THE UNIVERSITY OF BRITISH COLUMBIA

# eXtreme Security Engineering

KONSTANTIN BEZNOSOV

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

http://www.ece.ubc.ca/~beznosov/


# What's security engineering?

Development/integration of "security-intensive" solutions

Examples

- Smart card device
- Enterprise security


# What's "good enough security"?

- "good enough" risk analysis
- "good enough" requirements engineering
- "good enough" modeling
- "good enough" design, development, composition
- "good enough" testing
- "good enough" assurance

Customer is comfortable with

# Premises and beliefs

- Customers cannot afford and/or don't want "absolute security"
  - "good enough security" cannot be defined well enough *a priory*
  - No "good enough common criteria"!
- Uncertainty and change are inevitable in security engineering projects
  - "how to manage", not "how to limit"

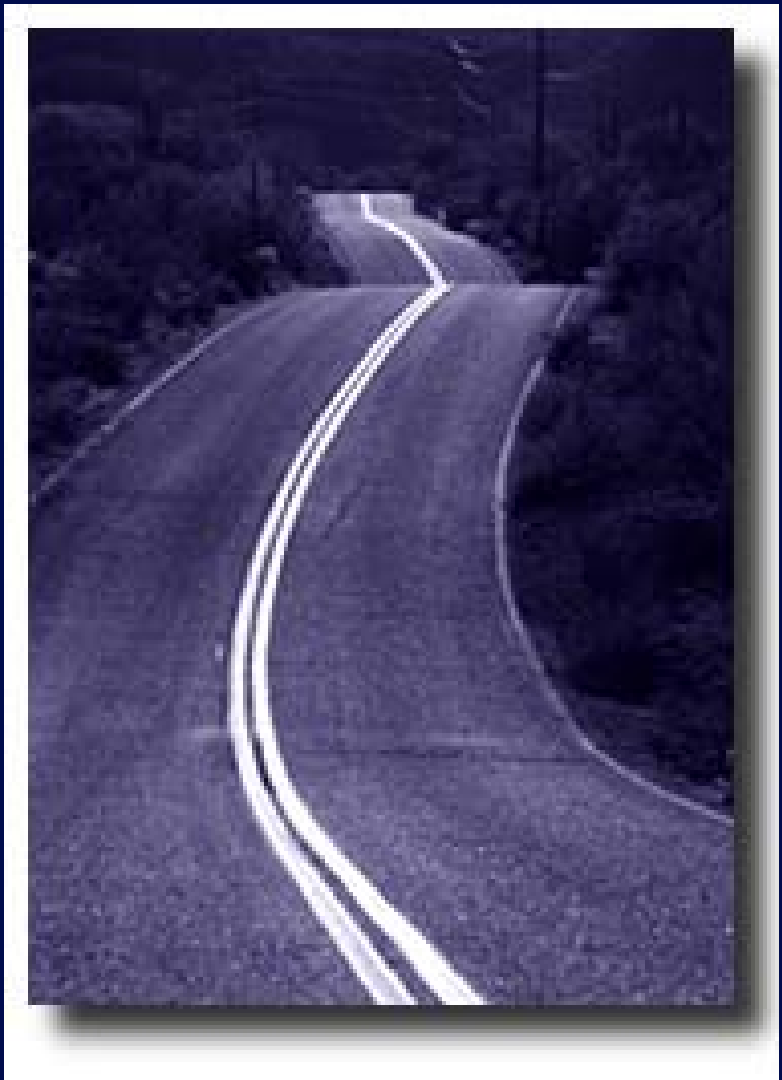
# Key points

- "good enough security"
  - don't define *a priory*
  - define as you go
  - let the customer(s) "define" and change it
    - ***How?***
- eXtreme Security Engineering (XSE)
  - adoption of XP
  - benefits
    - project success rate
    - customer satisfaction
    - "define" and adjust "good enough security" just-in-time
- position (i.e., speculation)


# Reasoning

- Why XP?
  - ASD/XP
    - "good enough" software
    - short feedback loop
  - software eng. ≈ security eng.
  - iterative and incremental development (IID) in non-software manufacturing
- XSE applicability
  - scope
  - anticipated dificulties

# What's XP?



small releases

planning game

user stories

metaphor

simple design

tests

refactoring

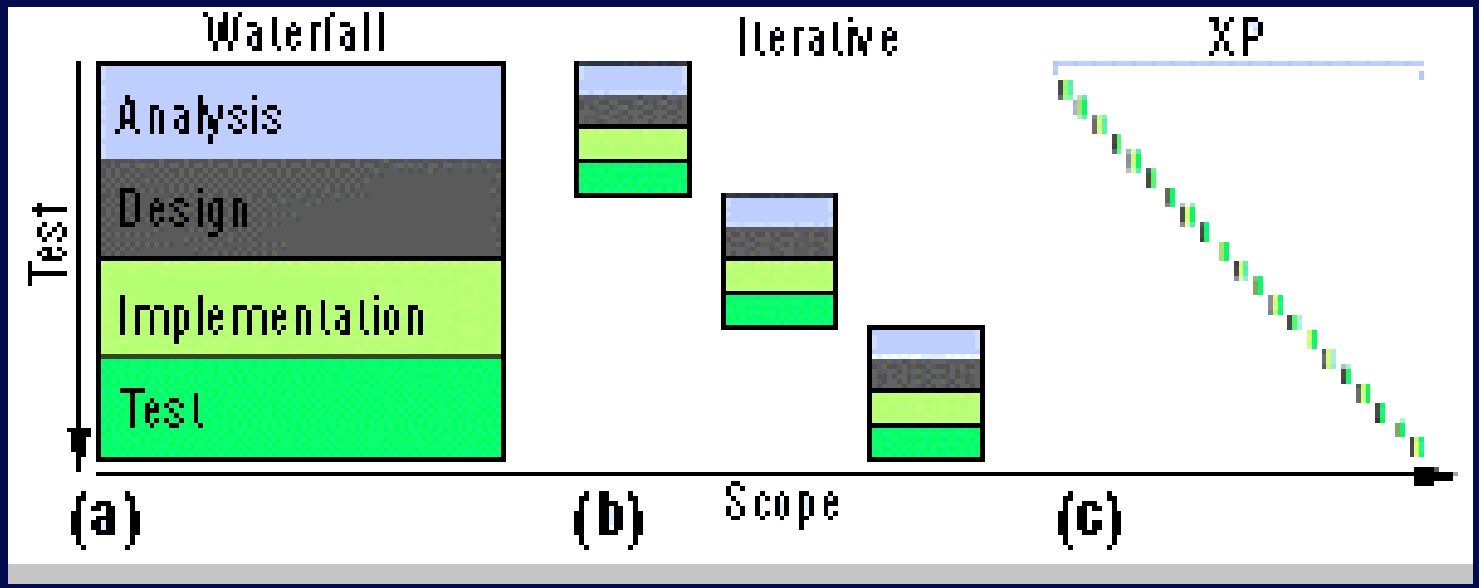
pair programming

continuous integration

collective ownership

onsite customer

# How short is feedback loop in XP?



# Why software eng. ≈ security eng.?

1. A system's users seldom know exactly what they want and cannot articulate all they know.
2. Even if we could state all requirements, there are many details that we can only discover once we are well into implementation.
3. Even if we knew all these details, as humans, we can master only so much complexity.
4. Even if we could master all this complexity, external forces lead to changes in requirements, some of which may invalidate earlier decisions.

Parnas and Clements, "A Rational Design Process: How and Why to Fake It"


# What's XSE applicability scope?

- nonsafety-critical projects
  - volatile requirements
  - development teams
    - small
    - skilled
    - collocated
- Boehm and Turner
  - size, criticality, dynamism, personnel, culture
  - incorporate agile and plan-driven approaches


# Anticipated difficulties

- analysis and testing
- refactoring
  - COTS and hardware
  - "distributed undo"
- "no map"


# Conclusions

- adoption of XP to security eng.
- "embrace" changes
  - extremely short feedback loop
  - higher success rate
  - better customer satisfaction
- customers drive "good enough security"
- benefits and applicability extrapolated from XP
- expected difficulties
- idea/speculation