Implementation and Integration

- **Design —> Implementation**
  - **implement modules and verify they meet their specifications**
  - **combine modules according to architectural design**
  - **how the product does its task**
- **Difficulties**
  - **module interaction errors**
  - **order of integration has a critical influence on product quality and productivity**
- **Verification and Testing**
  - **extensive code reviews (inspections with checklists) to determine that implementation conforms to requirements and design**
  - **develop and test on unit/module test plan: focus on individual module functionality**
  - **test on integration test plan: focus on module interfaces**
  - **test on system test plan: focus on requirements and determine whether product as a whole functions correctly**

Operation and Maintenance

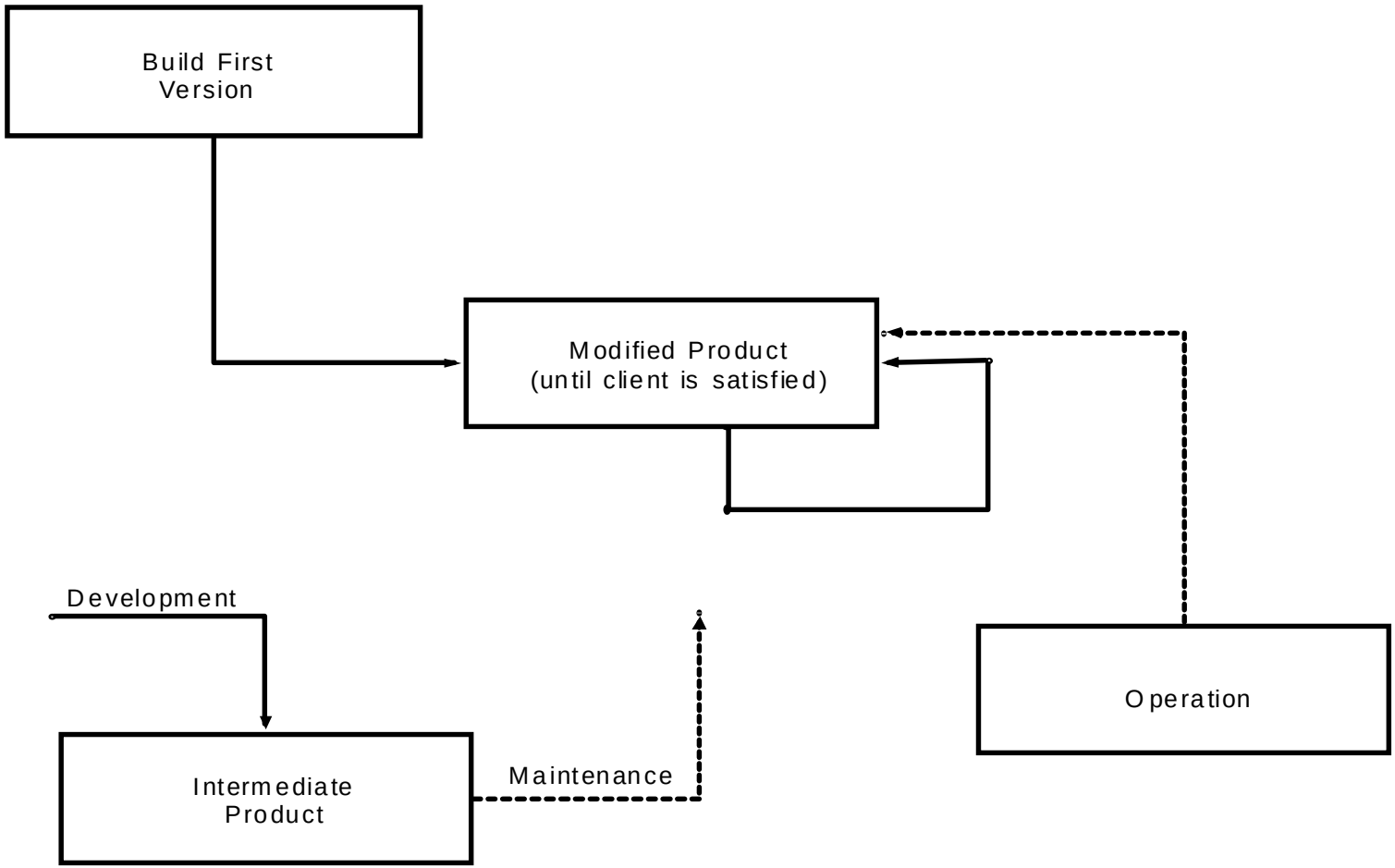
- **Operation —> Change**
  - maintain software after (and during) user operation
  - integral part of process
  - determine whether product as a whole still functions correctly
- **Difficulties**
  - design not extensible
  - lack of up-to-date documentation
  - personnel turnover
- **Verification and Testing**
  - extensive review to determine that change is made correctly and all documentation updated
  - test to determine that change is correctly implemented
  - test to determine that no inadvertent changes were made to compromise system functionality (check that no affected software has regressed)

Lifecycle Models

- Over time different lifecycle models were developed, e.g.,
  - build-and-fix model
  - waterfall model
  - prototyping model
  - incremental model
  - spiral model
  - ....
- Different lifecycle models decompose software engineering activities in different ways
- No "right" or "wrong" lifecycle model

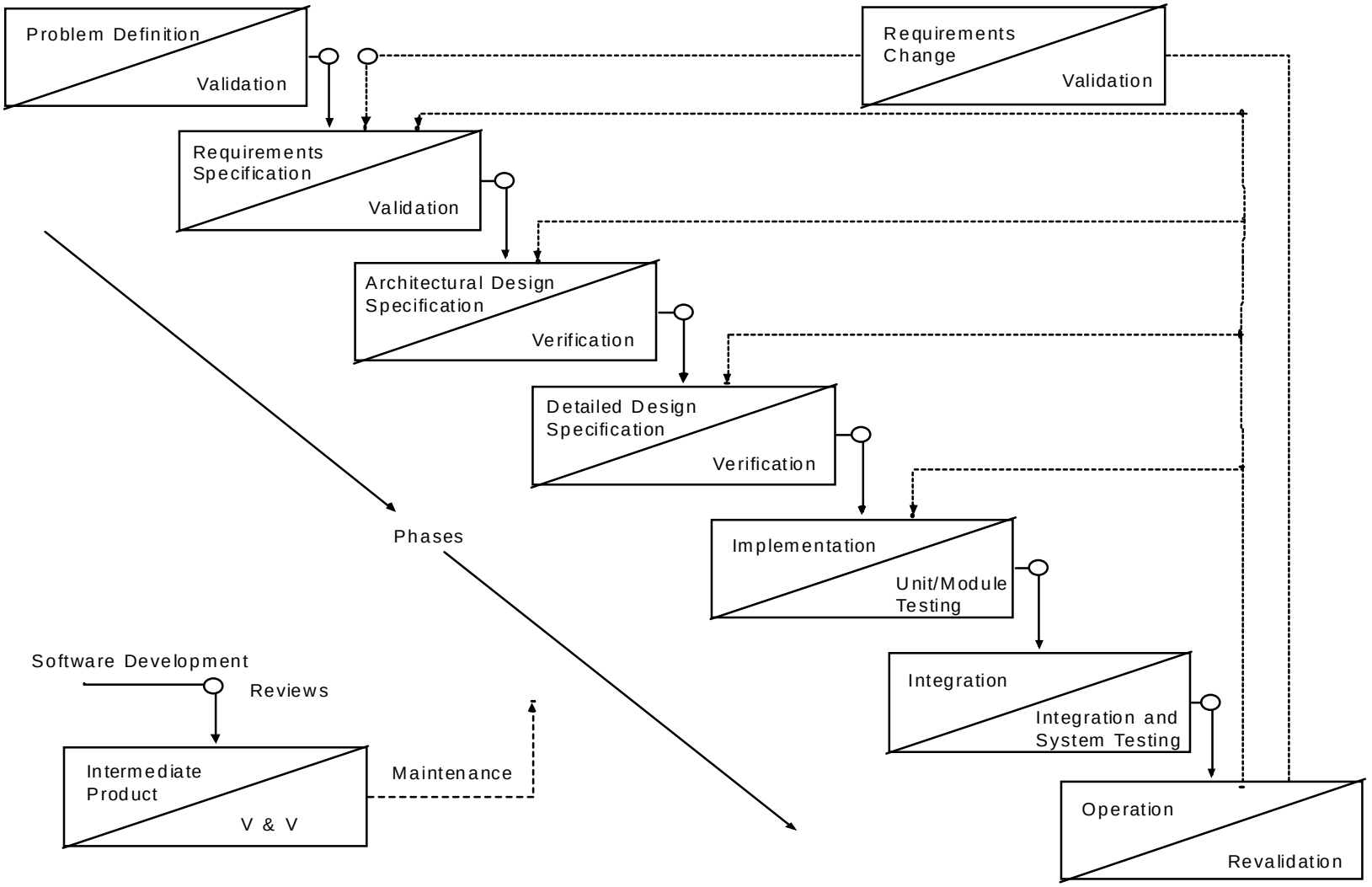
Build and Fix Approach

– **Build entire product; deliver to client who requires changes; change until client feels software can be used productively**



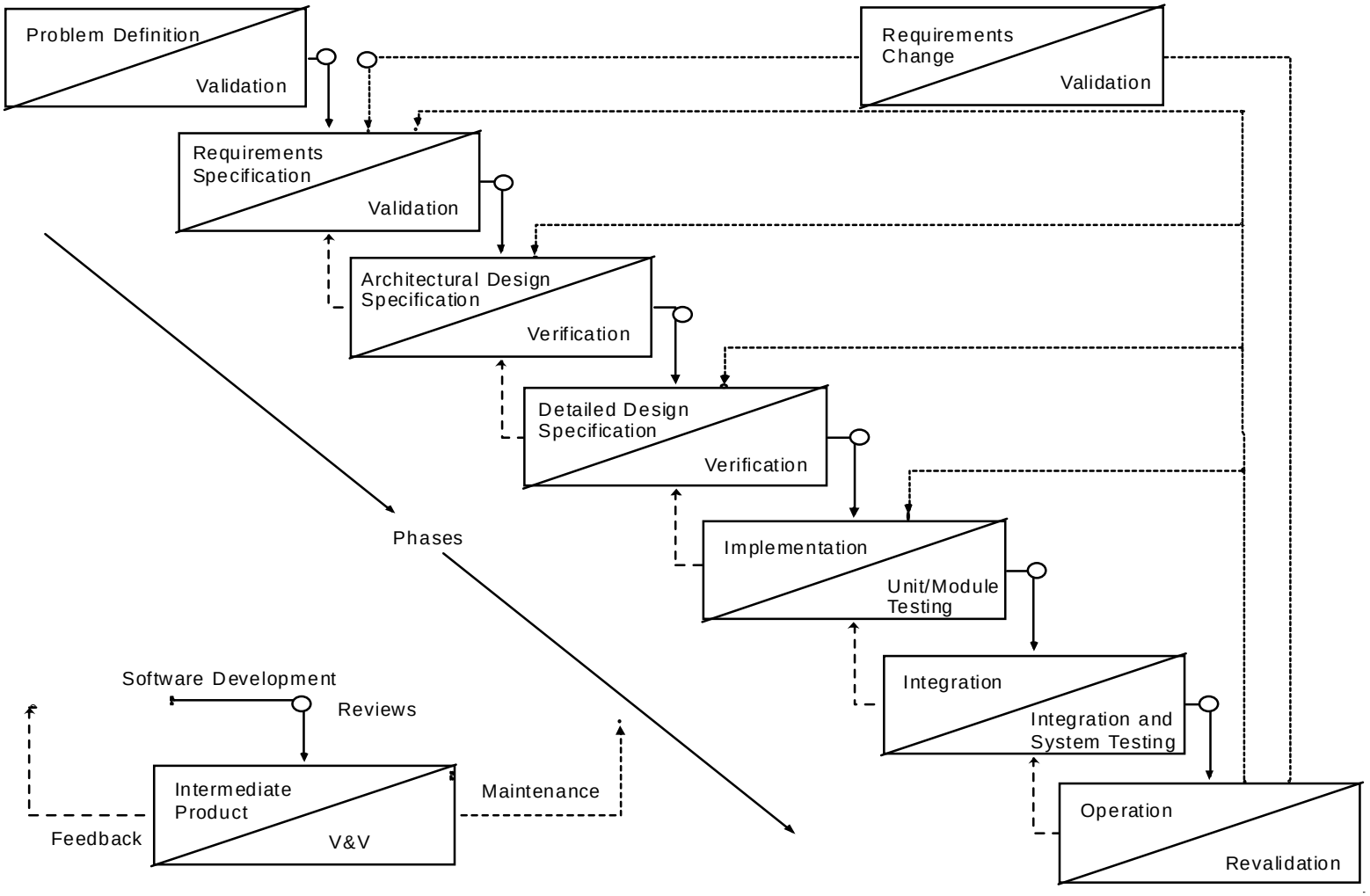
Stagewise Development

– **Software developed in successive stages (lifecycle phases)**



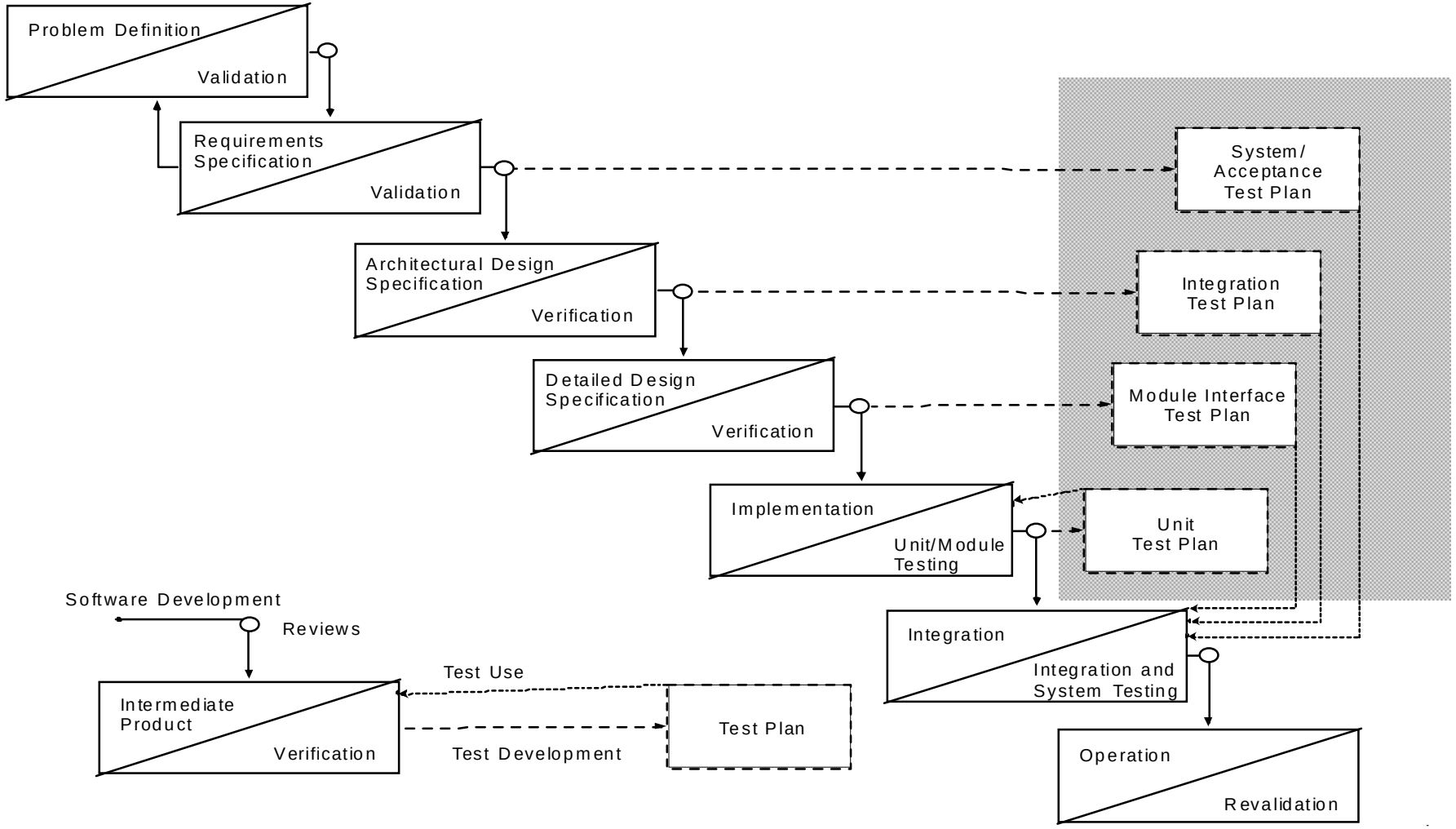
Waterfall Model [Royce,1970]

– **Includes feedback confined between successive phase to minimize impact**



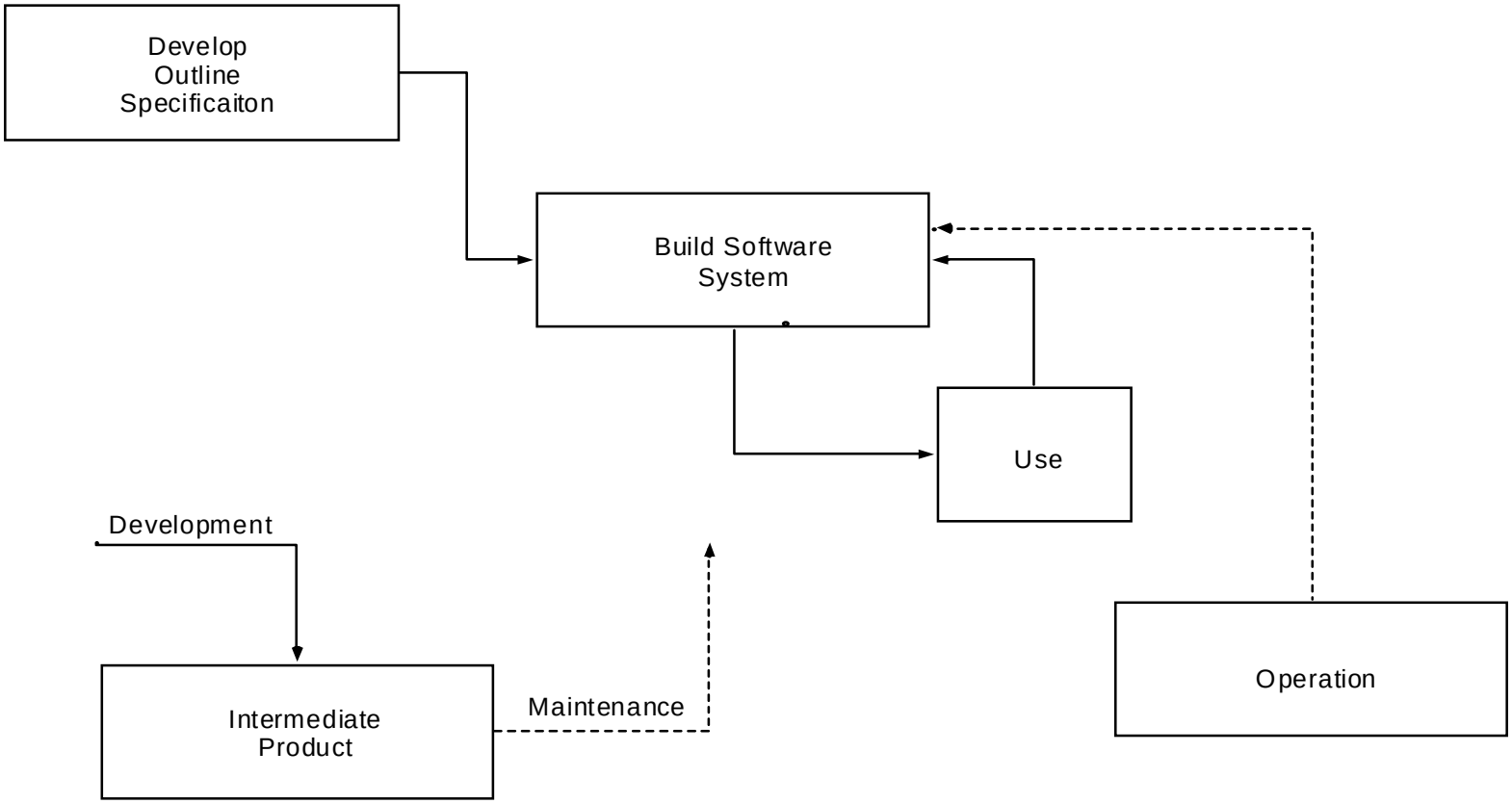
Test Development

– **Develop Test Plans in conjunction with each lifecycle phase**



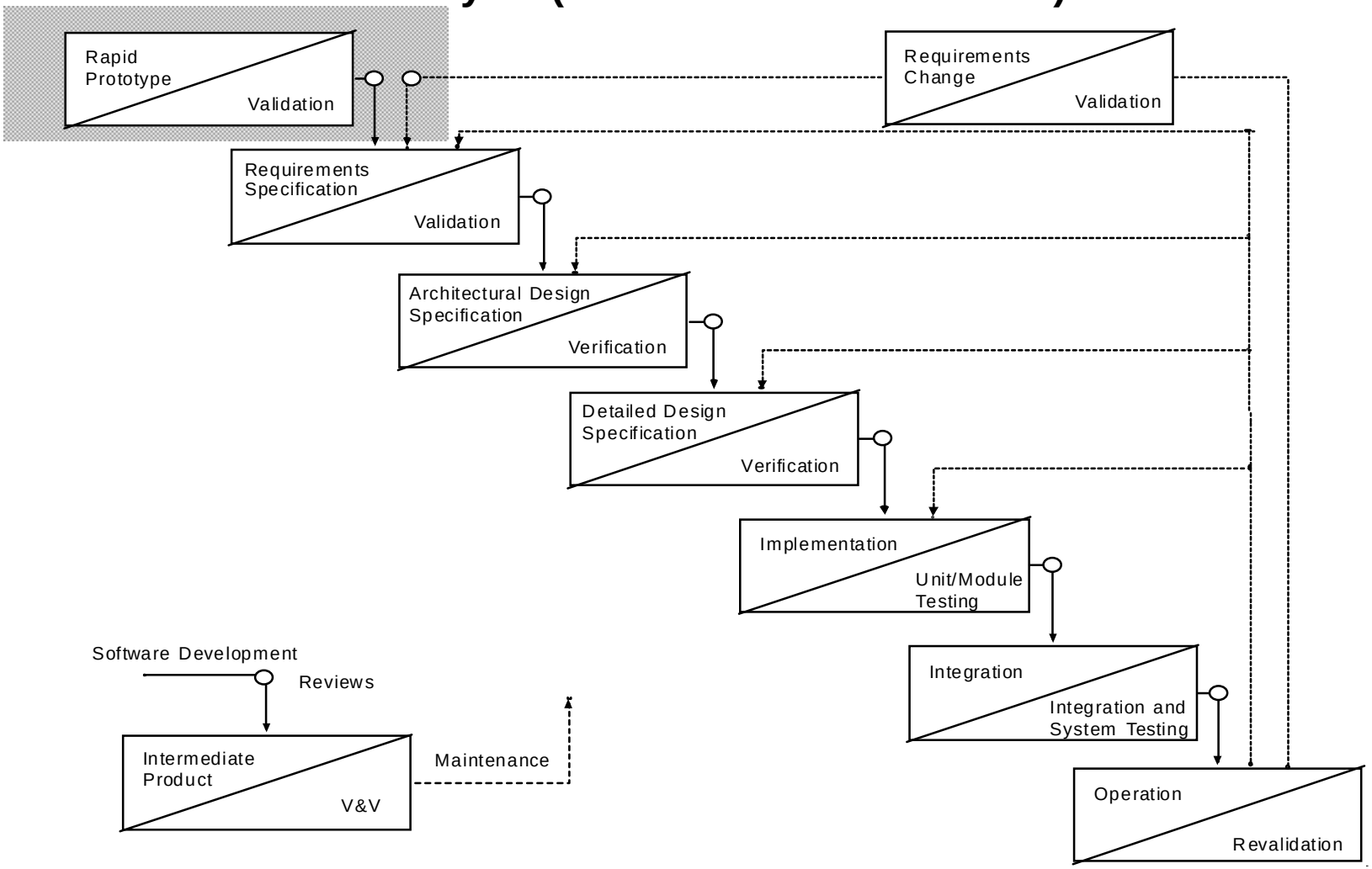
Exploratory Programming

- **Develop outline specification because full requirements are not known, build system and expose to user review, modify system until performance is adequate**



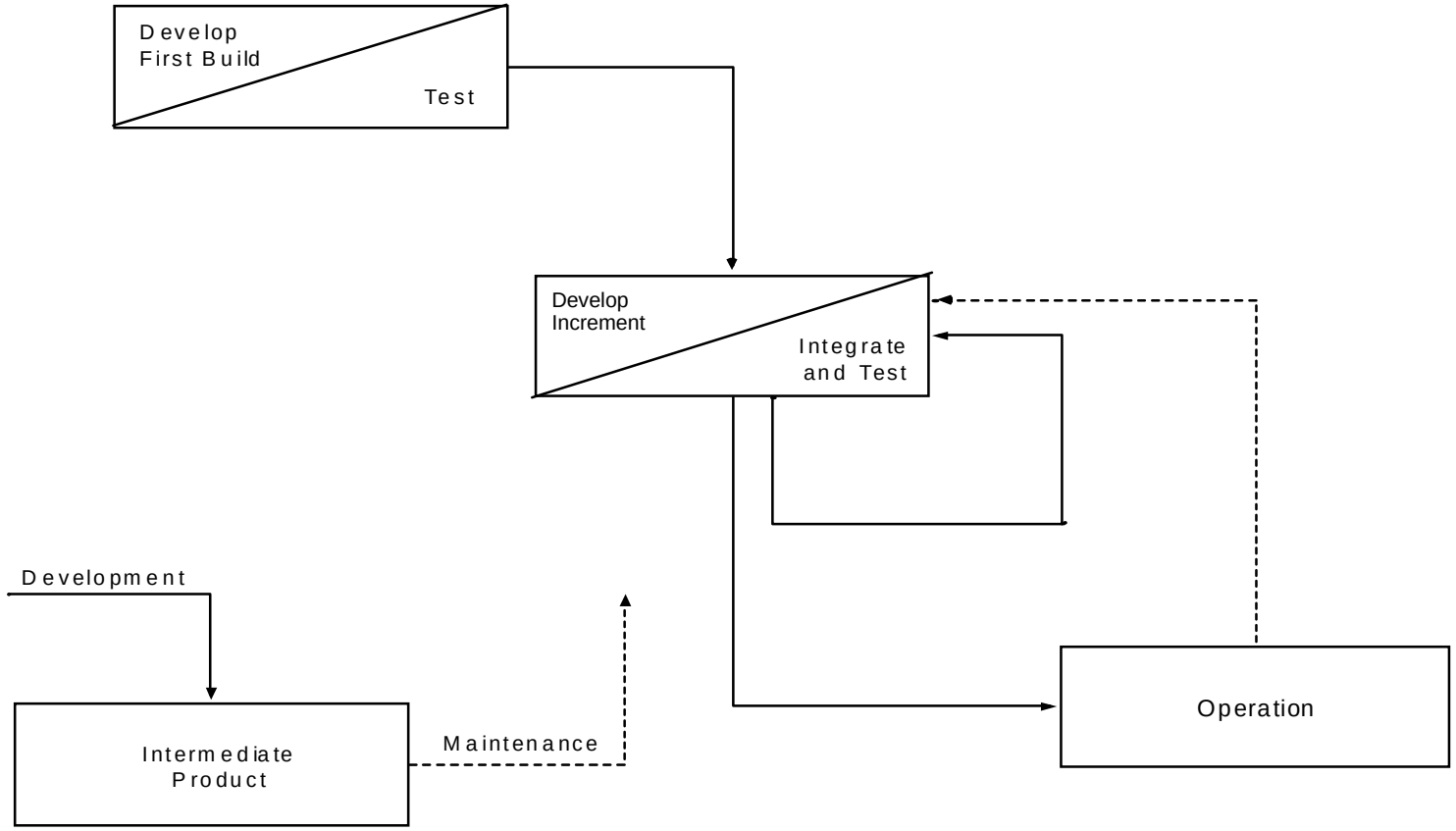
Prototyping Model

– **Develop prototype implementation to establish requirements, then follow traditional lifecycle (could also have feedback)**



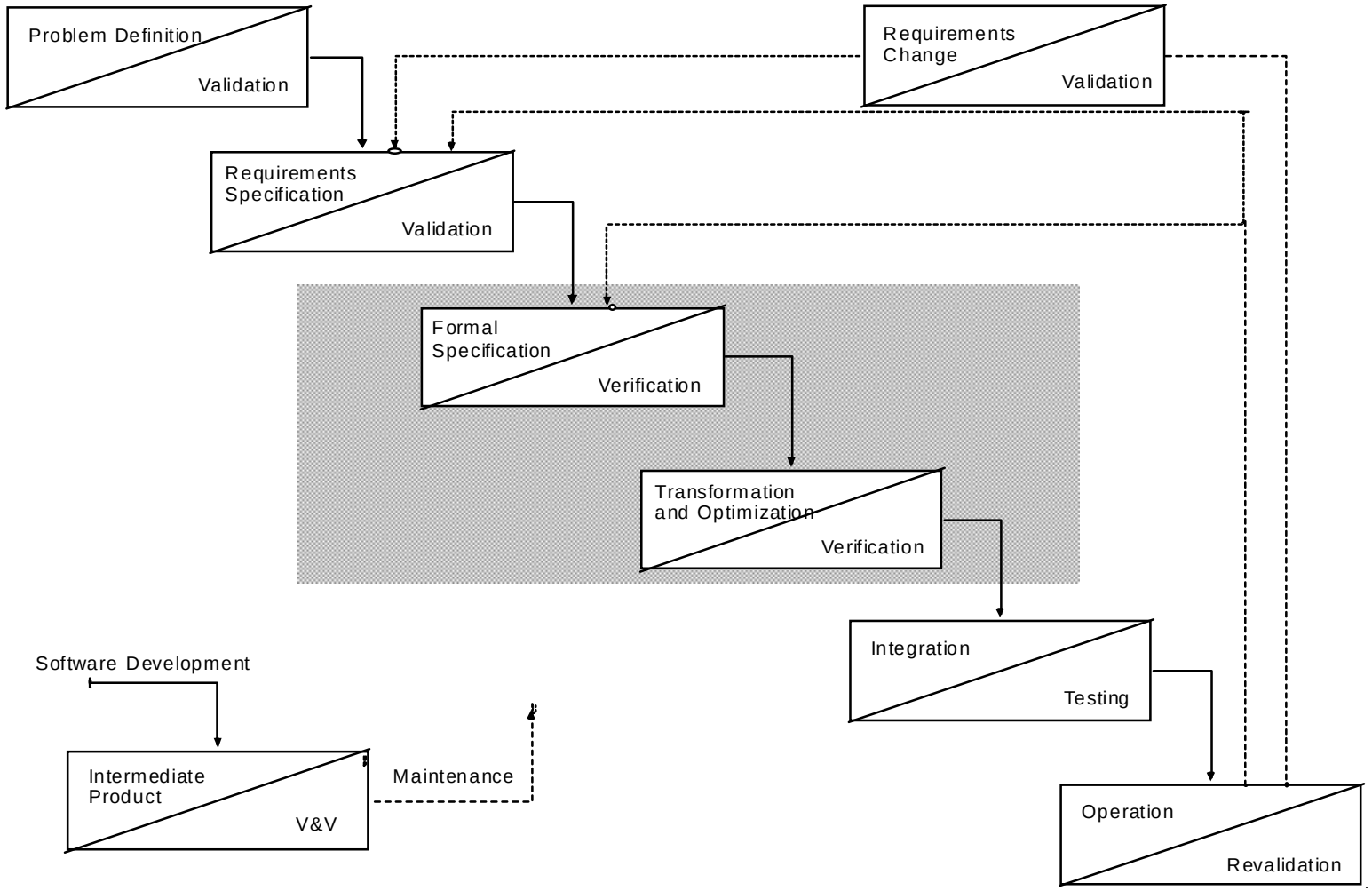
Evolutionary/Incremental Model

- **Develop first implementation, develop successive increments of an operational product until complete, direction of evolution determined by operational experience (development process should use waterfall model)**



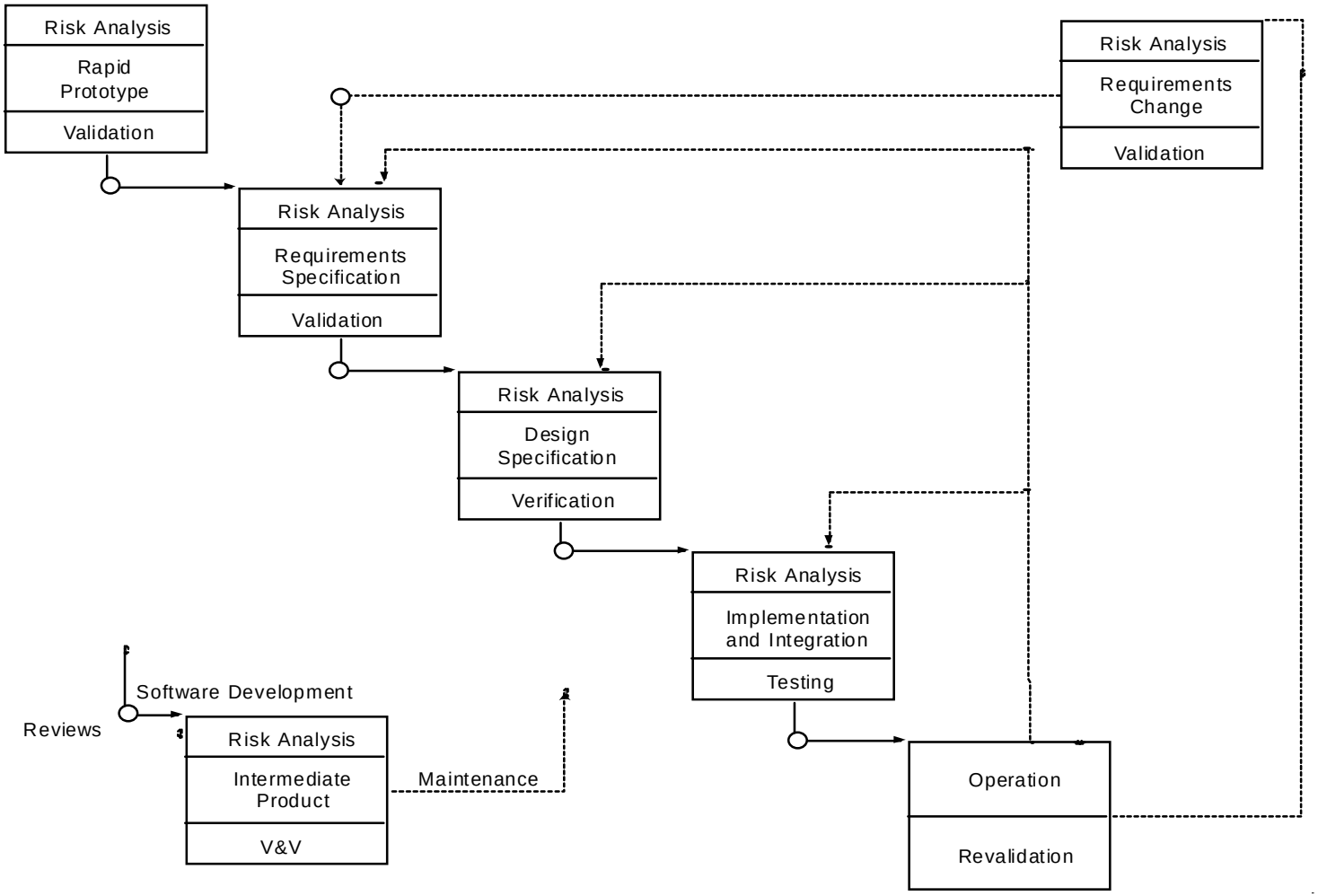
Transformation Model

– **Develop formal specification,transform into implementation using correctness-preserving transformations**

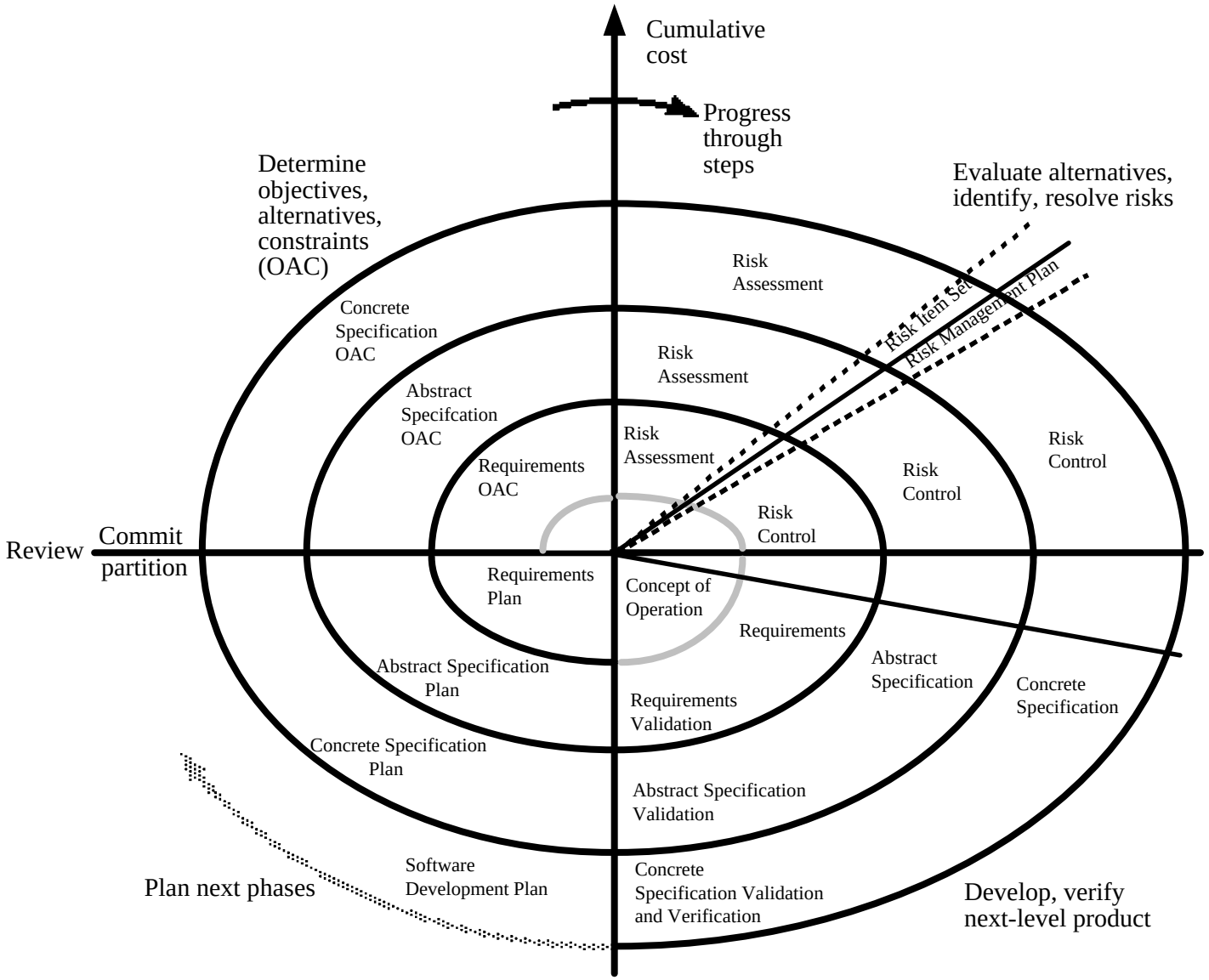


Simplistic View of Spiral Model

– Include risk analysis with each development phase



The Spiral Model [Boehm,1988]



Capability Maturity Model (CMM) [Watts Humphrey,1989]

- CMM is not a software lifecycle model
- Strategy for improving the software process regardless of the process model followed
  - Basic premise: the use of new software methods alone will not improve productivity and quality, but rather software management is in part the cause of problems
  - CMM assists organizations in providing the infrastructure required for achieving a disciplined and mature process
- Includes both,
  - technical and
  - managerial aspects of software production

Capability Maturity Model - 2

- **Five *maturity levels***
  1. **initial – ad hoc process**
  2. **repeatable process – basic project management**
  3. **defined process – process modeling and definition**
  4. **managed process – process measurement**
  5. **optimizing process – process control and dynamic improvement**

- **to aid in maturation, the SEI has a series of questionnaires and conducts process assessments that highlight current shortcomings**

ISO 9000

- **Further attempt to improve software quality based on International Standards Organization (ISO)**
- **ISO 9000 = series of five related standards**
  - **within ISO 9000 standard series ISO 9000-3 focuses on software and software development**
- **Basic features:**
  - **stress on documenting the process in both words and pictures**
  - **requires management commitment to quality**
  - **requires intensive training of workers**
  - **emphasizes measurement**
- **Adopted by over 60 countries (e.g., USA, Japan, European Union, ...)**
- **Company needs to be certified that its process complies with the ISO 9000 standard**

ICS 121 Lifecycle Model