Material and some slide content from:
- Software Architecture: Foundations, Theory, and Practice
- Krzysztof Czarnecki

Security as a Architectural Concern, Chrome Arch, and NFP Measurement

Reid Holmes

NFP: Security

- Security: “The protection afforded a system to preserve its integrity, availability, and confidentiality if its resources.”
- Confidentiality
  - Preserving the **confidentiality** of information means preventing unauthorized parties from accessing the information or perhaps even being aware of the existence of the information.
- Integrity
  - Maintaining the **integrity** of information means that only authorized parties can manipulate the information and do so only in authorized ways.
- Availability
  - Resources are **available** if they are accessible by authorized parties on all appropriate occasions.

[TAILOR ET AL.]

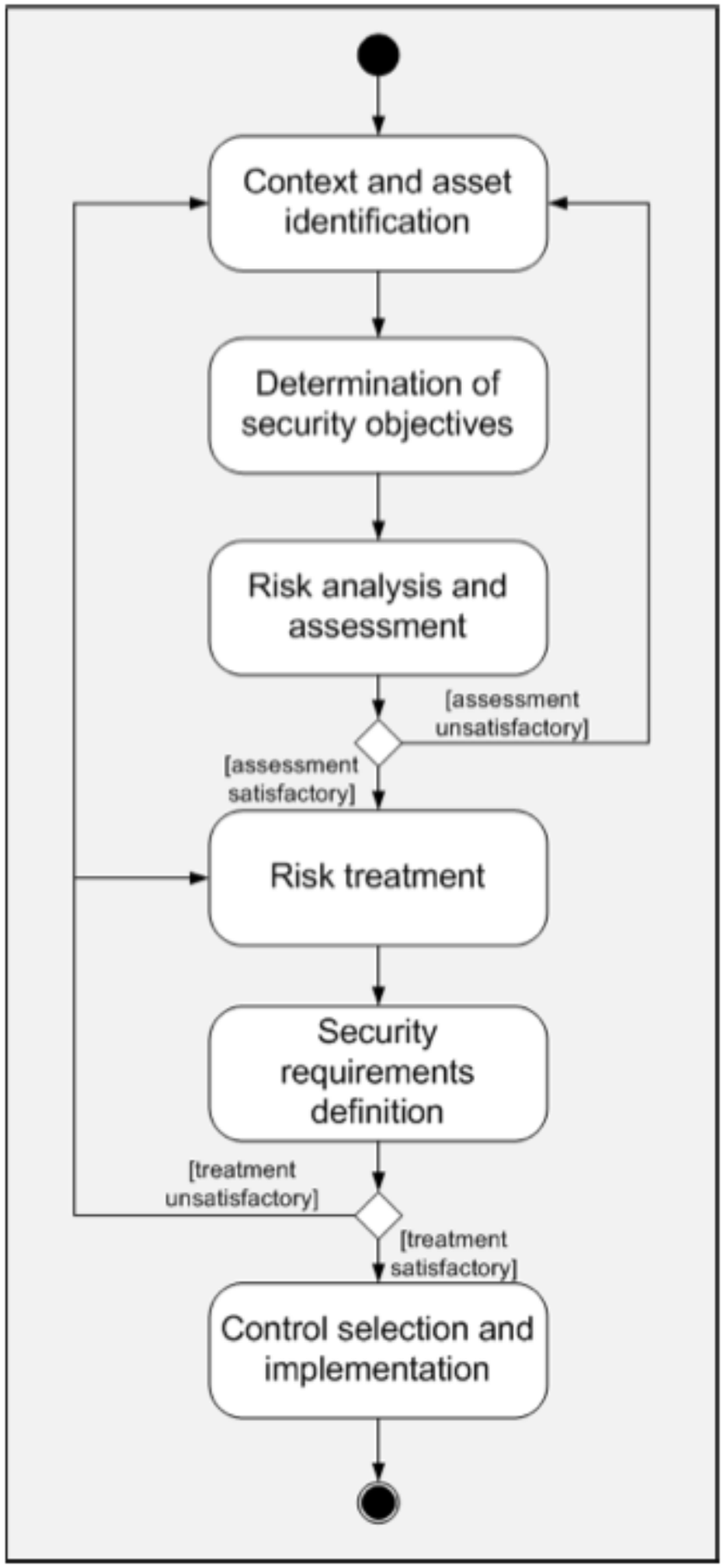
Security arch. principles

- Least privilege:
  - Give each component only the privileges it requires.
- Fail-safe defaults
  - Deny access if explicit permission is absent.
- Economy of mechanism
  - Adopt simple security mechanisms.
- Open design
  - Secrecy != security.

[TAILOR ET AL.]

Security arch. principles

- Separation of privilege
  - Introduce multiple parties to avoid exploitation of privileges.
- Least common mechanism
  - Limit critical resource sharing to only a few mechanisms.
- Psychological acceptability
  - Make security mechanisms usable.
- Defence in depth
  - Have multiple layers of countermeasures.



Security Process

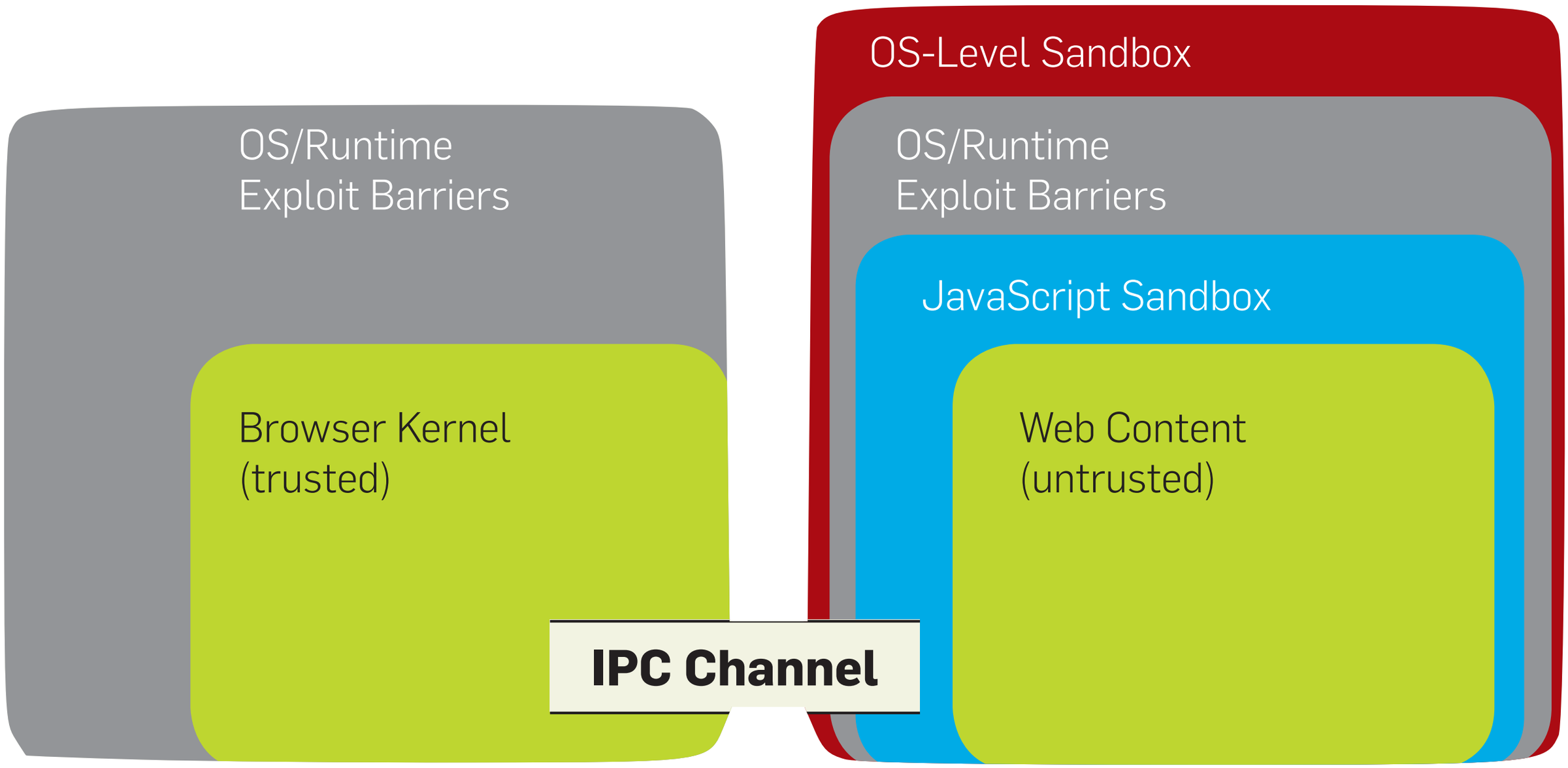
- Damage potential: how bad would the damage be?
- Reproducibility: how easy it is to repeat?
- Exploitability: how easy it is to do?
- Affected users: how many users can be affected?
- Discoverability: how easy it is to discover potential threats?

Chrome

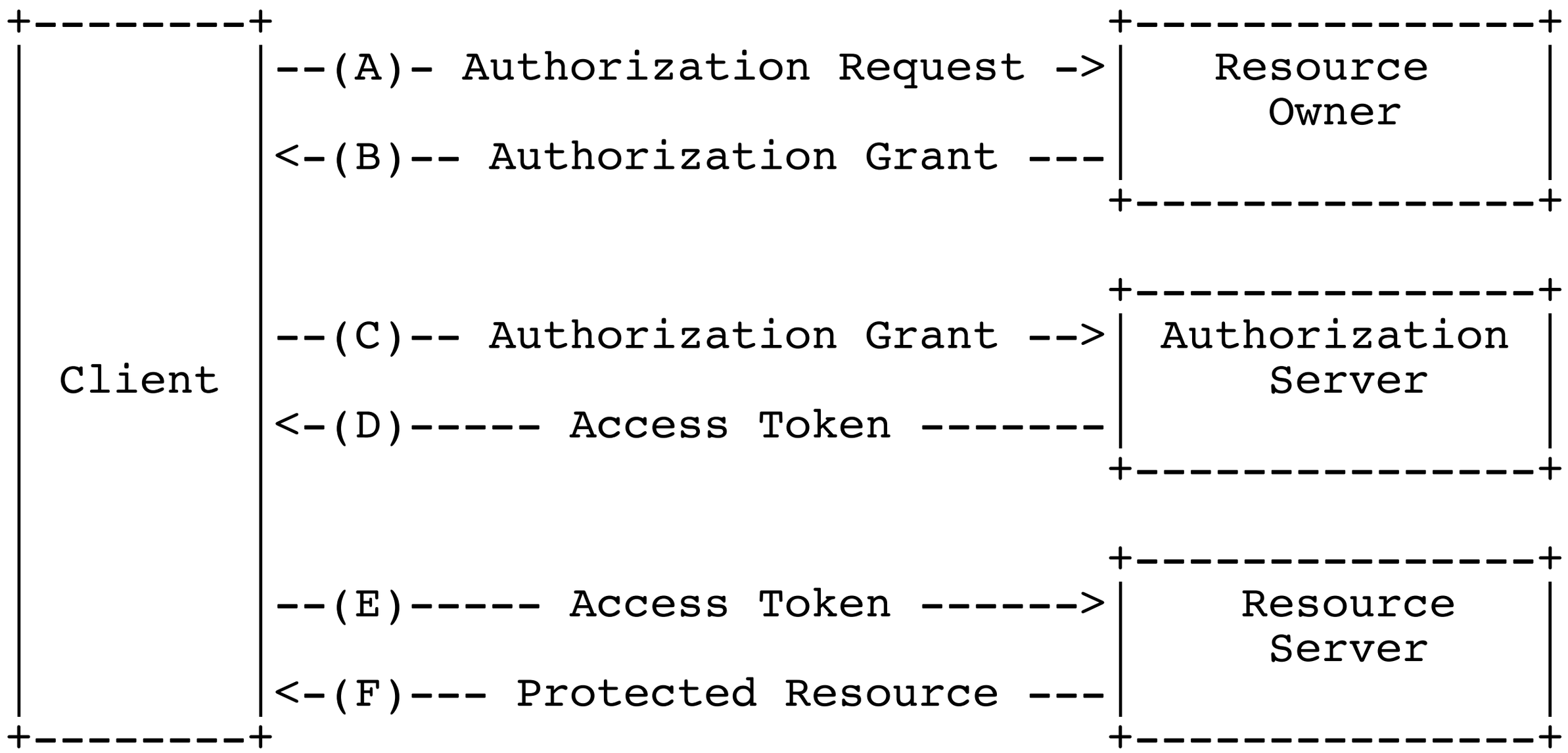
- Online content is insecure and can compromise:
  - Confidentiality: Leak user data
  - Integrity: Read/write arbitrary data on disk
  - Availability: Crash host application and/or OS

Chrome relies on **least privilege**, **separation of privilege**, and **defence in depth** to securely parse and render insecure content.

Chrome architecture



OAuth 2.0


Client
Resource Owner
Authorization Server
Resource Server
(A)- Authorization Request
(B)- Authorization Grant
(C)- Authorization Grant
(D)- Access Token
(E)- Access Token
(F)- Protected Resource

Evaluating NFPs

- It is tempting to treat NFPs abstractly
- Thinking about NFPs concretely means thinking about how they might be measured
- If you do not do this, it is hard to validate whether a design / arch decision supports or inhibits an NFP
- e.g.,
  - Reliability:
    -
    -
    -
    -

Evaluating NFPs

- It is tempting to treat NFPs abstractly
- Thinking about NFPs concretely means thinking about how they might be measured
- If you do not do this, it is hard to validate whether a design / arch decision supports or inhibits an NFP
- e.g.,
  - Reliability:
    - X bugs / KLOC
    - Coverage
    - Mutation Kill Score
    - MTBF

Evaluating NFPs

- Robustness
  -
  -
- Performance
  -
  -
  -
- Usability
  -
  -
  -
  -
  -
- Portability
  -
  -
  -

Evaluating NFPs

- Robustness
  - Time to restart after failure
  - % of transactions that will cause failure
- Performance
  - Transactions / second
  - Response time (latency)
  - max cpu usage (so background tasks can run)
- Usability
  - Training time (ease of learning)
  - Walkthroughs / user scenarios (task efficiency)
  - Ease of remembering
  - Objective satisfaction
  - Understandability (opaqueness)
- Portability
  - # target systems
  - % code that is platform-specific
  - # of platform-specific checks in normal execution

Evaluating NFPs

- Complexity
  -
  -
  -
- Security
  - Availability
    -
    -
  - Integrity & Confidentiality
    -
- Scalability
  -
  -
  -

Evaluating NFPs

- Complexity
  - Max LOC / function
  - Cyclomatic complexity
  - Maximum order of modules in dependency graph
- Security
  - Availability
    - % of time system is available to legitimate users
    - Longest duration of compromise
  - Integrity & Confidentiality
    - Penetration / fuzz testing / intentional misuse
- Scalability
  - Connections per second
  - Max # of simultaneous connections
  - Max delay to spin up new app server if saturated

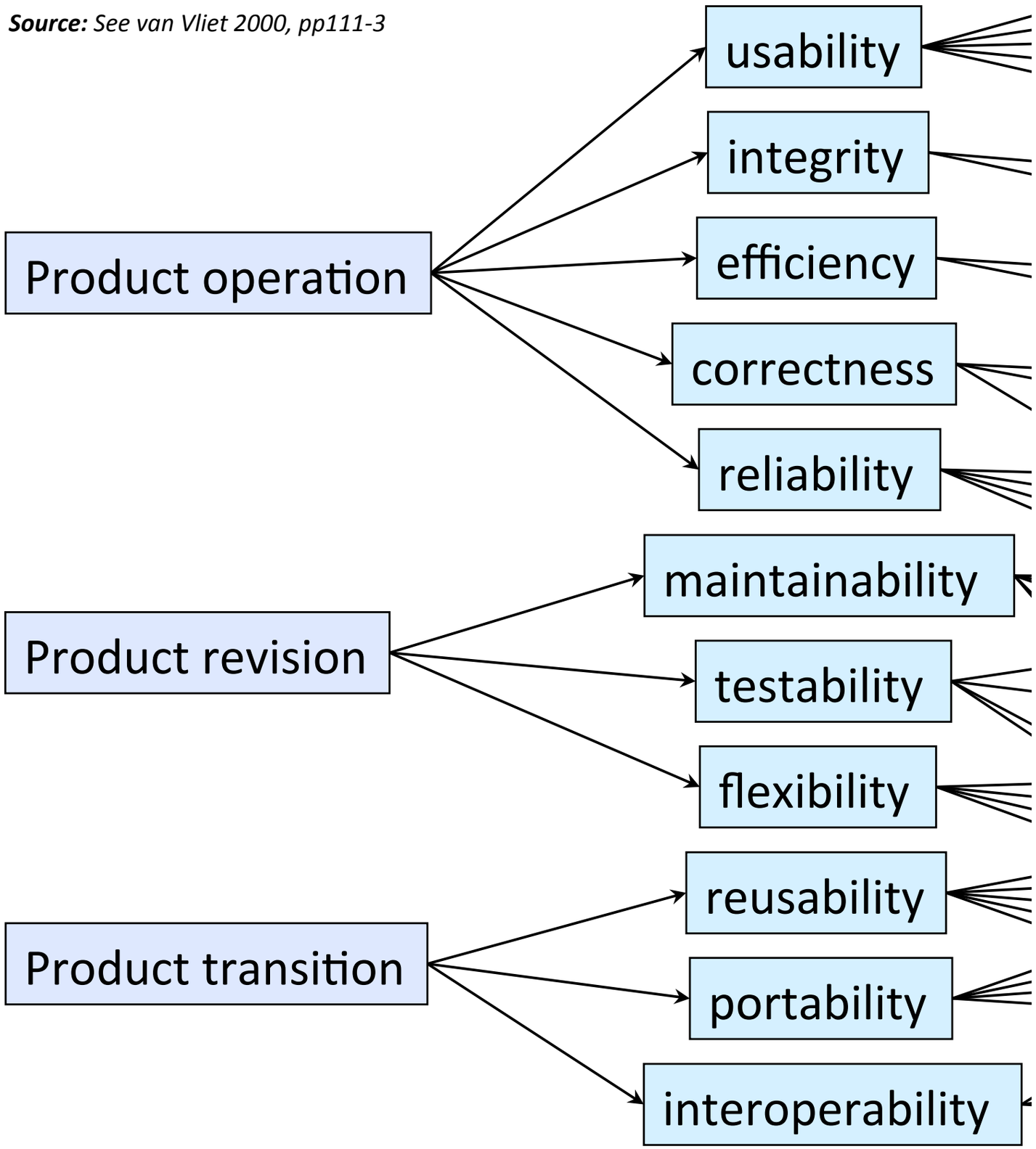
Evaluating NFPs

- Testability
  - Controllability
    -
    -
  - Observability
    -
    -
  - Repeatability
    -

Evaluating NFPs

- Testability
  - Controllability
    - time required to roll back
    - programmatic support for deployment
  - Observability
    - internal state must be externally visible (interceptors / loggers)
- Repeatability
  - automated access required for efficient testing
    - (may contradict with security constraints)

Source: *See van Vliet 2000, pp111-3*



Source: *See van Vliet 2000, pp111-3*