Software Security Touchpoints

Security Engineering

- Reduce the need for reactive technologies (e.g., intrusion detection) by safer products
- Need for:
  - □ Software developers
  - □ Operations people
  - □ Administrators
  - □ Users
  - □ Executives

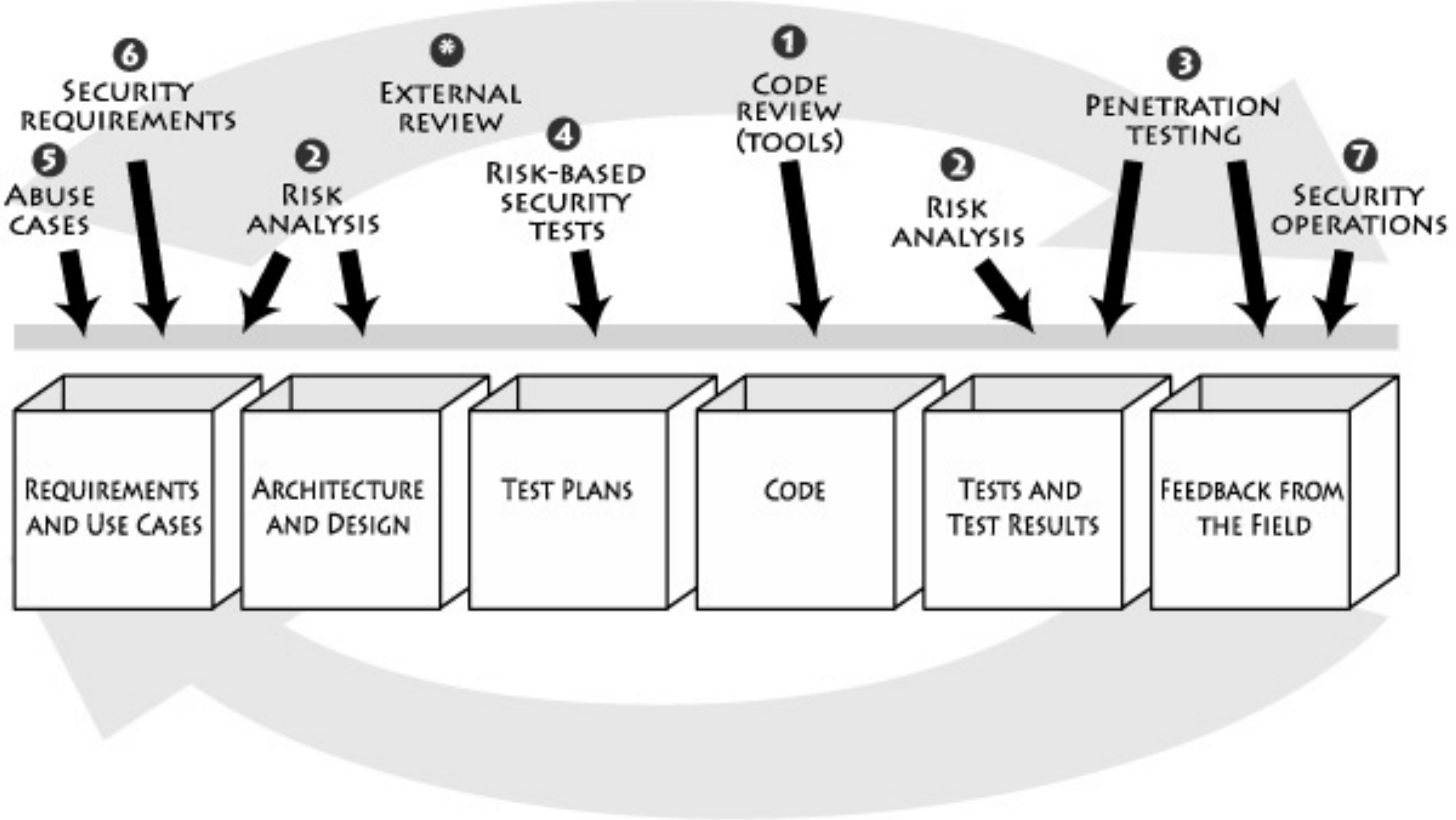
Why do these people need Security Engineering?

Software Security Touchpoints

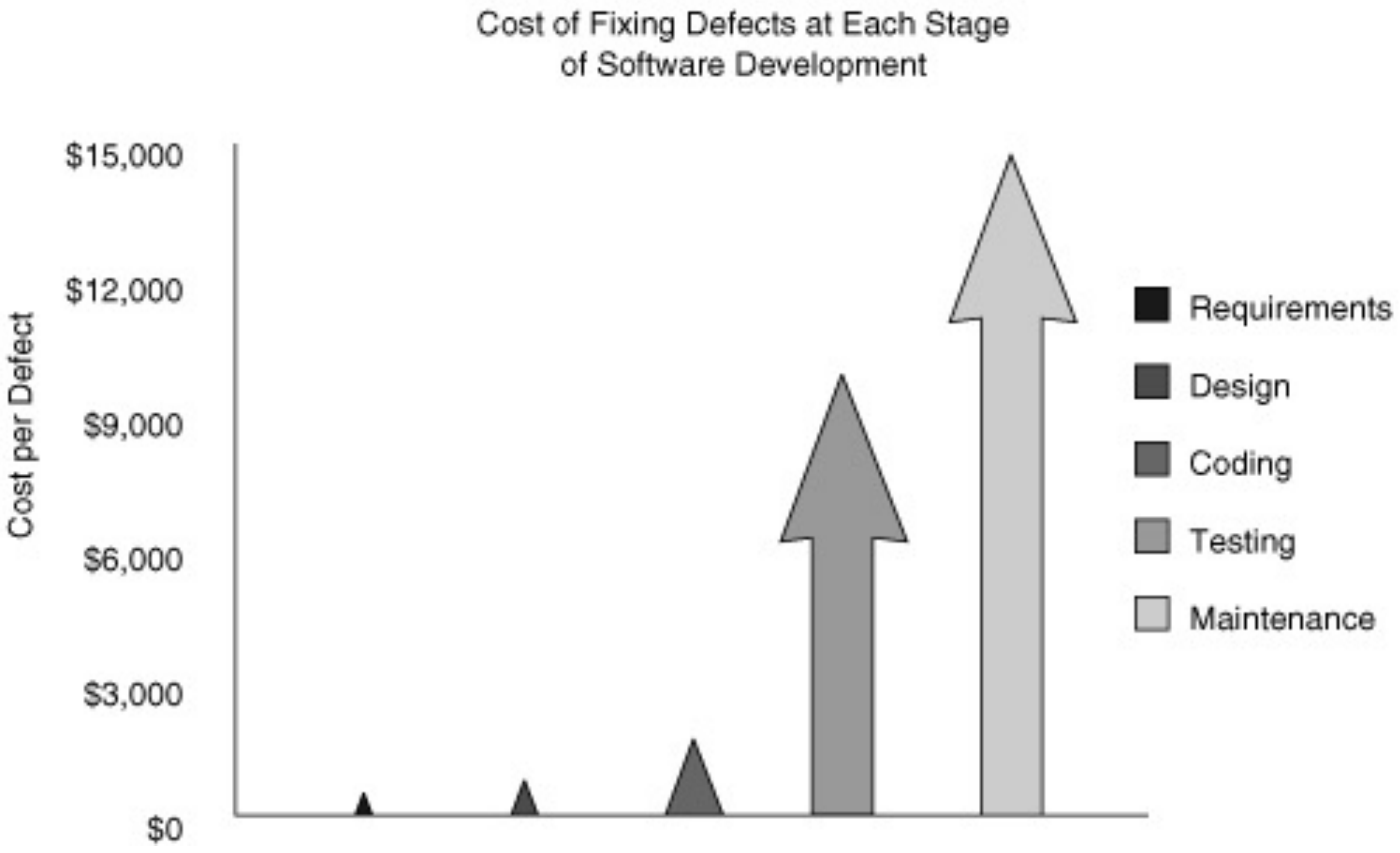
- Best Practices
- Both White Hat (constructive) and Black Hat (destructive) activities
- Throughout the SDLC

How do you think about security?
Which is more important? White hat or Black hat type of activities?

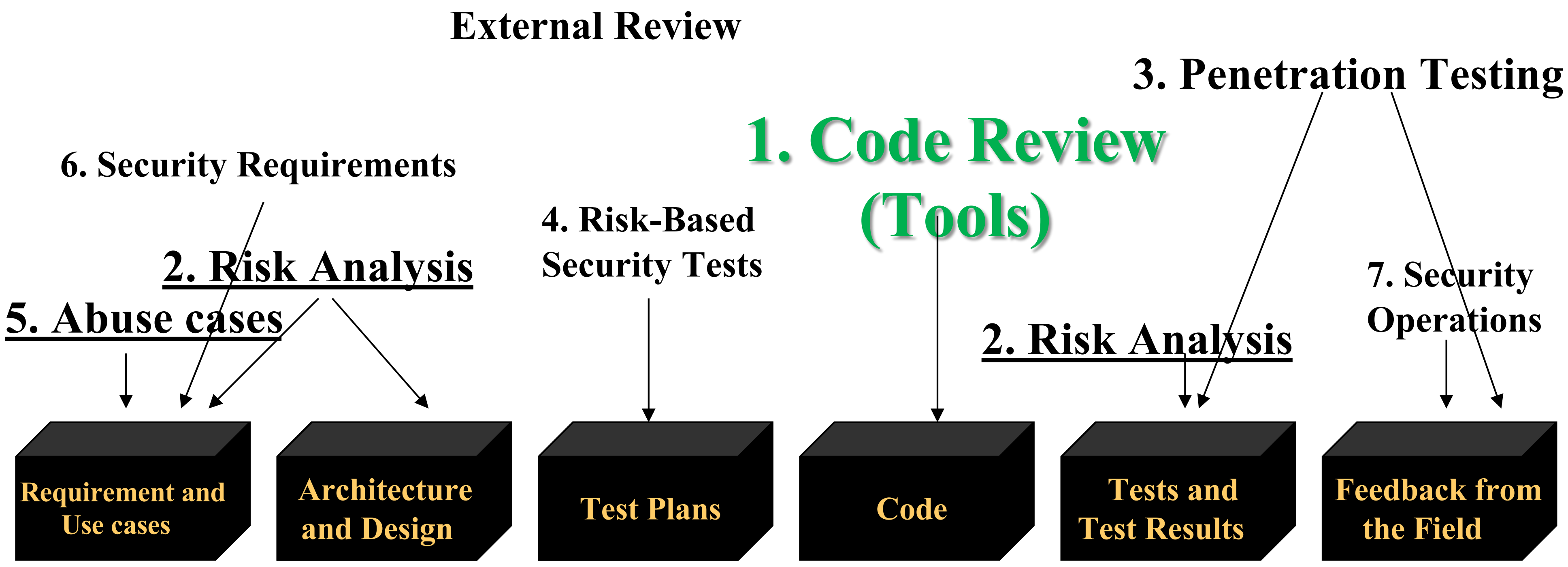
Application of Touchpoints



Cost of fixing defect at each stage



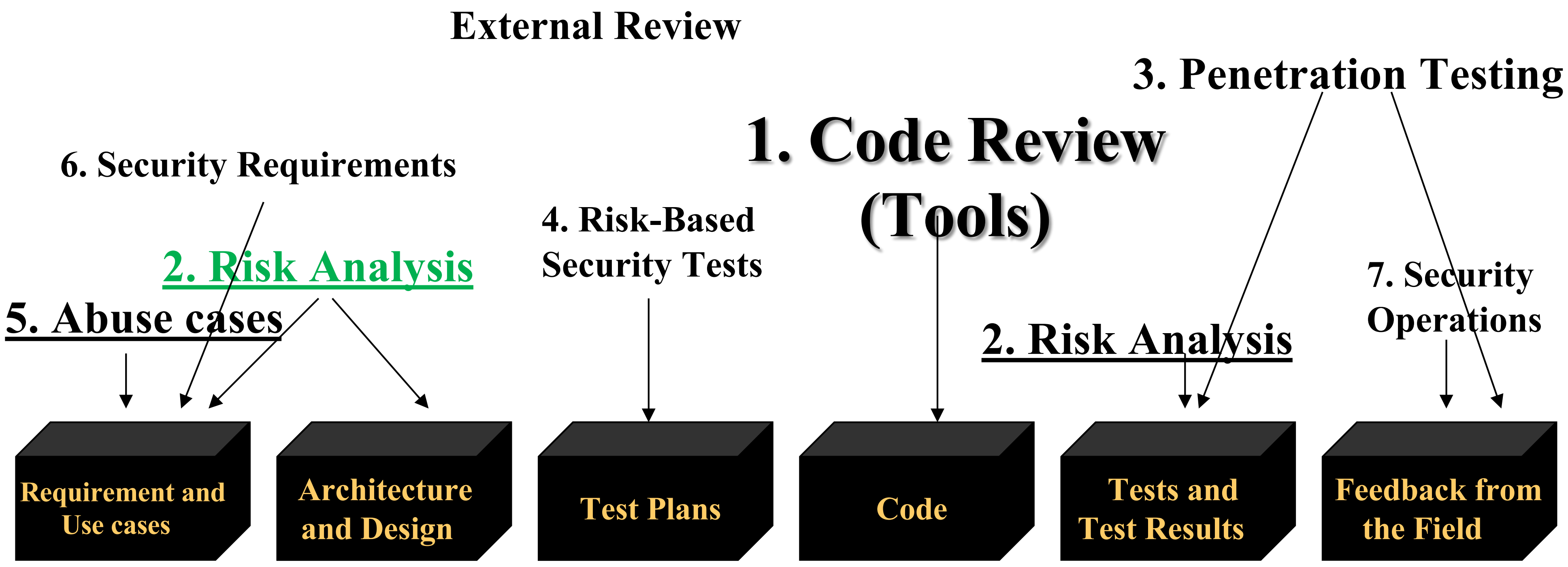
Application of Touchpoints



Code Review (Tool)

- Artifact: Code
- Implementation bugs
- Static Analysis tools
- White Hat

Application of Touchpoints



External Review

3. Penetration Testing

1. Code Review (Tools)

6. Security Requirements

4. Risk-Based Security Tests

2. Risk Analysis

5. Abuse cases

7. Security Operations

2. Risk Analysis

Requirement and Use cases

Architecture and Design

Test Plans

Code

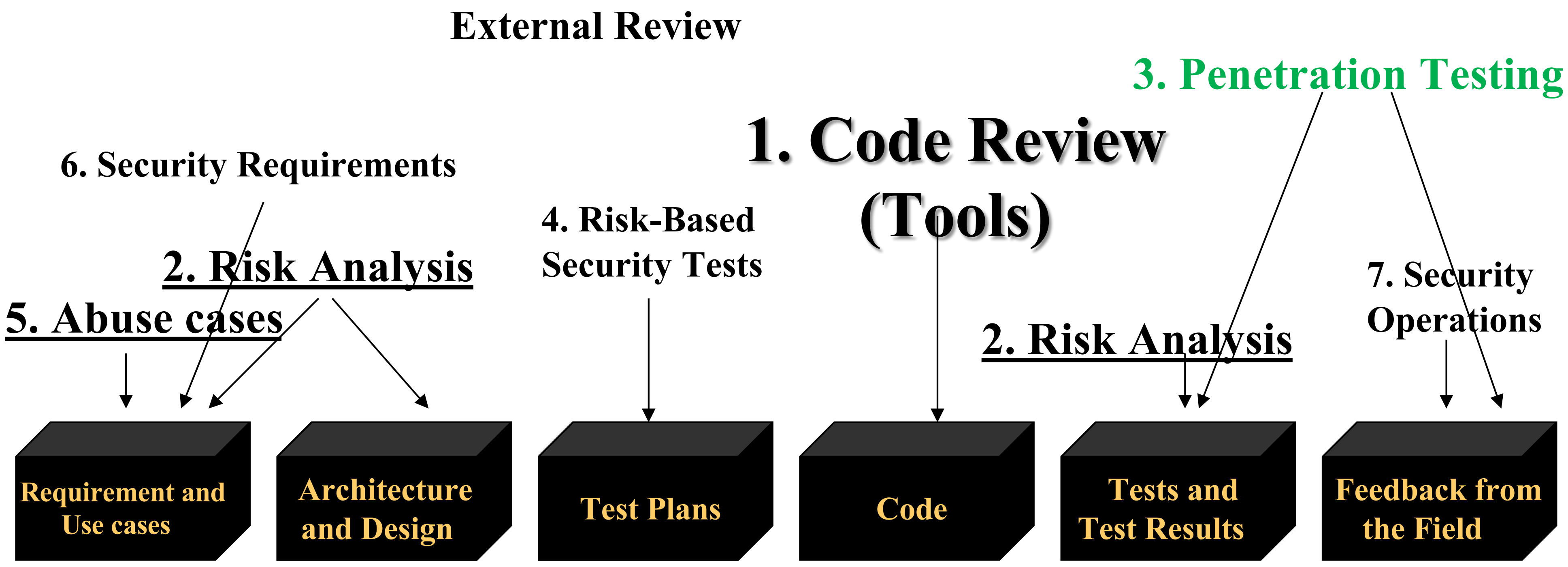
Tests and Test Results

Feedback from the Field

Architectural Risk Analysis

- Artifact: Design and specification
- System must be coherent and present a uniform security front
- Document assumptions and identify possible attacks
- Both at specification-based architecture stage and class-hierarchy design stage
- White hat

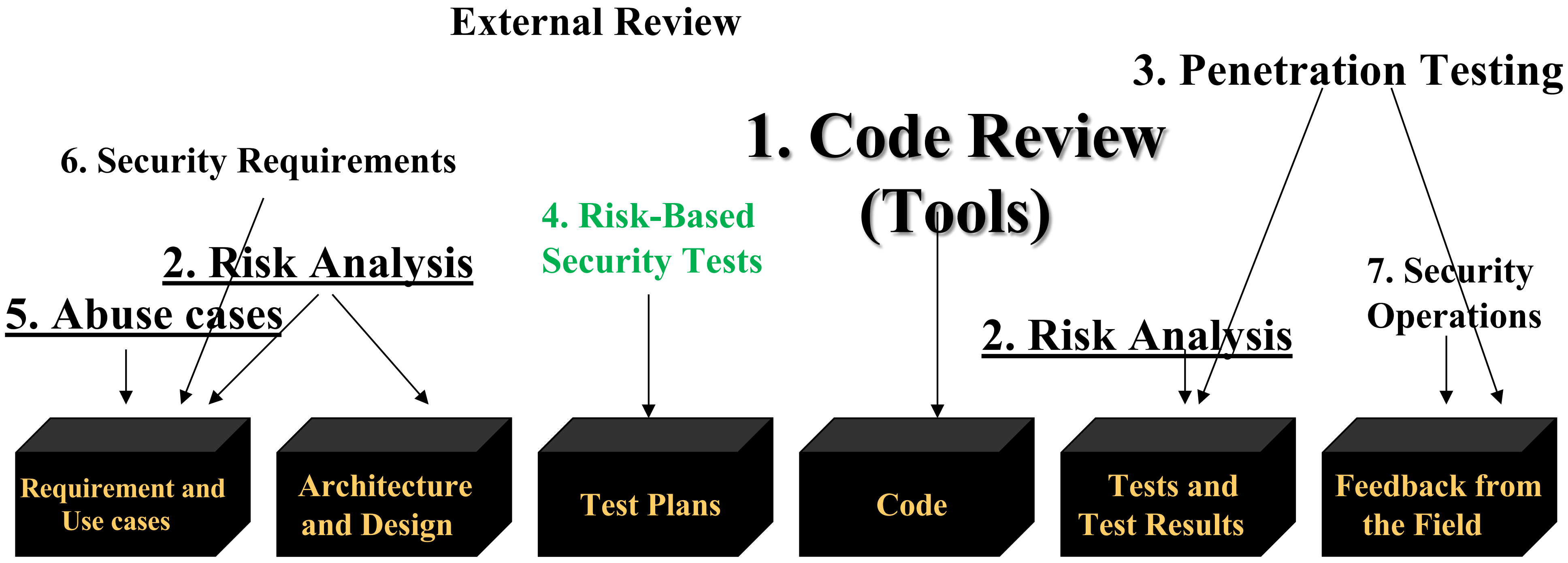
Application of Touchpoints



Penetration Testing

- Artifact: system in its environment
- Understanding fielded software in its environment
- Information supplied by architectural risk analysis
- Who does it?
- Black Hat

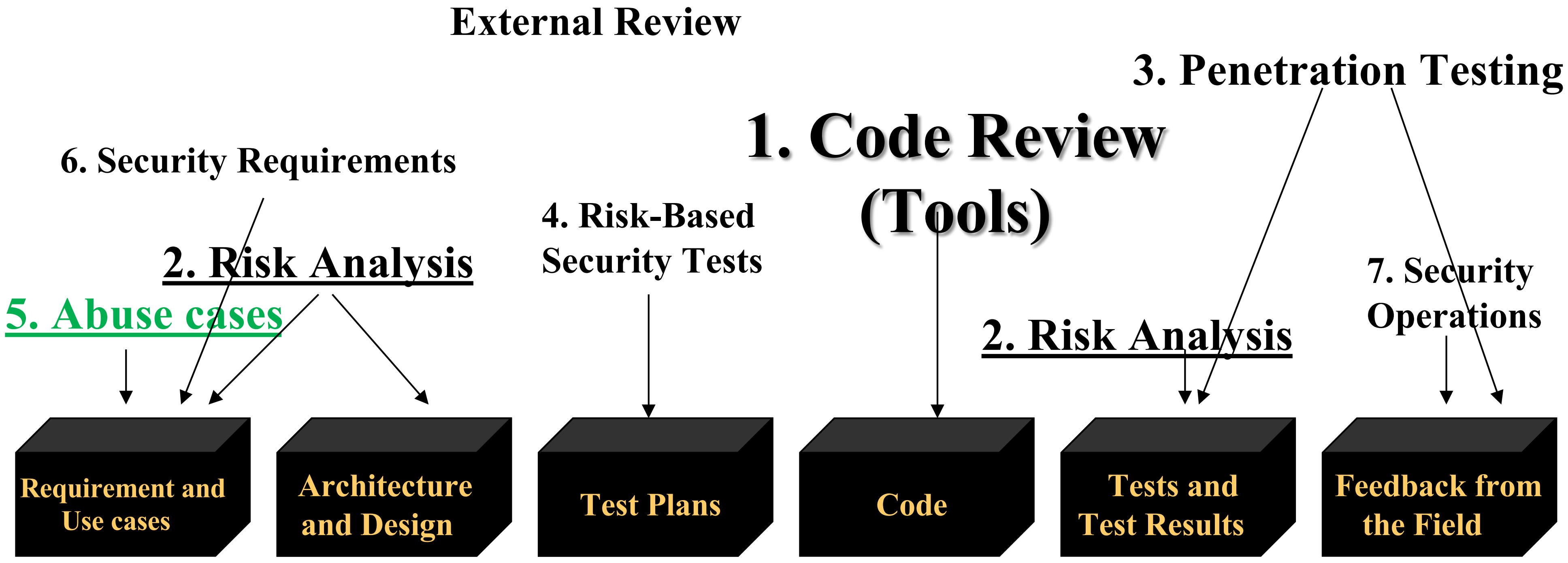
Application of Touchpoints



Risk-Based Security Testing

- Artifact: unit and system
- Strategies:
  - Testing of security functionality (standard functional testing)
  - Risk-based security testing (attack pattern, risk analysis, abuse cases)
- Attacker's mindset
- White Hat + Black Hat

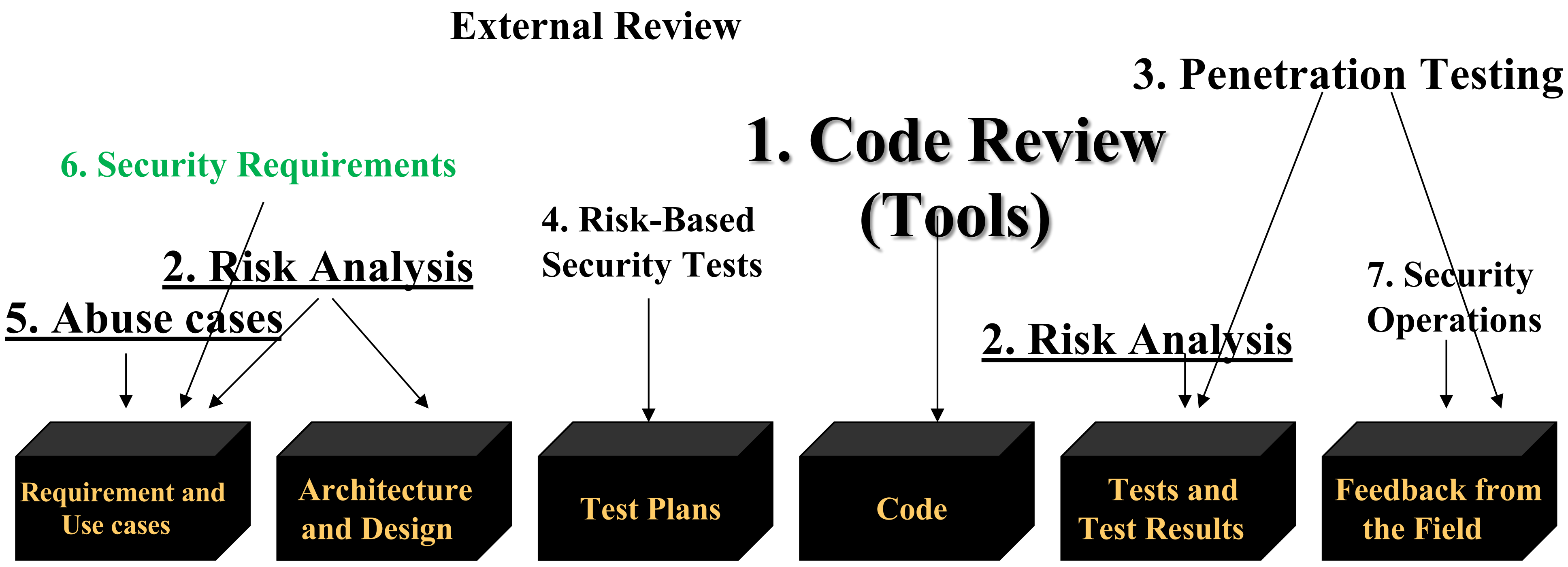
Application of Touchpoints



Abuse Cases

- Artifact: requirements and use cases
- Describe system behavior under attack
- Explicit coverage of
  - What should be protected
  - From whom
  - For how long
- White Hat + Black Hat

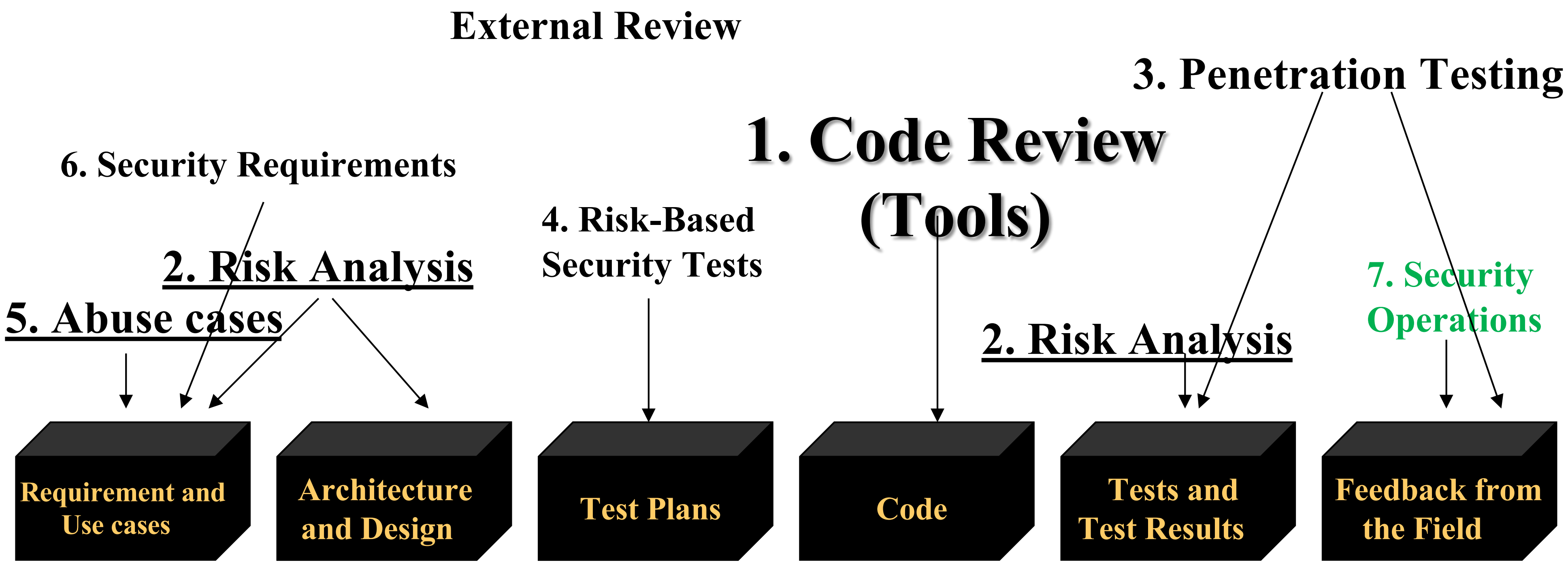
Application of Touchpoints



Security Requirements

- Artifact: Requirements
- Security explicitly worked into the requirements level
- Both functional security and emergent characteristics
- White Hat

Application of Touchpoints



External Review

3. Penetration Testing

1. Code Review (Tools)

6. Security Requirements

4. Risk-Based Security Tests

2. Risk Analysis

5. Abuse cases

2. Risk Analysis

7. Security Operations

Requirement and Use cases

Architecture and Design

Test Plans

Code

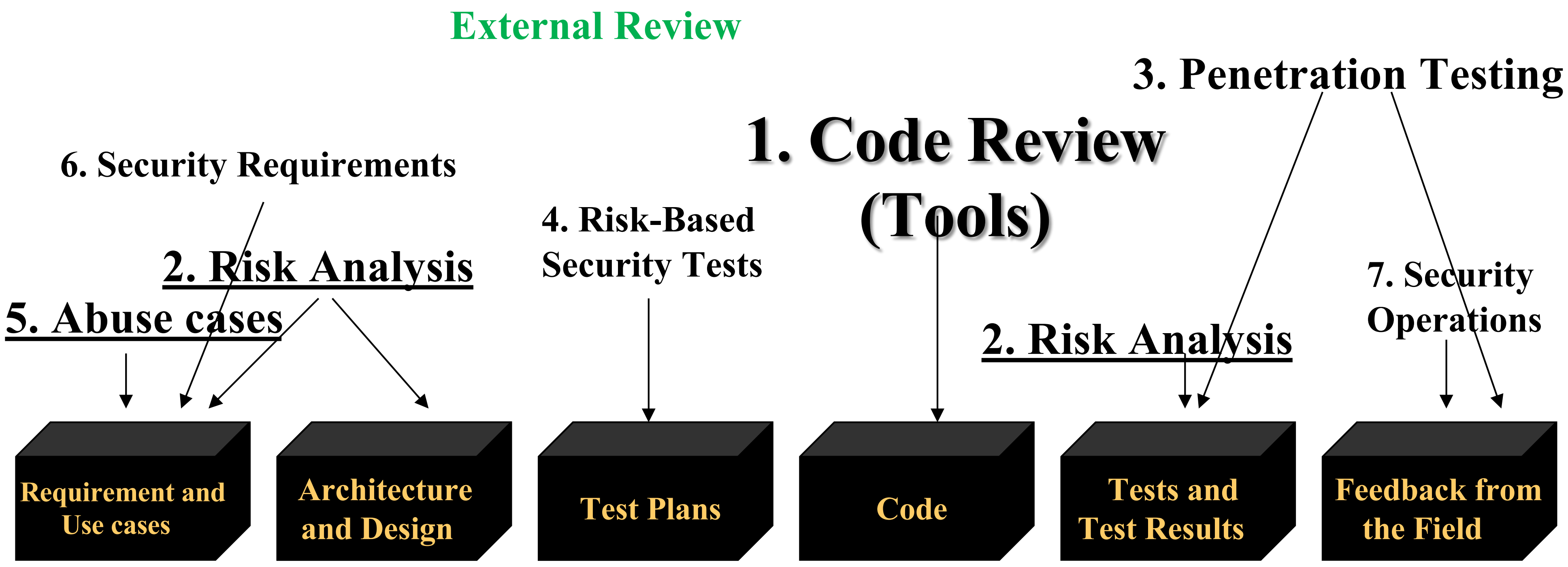
Tests and Test Results

Feedback from the Field

Security Operations

- Artifact: fielded system
- Monitoring system usage
- Combines both network centric and software specific operations
- White Hat

Application of Touchpoints



External Analysis

- Evaluate security by outside members
- Why?
- Advantages/disadvantages

When to Apply Security?

- Economical consideration: early is better
- Effectiveness of touchpoints:
  - Economics
  - Which software artifacts are available
  - Which tools are available
  - Cultural changes
- Bad: reactive strategy → need: secure development

Best Practices

- Earlier the better
- Change “operational” view to secure software
- Best practices: expounded/presented by experts and adopted by practitioners

Worst Practices to Avoid

From T. Demopoulos, “Worst Practices in Developing Secure Software

1. Assuming only “important’ SW needs to be secure
2. Emphasizing hitting deadlines over “good code”
3. Having IT make all risk management decisions
4. Not considering security during the entire SDLC
5. Assuming the SW won’t be attacked
6. Not doing any security testing
7. Not planning for failure
8. Counting on “security through obscurity”
9. Disallowing bad input instead of only allowing good input
10. SW that is not secure by default
11. Coding your own cryptography

Who Should Care?

- Developers
- Architects
- Other builders
- Operations people

Do not start with security people.
Start with software people.

Questions?