TOWARDS EFFICIENT SECURITY ASSURANCE FOR INCREMENTAL SOFTWARE DEVELOPMENT

THE CASE OF ZEN CART APPLICATION

Azmat Ali
TU Darmstadt
Germany

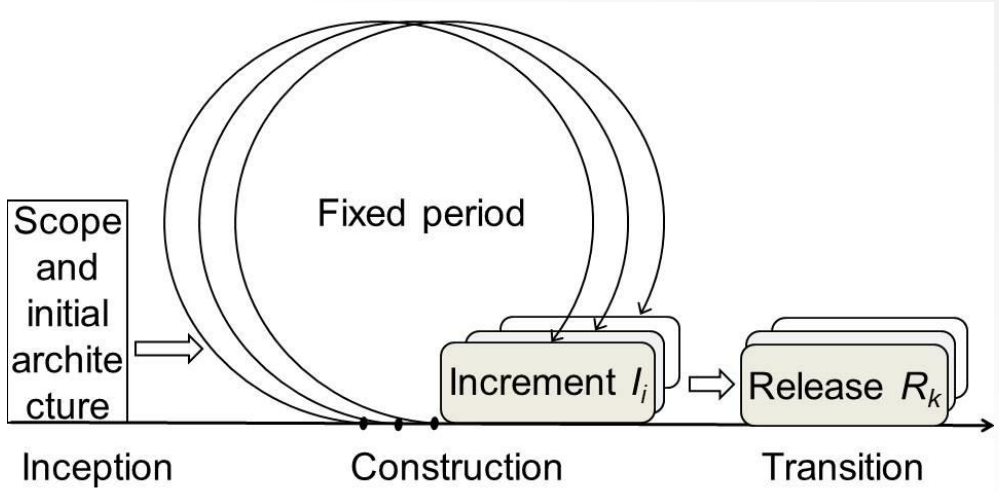
Lotfi Ben Othmane
Fraunhofer SIT
Germany

International Workshop on Agile Secure Software Development

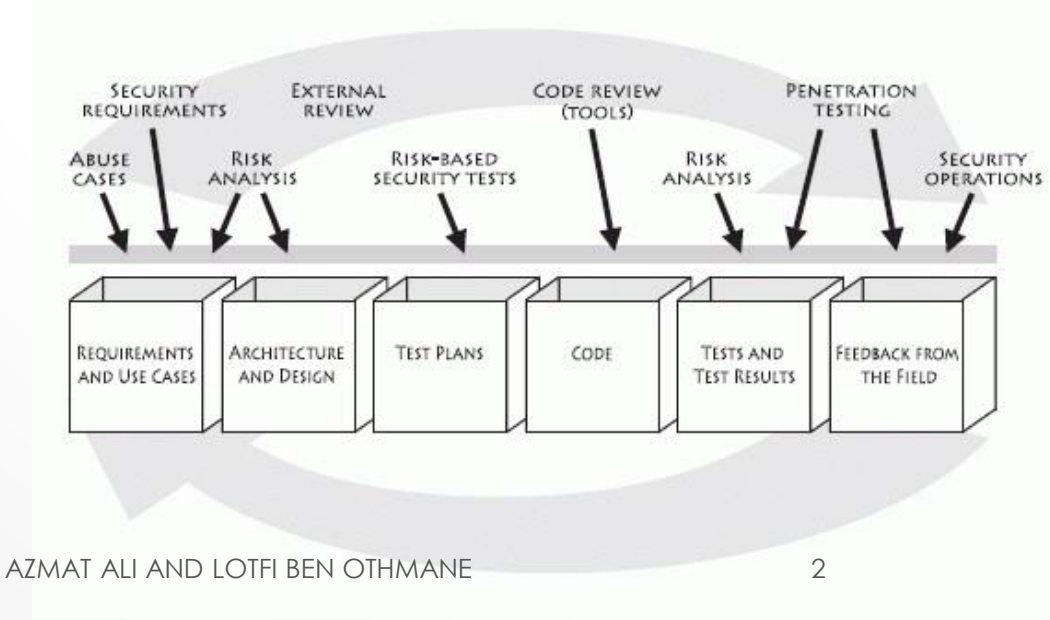
Salzburg, Austria September 1, 2016

MOTIVATION

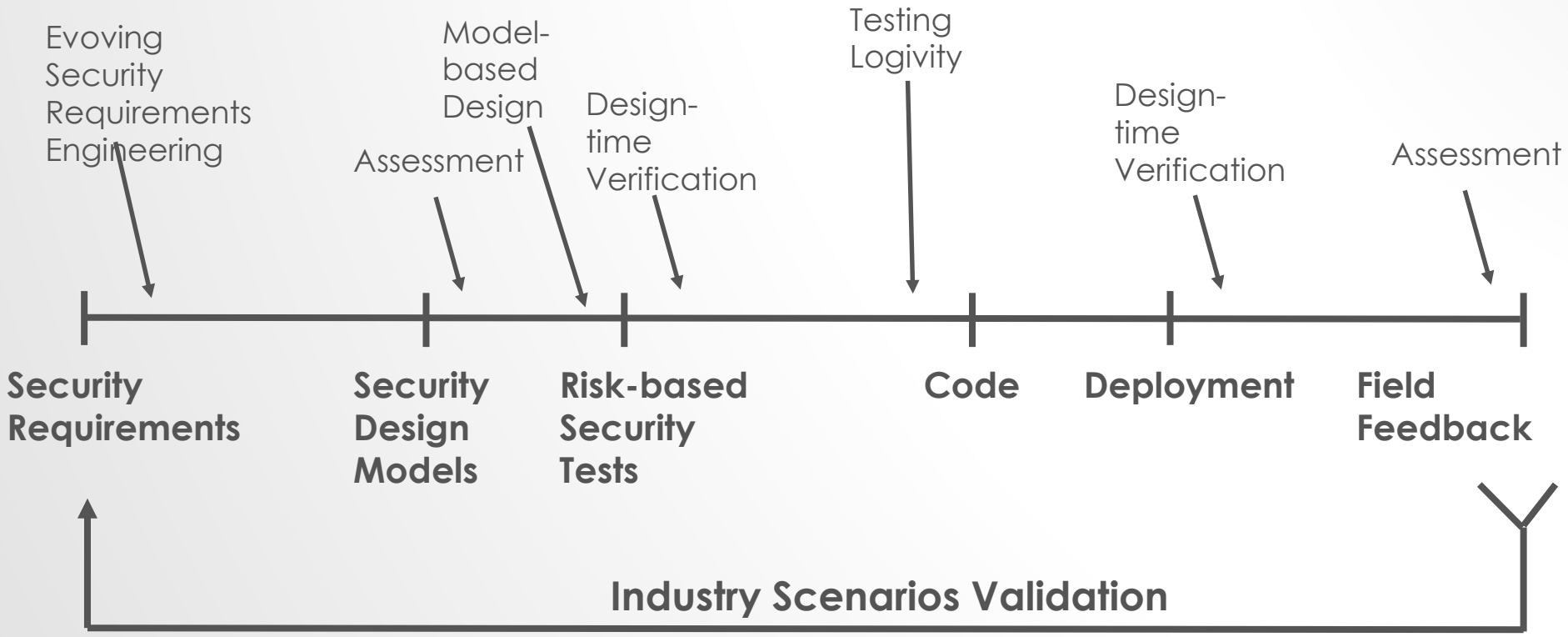
- Software development is iterative and incremental



- Code changes invalidates software security assurance
  - Security mechanisms become invalid
  - Security evaluations become invalid or obsolete



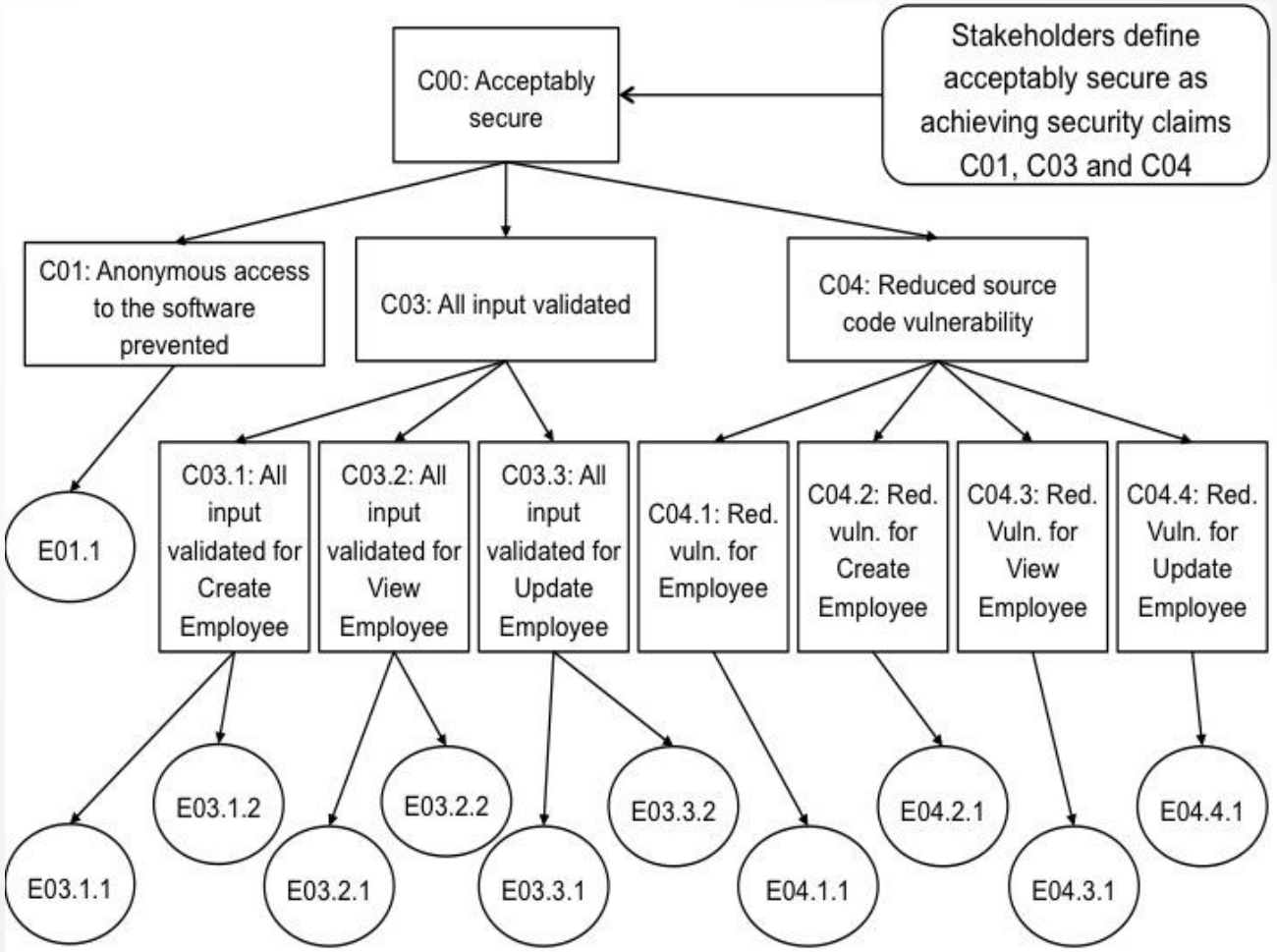
MODEL-BASED SOLUTION– SECURE CHANGE



http://www.securechange.eu/

USE OF SECURITY ASSURANCE CASED

- Example of assurance case for employees management – iteration1

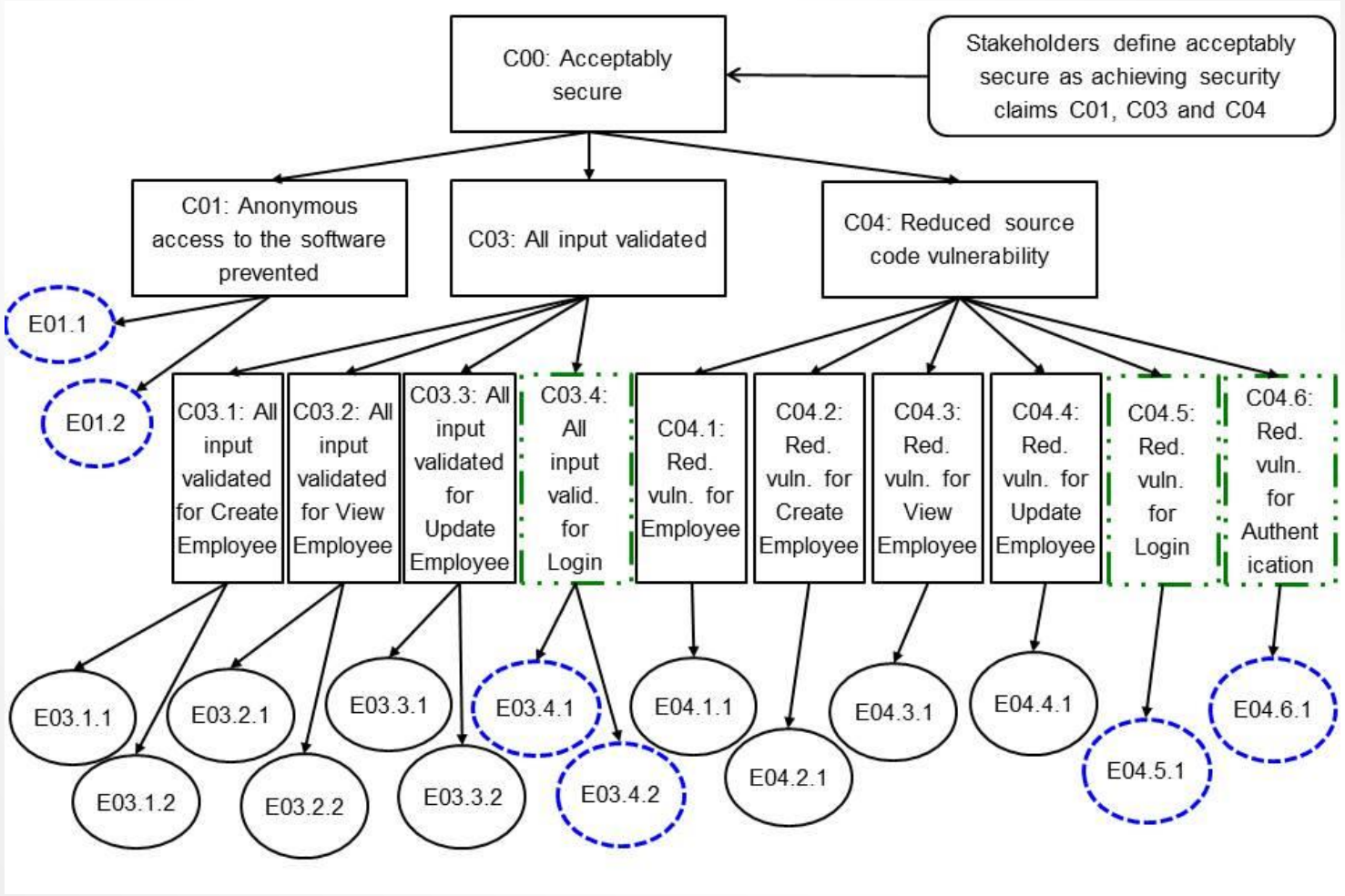


- Rectangles are for claims
- Circles are for evidence

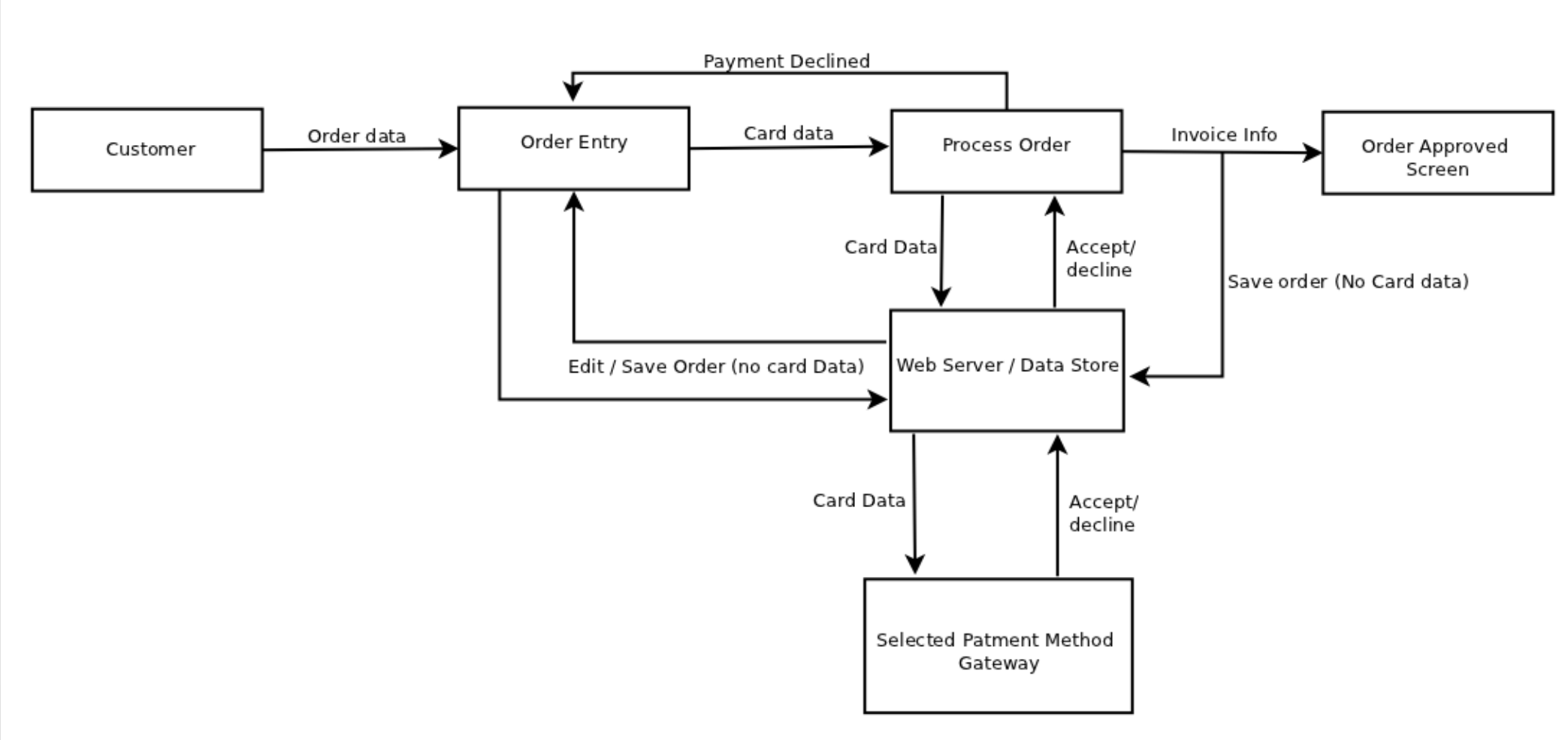
USE OF SECURITY ASSURANCE CASED

- Use of Login Web page instead of OS login

- The shapes representing changed artifacts have dashed lines



ZEN CART – COLLABORATION DIAGRAM??



SECURE CODE CHANGE AT ZEN CART

- No use of architecture diagrams => they become obsolete rapidly
- Use OWASP 10 threats and PA-DSS requirements

- Results of the interview with the security lead (March 2016)
  1. Changes to security requirement of the payment card industry (PA-DSS) implies a complete reassessment
  2. Code changes may impact security mechanisms
  3. Framework changes may require redesign of security mechanisms
  4. One of each 50 comment-changes are related to security

SECURE CODE CHANGE AT ZEN CART

- Lessons learned

1. Test security requirements are misleading

2. Architecture diagrams have a short life time

3. Security solutions shall be at from architecture level
   - You need either few sanitization APIs or inefficient fragmented and repeated code

4. New techniques could identify vulnerabilities in old code
   - E.g., dynamic code analysis for XSS

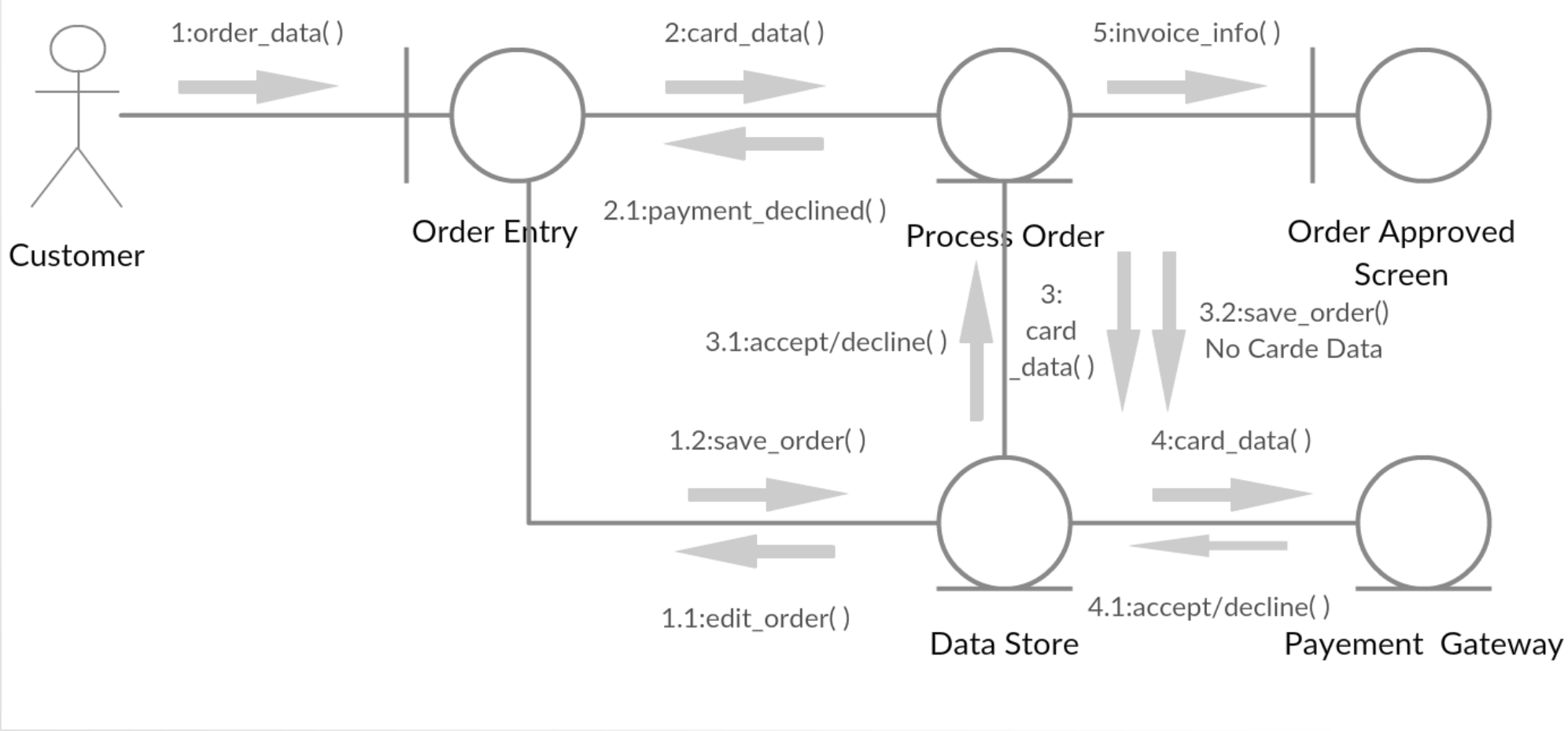
THREAT MODELING

- Failed attempts for architecture extraction
  - Used PHP Pear module and AgroUML
  - Class diagram was very big and no obvious patterns

- Failed automated extraction of attack surface
  - Used PHPCallGraph, RIPS, and doxygen
  - Call grah very big due to number of functions

=> Developed a conceptual DFD

THREAT MODELING – DATA FLOW DIAGRAM



THREAT MODELING – DATA FLOW DIAGRAM

- Applied STRIDE method to extract threats
  - Used Microsoft threat modeling tool

- Main threats
  - Spoofing of destination (07)
  - Elevation using impersonation (04)
  - Potential excessive resource consumption (04)
  - Weak access control (03)
  - Memory tampering of processes (03)

  => Conceptual threats

ANALYSIS OF KNOWN VULNERABILITIES

- 35 vulnerabilities in NVD database (Dec. 2004 to Apr. 2015)
  - 17 design-based flaws (about 50%)
  - 17 code-based flows (about 50%)
  - 1 unknown

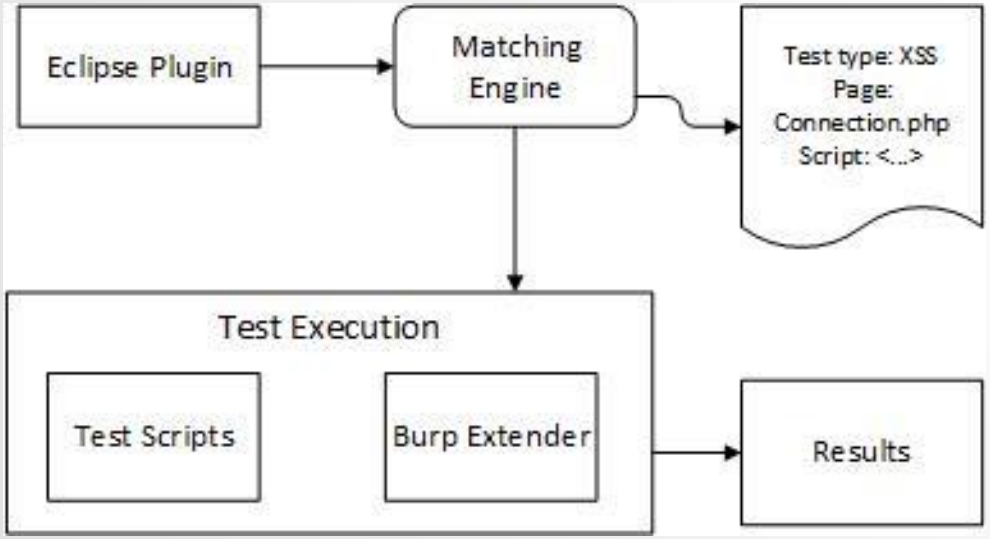
- Main vulnerabilities
  - SQL injection (7) XSS (7) Data validation (4)
  - Authentication issue (3) information leakage (3) path traversal (2 from 3)

- No. of Vulnerabilities decreases over time => product becomes mature ?

  => Mature with respect to known vul. And known discovery techniques

PENETRATION TESTING

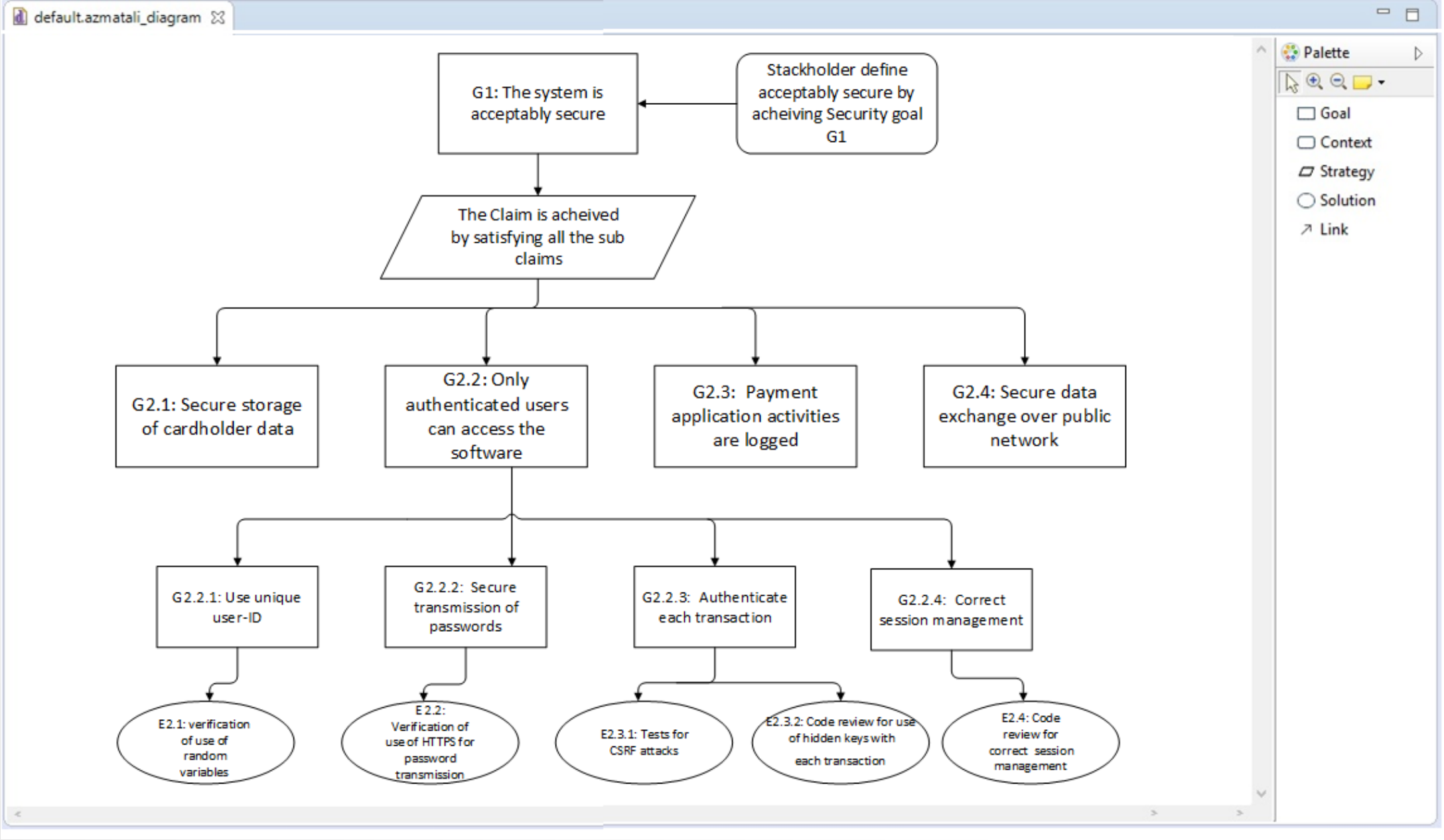
- Performed 25 tests cases XSS and SQL injection for version 1.5.4
  - No issues
- Automated the process using Selenium and JSoup



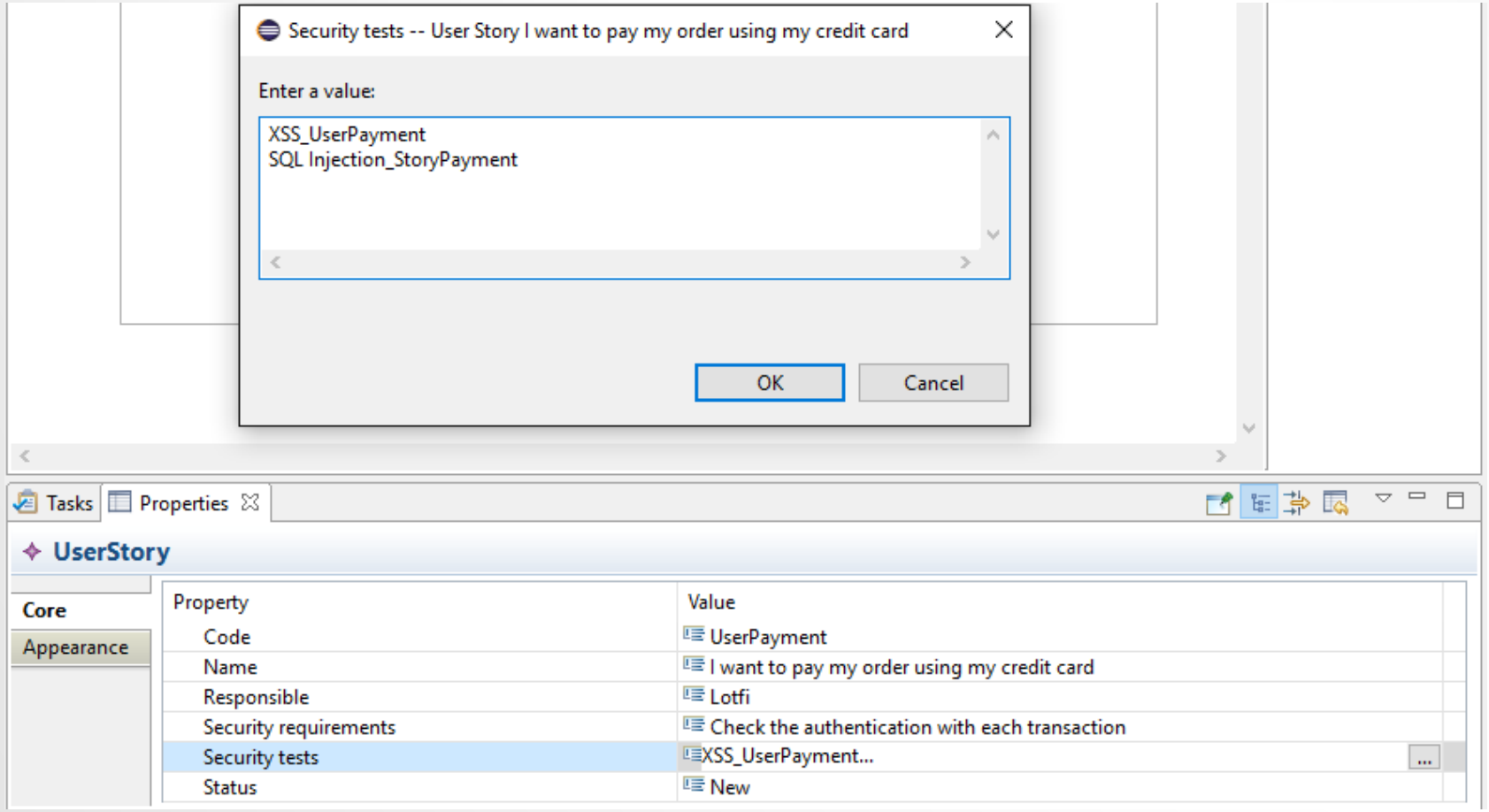

```
<test>
        <TestType>Cross Site Scripting</TestType>
        <Page>index.php, demoadmin/index.php</Page>
        <Script>&lt;body onload=alert("XSS")&gt;&lt;/body&gt;<Script>
</test>

<test>
        <TestType>SQL Injection</TestType>
        <Page>demoadmin/login.php</Page>
        <Script>username = admin Password = anything OR 'x' = 'x'</Script>
</test>
```

VISUAL DESIGN OF SECURITY ASSURANCE CASES



MAPPING OF USER STORIES TO ASSURANCE CASES



SUMMARY

- Three changes impact the security assurance of software
  - Requirement changes
  - Code changes
  - Security mechanism changes
- Zen card had 35 Vulnerably in the NVD
  - 17 are code-level
  - 17 are design-level
- User stories could be mapped to security assurance artifacts
- More empirical research is needed to get insights

Thank you

Lotfi ben Othmane

lotfiben.othmane@cased.de