Software Engineering 2UCCE501

Module 5

Module 5 Testing & Maintenance

5.1 Testing Concepts: Purpose of Software Testing, Testing Principles, Goals of Testing, Testing aspects: Requirements, Test Scenarios, Test cases, Test scripts/procedures,

5.2 Strategies for Software Testing, Testing Activities: Planning Verification and Validation, Software Inspections, FTR

5.3 Levels of Testing : unit testing, integration testing, regression testing, product testing, acceptance testing and White-Box Testing

5.4 Black-Box Testing: Test case design criteria, Requirement based Testing, Boundary value analysis, Equivalence Class Partitioning

5.5 Object Oriented Testing: Review of OOA and OOD models, class testing, integration testing, validation testing

5.6 Reverse & Reengineering, types of maintenance

Software Rejuvenation(transformation)

Carried out using 4 steps:

1. **Re-documentation**
   - Creation or revision of alternative representations of software
     - at the same level of abstraction
   - Generates:
     - data interface tables, call graphs, component/variable cross references etc.
2. **Restructuring**
   - transformation of the system's code without changing its behavior

Software Rejuvenation

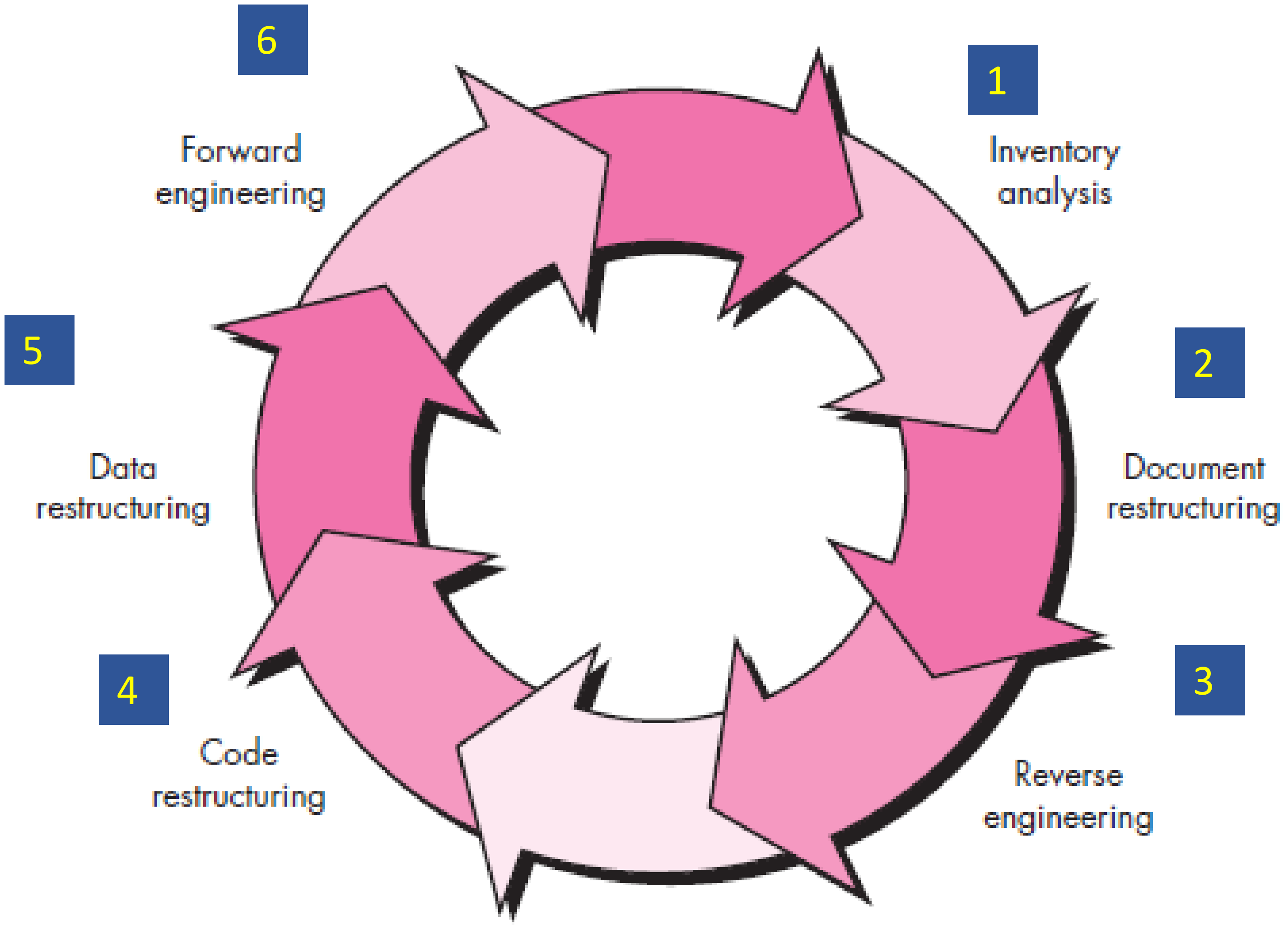
3. Reverse Engineering

- Analyzing a system to extract information about the behavior and/or structure
    - also Design Recovery - recreation of design abstractions from code, documentation, and domain knowledge
- Generates:
    - structure charts, entity relationship diagrams, DFDs, requirements models

4. Re-engineering

- Examination and alteration of a system to reconstitute it in another form
- Also known as renovation, reclamation
- Reengineering is a rebuilding activity.

Software Reengineering Process Model



Software Reengineering Process Model

1. **Inventory analysis.**

- The inventory can be nothing more than a spreadsheet model containing information that provides a detailed description (e.g., size, age, business criticality) of every active application.

- As status of application can change any time , inventory should be revisited on regular basis.

Software Reengineering Process Model

2. Document restructuring.

- Weak documentation is the trademark of many legacy systems.

1. ***Creating documentation is far too time consuming. –*** static programs
2. ***Documentation must be updated, but your organization has limited resources –*** re-document only changed portion
3. ***The system is business critical and must be fully re-documented*** - Even in this case, an intelligent approach is to pare (restrict) documentation to an essential minimum.

Software Reengineering Process Model

3. Reverse engineering

- A company disassembles a competitive hardware product in an effort to understand its competitor’s design and manufacturing “secrets.”

- Reverse engineering tools extract data, architectural, and procedural design information from an existing program.

Software Reengineering Process Model

4. Code restructuring.

5. Data restructuring

- A program with weak data architecture will be difficult to adapt and enhance.

- Current data architecture is dissected, and necessary data models are defined.

- Data objects and attributes are identified, and existing data structures are reviewed for quality.

Software Reengineering Process Model

6. Forward engineering

- Forward engineering not only recovers design information from existing software but uses this information to alter or reconstitute the existing system in an effort to **improve its overall quality.**

Reverse Engineering

- The process of recreating a design by analyzing a final product.
- The **abstraction** level of a reverse engineering **refers to the sophistication of the design information** that can be extracted from source code.
- The **completeness** of a reverse engineering process **refers to the level of detail that is provided at an abstraction level.**

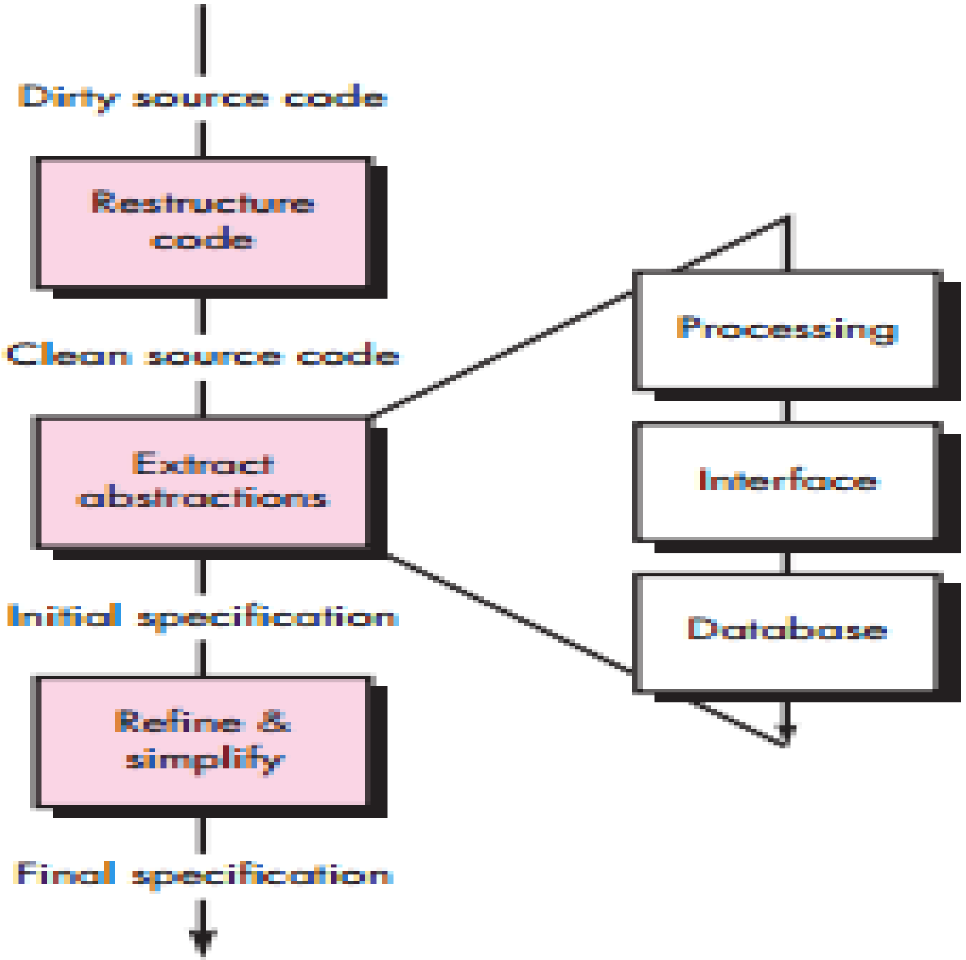
Reverse Engineering

- **Interactivity** refers to the **degree to which the human is “integrated” with automated tools** to create an effective reverse engineering process.

- Directionality – one way (maintenance activity) or two way (restructure)

Reverse Engineering

- Reverse Engineering Process



Software Maintenance

- Software maintenance is the general process of changing a system after it has been delivered.

- The change may be simple changes to correct coding errors, more extensive changes to correct design errors or significant enhancement to correct specification error or accommodate new requirements.

Software Maintenance

- **There are three different types of software maintenance:**

1. Fault repairs

- Coding errors are usually relatively cheap to correct.
- Design errors are more expensive as they may involve rewriting several program components.
- Requirements errors are the most expensive to repair because of the extensive system redesign which may be necessary.

Software Maintenance

2. Environmental adaptation

- This type of maintenance is required when some aspect of the system’s environment changes.

- Example: hardware, the platform operating system, or other support software changes.

- The application system must be modified to adapt it to cope with these environmental changes.

Software Maintenance

3. Functionality addition

- *This type of maintenance is necessary when the system* requirements change.

- The scale of the changes required to the software is often much greater than for the other types of maintenance.

Software Maintenance

- Other types of software maintenance with different names:

- **Corrective maintenance** is universally used to refer to maintenance for fault repair.

- **Adaptive maintenance** sometimes means adapting to a new environment

- **Perfective maintenance** sometimes means perfecting the software by implementing new requirements