Vorlesung

Methodische Grundlagen des Software-Engineering

im Sommersemester 2014

Prof. Dr. Jan Jürjens

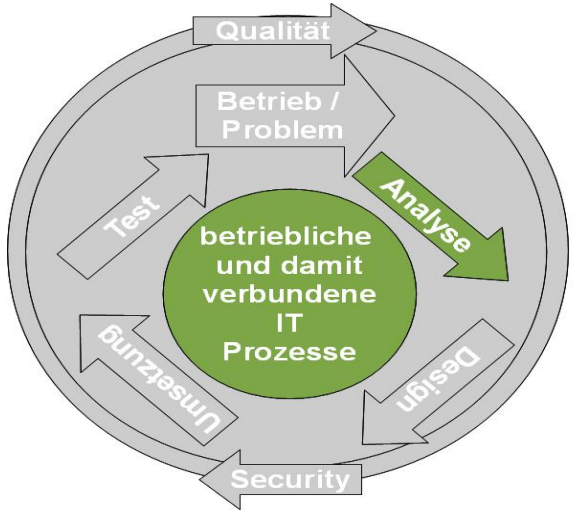
TU Dortmund, Fakultät Informatik, Lehrstuhl XIV

Teil 3.0: Einführung: Software Security

v. 17.06.2014

Einordnung
3.0 Einführung: Software Security

- Geschäftsprozessmodellierung
- Process-Mining
- Modellbasierte Entwicklung sicherer Software
  - Einführung: Software Security
  - Hintergrund IT-Sicherheit
  - Wiederholung: Modellbasierte Software Entwicklung
  - Modellbasierte Sicherheit mit UML
  - Sichere Architekturen
  - Kryptographische Protokolle
  - Protokollanalyse
  - Biometrische Authentisierung
  - Biometrische Authentisierung: Analyse
  - Elektronische Geldbörsen
  - Clouds
  - Elektronische Signatur
  - Bankarchitektur



Literatur:

[Jür05] Jan Jürjens: **Secure systems development with UML**, Springer-Verlag 2005. Unibibliothek (e-Book):
http://www.ub.tu-dortmund.de/katalog/titel/1361890
Papier-Version:
http://www.ub.tu-dortmund.de/katalog/titel/1091324
Kap. 4

- **Dieses Kapitel:**
  Modellbasierte Entwicklung sicherheitskritischer Software
- **Dieser Abschnitt:**
  Einführung: Software Security
  - Warum sichere Systeme?
  - Schwierigkeiten
  - Modellbasierte Entwicklung

- IT-Sicherheits-Risiken
- Schwierigkeiten bei der Softwareentwicklung
- Modellbasierte Entwicklung sicherer Systeme
- Beispiele

IT-Sicherheit

- Wirtschaft, Unternehmen und Gesellschaft hängen zunehmend ab von **Computernetzwerken** für
  - Kommunikation,
  - Finanzen,
  - Energieversorgung,
  - Transport...
- **Angriffe** können großen finanziellen Schaden verursachen.
- Vernetzte Systeme anonym und aus Entfernung angreifbar.

→ Computersysteme müssen **sicher** sein.

Problem: Sicherheit

Sicherheit (*Security*):

- Schutz von Daten/Systemen gegen **mutwillige Angriffe. Inhärent schwierig** (zielorientierter Angreifer).
- Beispiel (1997):
  - NSA Hacker Team bricht in U.S. Department of Defense Computer und U.S. Strom-versorgungssystem ein.
  - Demonstriert Strom- und Notrufausfälle in Washington, D.C..

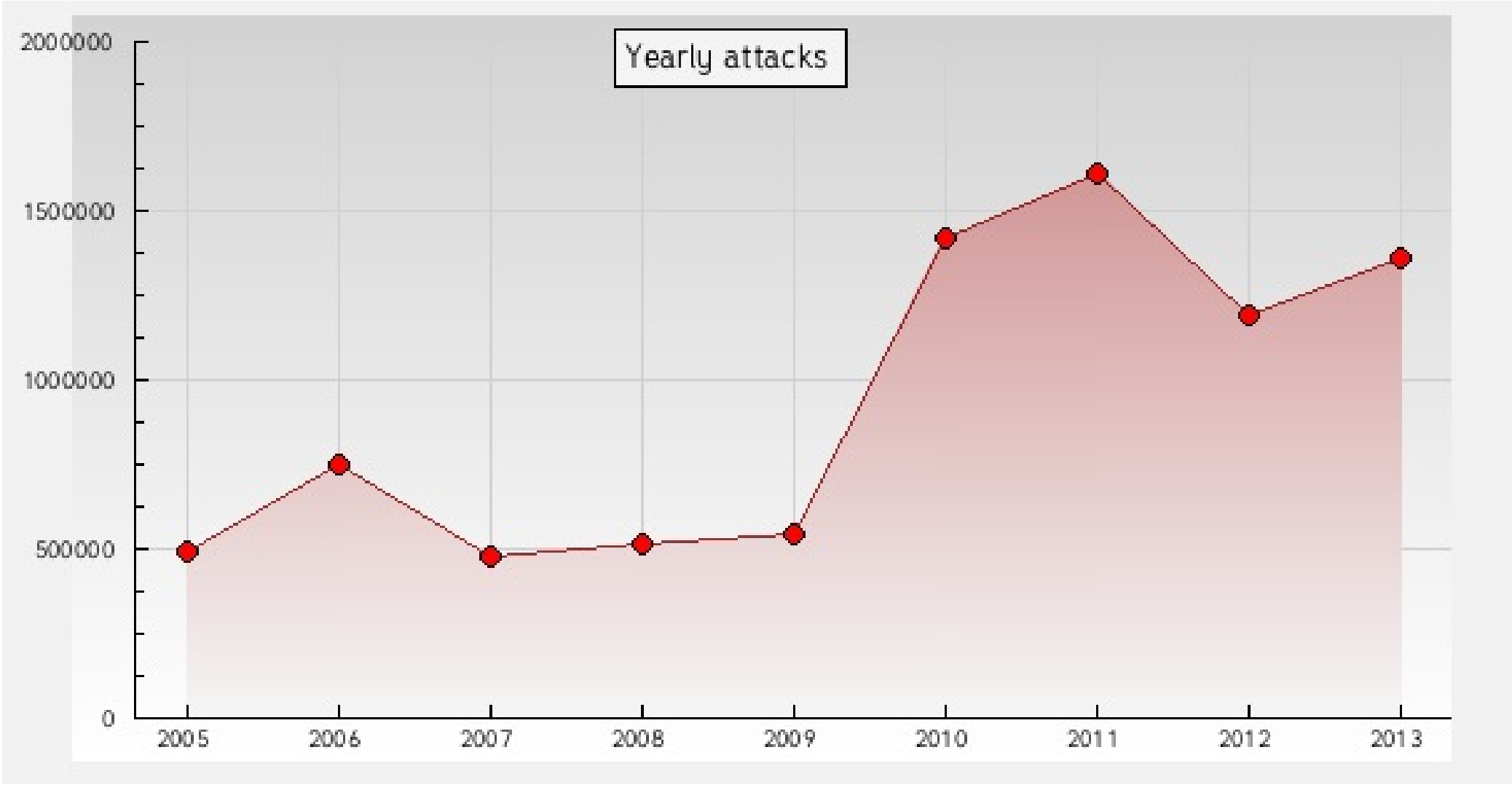
Beispiele für Sicherheitsvorfälle

- Einbruch in die Website **SalesGate.com:**
  - Diebstahl von **3.000** Kundendateien (z.B. Kreditkartennummern).
  - Z.T. im Internet veröffentlicht.
- Unkontrollierte Weiterleitung **persönlicher Informationen**
  - aus Hersteller-Sites (z.B. Finanzrechenprogramme auf **Intuit**)
  - zu Anzeigen-Sites (wie **DoubleClick**)

  ohne Wissen der Benutzer und oder von Intuit.
- Februar **2000**: massive **Denial-of-Service Angriffe**.

[Schneier: Secrets & Lies]

Computerattacken Allgemein

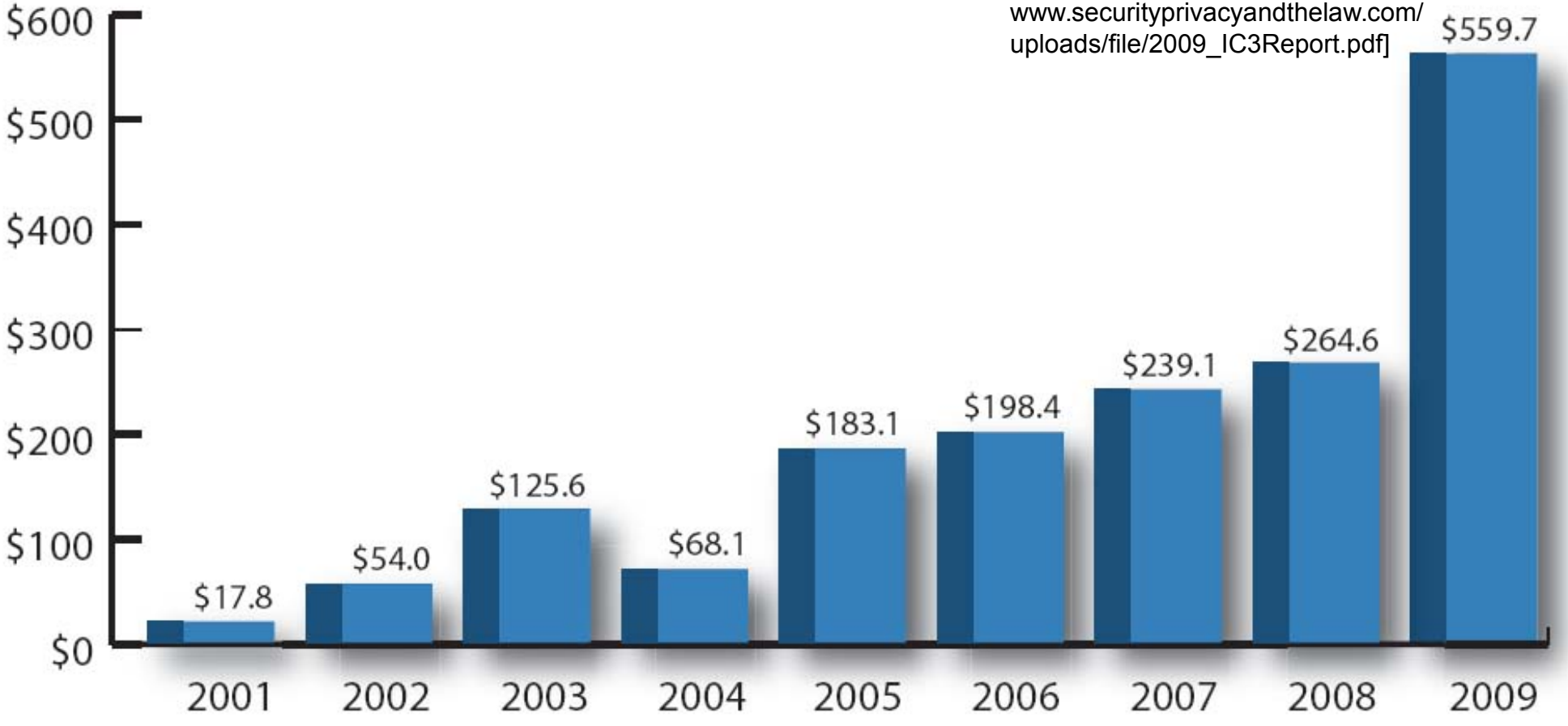


[zone-h.org, 2014]

Finanzieller Schaden

Figure 2: Yearly Dollar Loss (in millions) of Referred Complaints

[2009 Internet Crime Report, US Dep. Of Justice
www.securityprivacyandthelaw.com/
uploads/file/2009_IC3Report.pdf]



Beispiel (2010): Virusangriff auf Iranisches Nuclearprogramm

http://www.washingtonpost.com/world/national-security/stuxnet-was-work-of-us-and-israeli-experts-officials-say/2012/06/01/gJQAInEy6U_story.html

Stuxnet was work of U.S. and Israeli experts, officials say

A damaging cyberattack against Iran's nuclear program was the work of U.S. and Israeli experts and proceeded under the secret orders of President Obama, who was eager to slow that nation's apparent progress toward building an atomic bomb without launching a traditional military attack, say current and former U.S. officials.

The origins of the cyberweapon, which outside analysts dubbed Stuxnet after it was inadvertently discovered in 2010, have long been debated, with most experts concluding that the United States and Israel probably collaborated on the effort. The current and former U.S. officials confirmed that long-standing suspicion Friday, after a New York Times report on the program. […]

Overall, the attack destroyed nearly 1,000 of Iran's 6,000 centrifuges — fast-spinning machines that enrich uranium, an essential step toward building an atomic bomb. The National Security Agency developed the cyberweapon with help of Israel.

Beispiel (12.06.2011): Hacker attackieren den Währungsfond

http://www.zeit.de/digital/datenschutz/2011-06/hacker-iwf-wirtschaftsdaten/komplettansicht?print=true

Hacker haben den Internationalen Währungsfonds angegriffen und offenbar Daten gestohlen. Der IWF geht von Spionage aus und macht eine "bestimmte Regierung" verantwortlich.
Der Internationale Währungsfonds (IWF) ist Opfer einer Attacke auf seine Computer geworden. Nach einem Bericht von Bloomberg News wurden bei dem Angriff E-Mails und weitere Dokumente gestohlen. Der Fonds habe Ermittlungen eingeleitet, wie es zu dem Hacker-Angriff kommen konnte, erklärte ein IWF-Sprecher. Die Arbeit der Organisation sei durch den Angriff aber nicht beeinträchtigt. Über das Ausmaß des Schadens machte der Sprecher keine Angaben.
Ein Sprecher der Weltbank sagte, man habe alle Netzwerk-Verbindungen zur Schwester-Organisation gekappt. Es handle sich dabei aber um eine reine Vorsichtsmaßnahme, bis man die Attacke und ihre Auswirkungen genauer verstanden habe. Das Netzwerk ermöglicht den Austausch von Informationen zwischen beiden Organisationen, wird aber nicht für die Kommunikation geheimer Informationen oder sensibler Finanzdaten genutzt.
Auch FBI leitete inzwischen eine Untersuchung ein. Die Bundespolizei arbeite eng mit dem IWF zusammen, erklärte eine Sprecherin des US-Verteidigungsministeriums. Das FBI selbst lehnte eine Stellungnahme ab.
Nach Angaben des Internet-Sicherheitsexperten Tom Kellermann, der in dieser Funktion auch für den IWF und die Weltbank gearbeitet hat, zielte der Hackerangriff darauf, heimlich eine Software zu installieren, um einer bestimmten Regierung Zugang zu Insider-Informationen des IWF über andere Länder zu verschaffen. Um welche Regierung es sich handle sei noch unklar.
Der Angriff habe sich über mehrere Monate ereignet, zitierte die New York Times einen namentlich nicht genannten IWF-Mitarbeiter. "Das war ein sehr großer Einbruch." Die Organisation wollte offiziell jedoch nichts dazu sagen, wie umfangreich der Angriff war und welches Ziel er hatte.
Die Zeitung spekuliert über die Art des Eindringens, es habe sich dabei um "spear phishing" gehandelt. Beim Phishing werden unbedarfte Nutzer meist in einer Mail aufgefordert, Passworte und Zugangscodes auf Seiten einzugeben, die aussehen wie echte – beispielsweise für das Onlinebanking – jedoch echten nur nachempfunden sind, um diese Codes abzufangen. Spear phishing ist eine gezieltere Form dieses Verfahrens. Die Mail kommt dabei von jemandem, den der Nutzer kennt und dem er vertraut, der IT-Abteilung des Hauses beispielsweise.

Einige der größten Schäden durch Cyber-Angriffe

http://www.businesspundit.com/10-most-costly-cyber-attacks-in-history

2011: Exposure of over 100 million PlayStation Network and Sony Online Entertainment accounts is forging a new chapter in the history of cyber-attacks. The personal information — including credit and debit card data — of tens of millions of users was stolen by an as yet unknown group of assailants. Experts predict that the damage may range from $1 to $2bn, making it possibly the costliest cyber-hack ever to have been pulled off.
2008: Trusted payments processor Heartland Payment Systems fell victim to a plot to steal credit and debit card numbers. By secretly infesting the company's computer network with spyware, the criminal gang responsible were able to steal over 100 million individual card numbers. The episode ended up costing around $140m.
2007: Grocery retailer Hannaford Bros suffered a four-month long breach of their security from the winter of 2007 to the spring of 2008. During this period, over 4.2 million credit and debit card numbers were exposed, along with other sensitive information. Experts table the costs incurred at an estimated $252m.
2005: Massachusetts-based retailing company TJX was attacked by a gang able to get their hands on over 45 million credit and debit card numbers, a selection of which they then used to fund a multi-million dollar spending spree from Wal-Mart's stock of electronics equipment. The damage from the data-breach ended up costing over $250m in total.
2004: Sven Jaschan unleashed a virus which infected millions of computers around the world, reaching its highest degree of destruction when it comprehensively disabled the Delta Air Lines computer system, causing the cancellation of several transatlantic flights. Jaschan was eventually arrested after a three-month hunt, during which Mircosoft placed a $250,000 bounty on the hacker's head. An estimated $500 million worth of damage was generated.
2000: 15-year-old Michael Calce conducted notorious attacks against huge companies with high levels of security. Amongst those attacked were computer manufacturer Dell, media giant CNN, and shopping sites Amazon and Ebay. Prosecution for the estimated $1.2bn worth of damage caused went pretty smoothly, from Calce's perspective. He ended up with a sentence of eight months open custody.

IT-Risiken vs. Basel II etc.

- Basel II: **risikogerechtere** Regelung der Eigenkapitalanforderungen.
- Genauere Analysemethoden (Kreditrisiko, **operationelles Risiko**, internal-ratings-based) offenzulegen.
- Insbesondere **IT Risiken** (unexpected loss, z.B. Virenbefall, Hackerangriff, …).

Ursachen I

- Korrektes Entwerfen sicherer Systeme **schwierig**.
- Selbst Experten irren sich:
  - Needham-Schroeder-Protokoll (1978)
  - Gefundene Angriffe 1981 (Denning, Sacco), 1995 (Lowe)
- Designern **fehlt** es an **Erfahrung** in Sicherheit.
- Sicherheit **im Nachhinein**.
- Angriffs-Informationen verbreiten sich schnell.
- Keine Rückmeldung über **Sicherheit aus Kundensicht**.

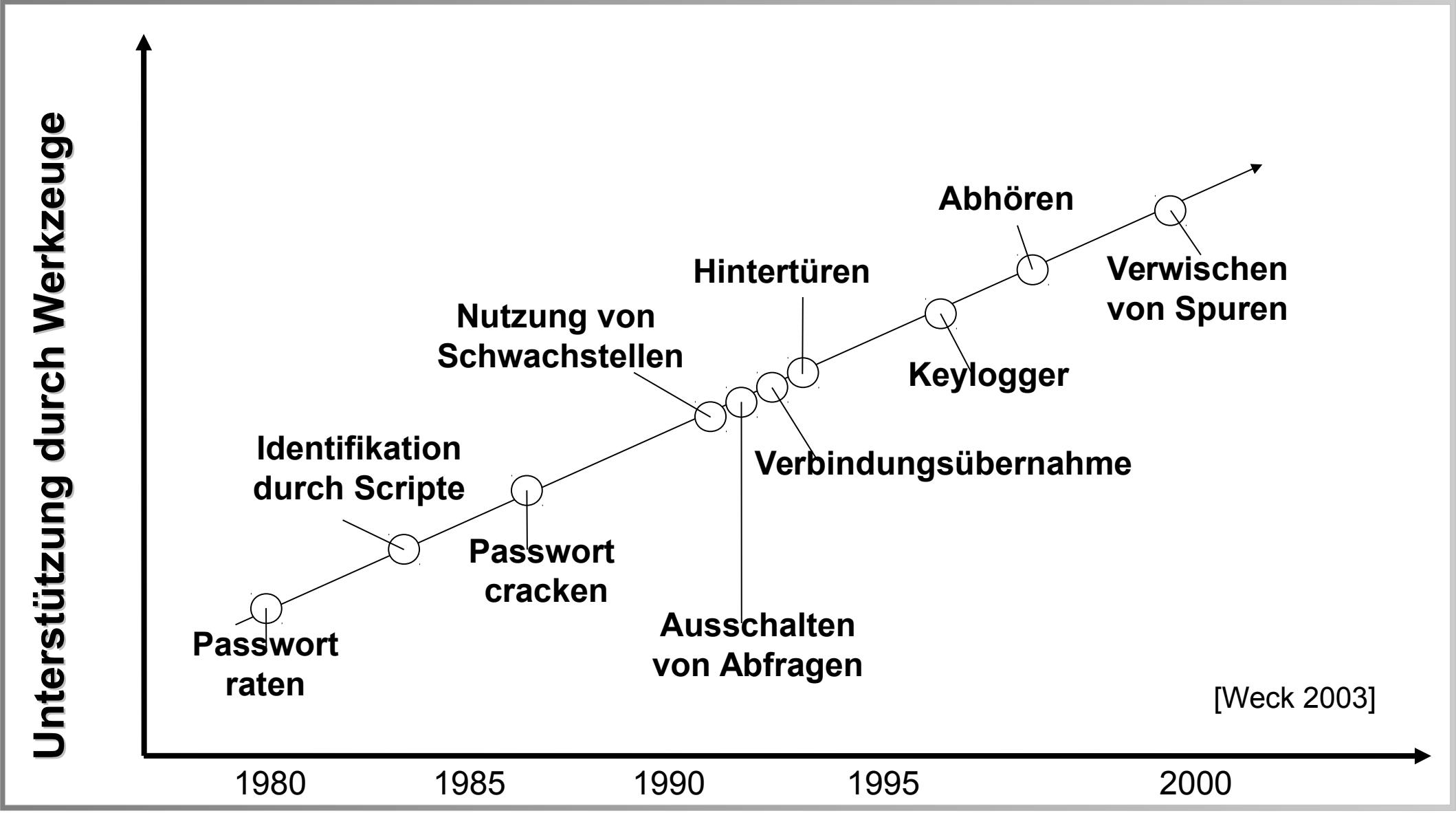
Ursachen II

„Blindes“ Vertrauen in Mechanismen:

- Sicherheitstechnik ist gefährdet, da sie **umgangen** werden kann (ohne **gebrochen** zu werden).

- Annahmen in Bezug auf System-**Kontext** und physische Umwelt.
  - "Diejenigen, die glauben ihre Probleme durch den einfachen Einsatz von Kryptographie lösen zu können, verstehen diese nicht, und verstehen ihr Problem nicht" (R. Needham)

Immer bessere Hackerwerkzeuge



[Weck 2003]

- IT-Sicherheits-Risiken
- Schwierigkeiten bei der Softwareentwicklung
- Modellbasierte Entwicklung sicherer Systeme
- Beispiele

Sichere Systeme

- Analyse von sicherheitskritischen Systemen **schwierig** (motivierter Angreifer).
- Viele entwickelte und eingesetzte Systeme genügen Sicherheitsanforderungen **nicht**.
- Sichere Produkte auf **unsichere** Weise eingesetzt.
- Viele z.T. spektakuläre Angriffe.
- Problem: **Qualität vs. Kosten**.

- "Penetrate-and-patch"

  (auch genannt "Bananen-Strategie“):

  - **Unsicher.**
  - **Störend.**
- Traditionelle formale Methoden: **teuer**.
  - **Ausbildung** von Entwicklern.
  - Erstellung **formaler Spezifikationen**.

Problem: Sicherheit ist schwierig

- **Klassische Schwäche** in alten Unix-Systemen:
  "Falsches Passwort"-Meldung nach Eingabe des ersten falschen Zeichens.

  Was ist damit das Problem ?

Problem: Sicherheit ist schwierig

- **Klassische Schwäche** in alten Unix-Systemen:
  "Falsches Passwort"-Meldung nach Eingabe des ersten falschen Zeichens.

  → Durch **Zeitmessungs-Angriff** reduziert sich Passwort-Raum von 26^n auf 26*n (n = Passwortlänge).

- **Neuere Schwäche** von Smart-Cards:
  - Rekonstruktion des geheimen Schlüssels.
  - durch Zeit-Messung des Stromverbrauchs während Crypto-Operationen.

  → **Frage:** Wie findet man diese Schwächen mit klassischen Tests?

Problem: Nicht vertrauenswürdige Programmierer

- Bei hoch-sicherheitskritischen Systemen:
  - Absichtlich eine **Hintertür** in den Code einbaubar.
- Diese durch Zufalls-/Black-Box-Tests zu finden. → Unmöglich!
- Versteckte Schwächen einbauen und verwenden (vorherige Folie).
  → Schlimmer.

→ **Frage:** Wie kann man dies verhindern **?**

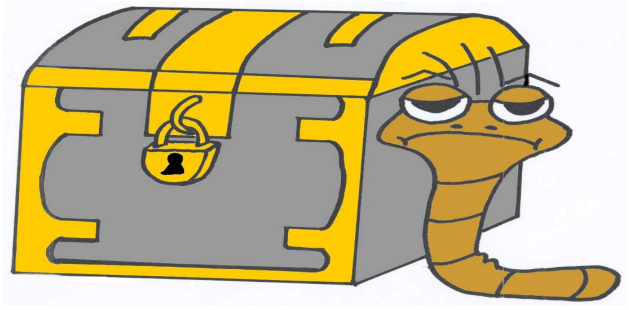
Besonderes Problem: Kryptographie

- Kryptographie: wichtige Rolle in sicherheitskritischen Anwendungen.
- Per Definition: sicher gegen **Brute-Force-Angriffe** sein.

→ **Paradox**: Wie bekommt man **ausreichend Testabdeckung** (für die aus Internet erreichbaren Eingabeschnittstellen für bestimmten Angreifer) eines Systems, um sicher gegen Brute-Force-Angriffe zu sein?

- IT-Sicherheits-Risiken
- Schwierigkeiten bei der Softwareentwicklung
- Modellbasierte Entwicklung sicherer Systeme
- Beispiele

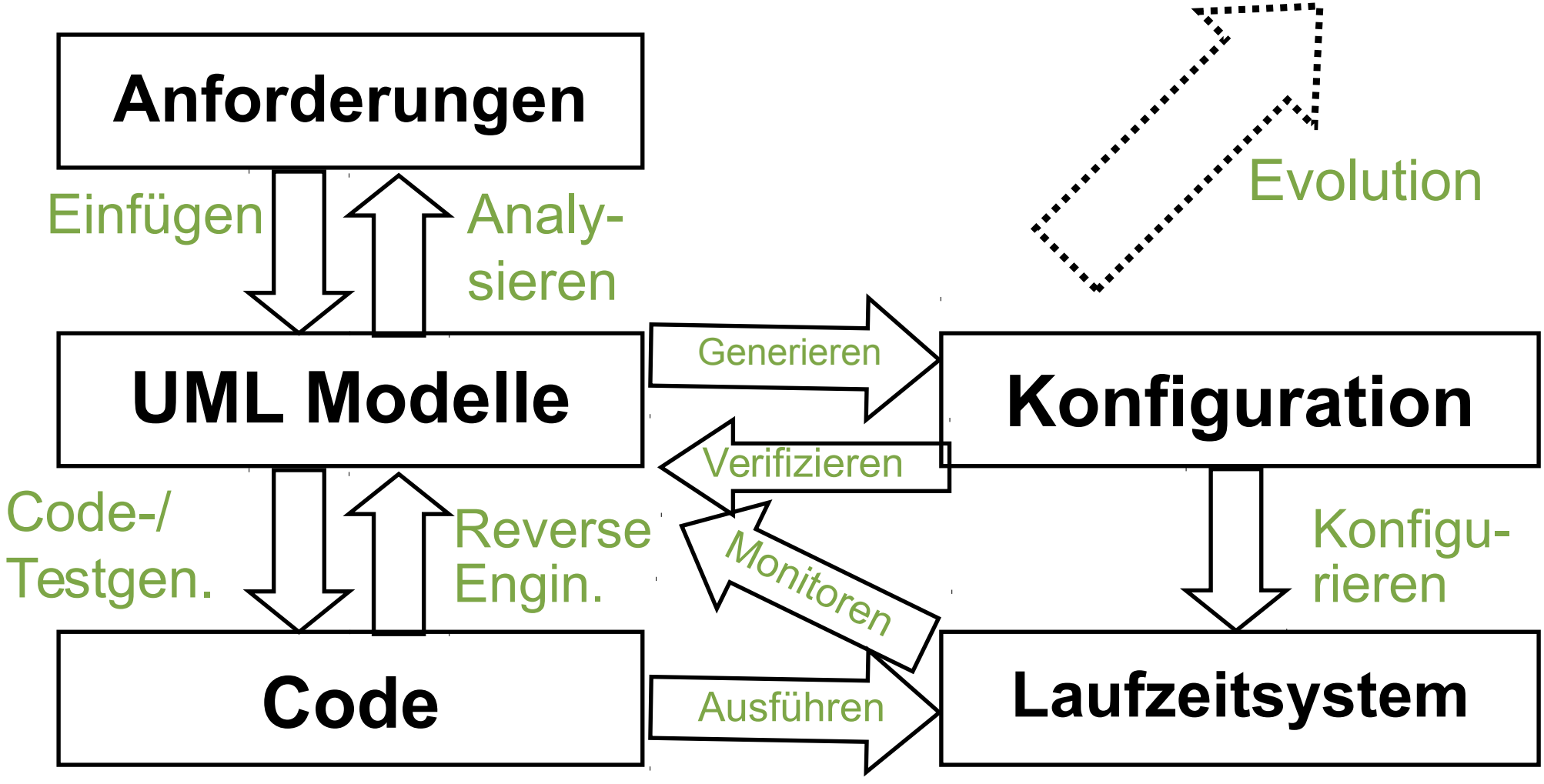
Ziel: Sicherheit erhöhen, bei begrenzter Investition an **Zeit und Kosten**.



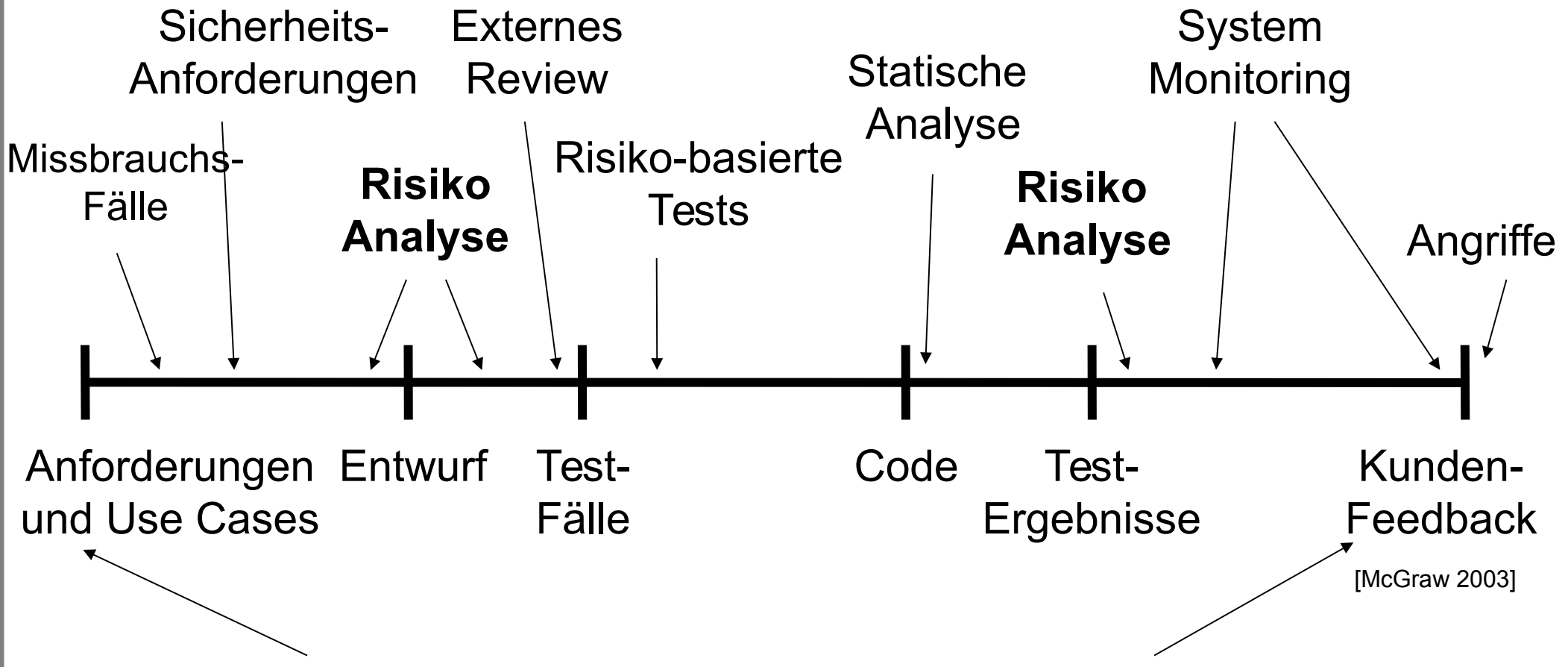
Lösungsansatz:

- Aus **Artefakten** in **industrieller Entwicklung** und **Betrieb sicherheitskritischer Software: Modelle** extrahieren (UML, Quellcode, Konfigurationen).
- **Werkzeugunterstützung** für theoretisch fundierte, effiziente (automatische) Sicherheitsanalyse.

→ **Modell-basierte Entwicklung sicherer Systeme.**


Anforderungen
Einfügen
Analy-
sieren
Evolution
UML Modelle
Generieren
Konfiguration
Verifizieren
Code-/
Testgen.
Reverse
Engin.
Monitoren
Konfigu-
rieren
Code
Ausführen
Laufzeitsystem

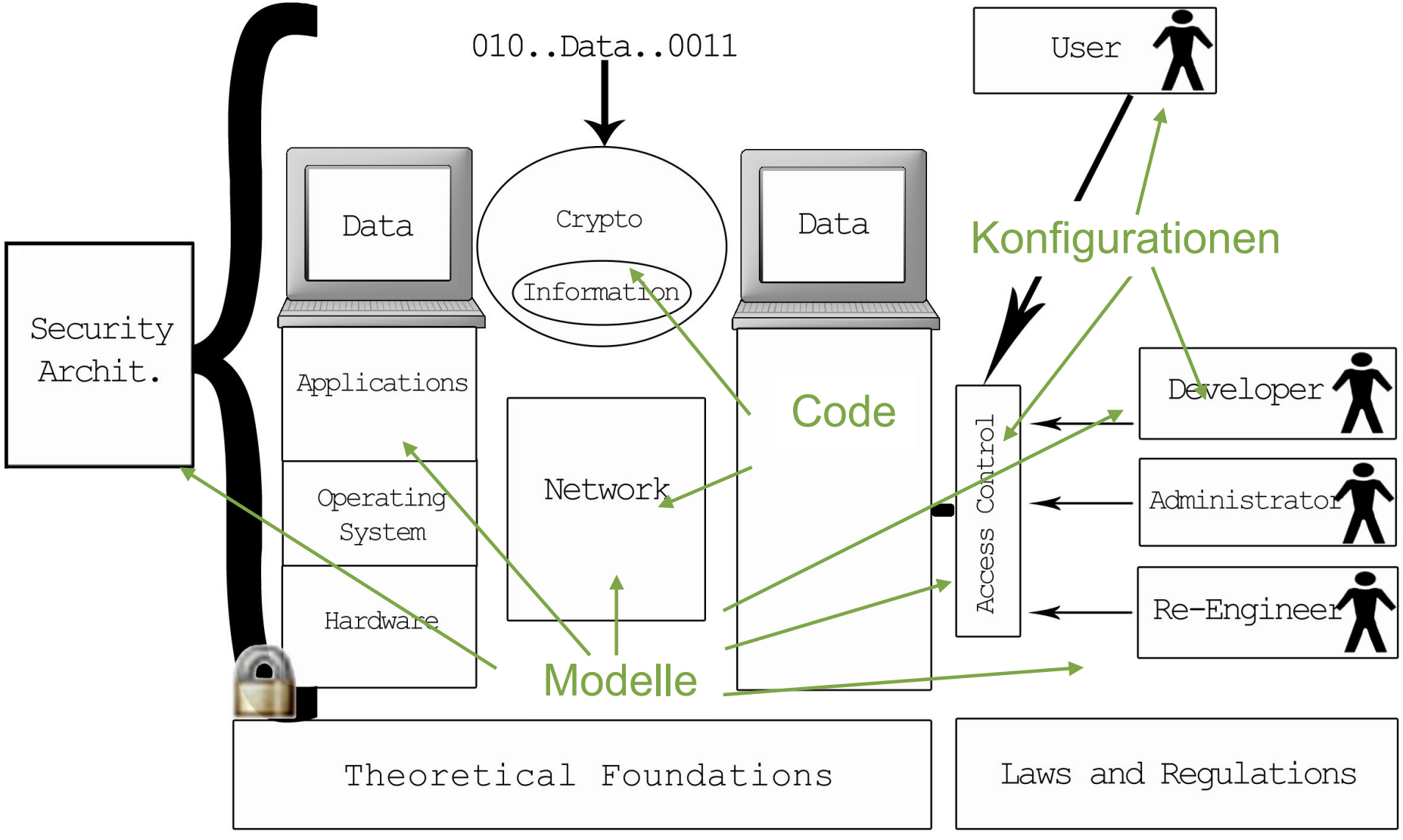
Lebenszyklus eines sicherheitskritischen Systems



[McGraw 2003]

Modell-basierte Entwicklung sicherer Systeme

Architekturebenen



Einsatzfelder

- **Systementwurf:**
  - z.B. Architekturbewertung, Plattformenwahl, Altsystemeinbindung.
  - **Autom. Sicherheits-** und **Risikoanalyse modellierter GP und Softwarearchitekturen** unter Einbezugnahme des Systemkontexts.
- **Implementierung:**
  - Quellcodeanalyse, Testfolgengenerierung.
- **Laufender Betrieb:**
  - Konfigurationsmanagement, Überprüfung von Berechtigungen, Einrichtungen von Firewalls…
  - **Autom. Checks** von **Systemkonfigurationen** (z.B. SAP Berechtigungen, …).
- **Sichere GP/WF: Berücksichtigung von Sicherheit ab Geschäftsprozessentwurf.**
- Einsatz in **Sicherheitsaudits:** Erhöht Vertrauen in **Korrektheit** und **Vollständigkeit**.

Warum UML ?

Modellbasierte Sicherheitsanalyse von GP und Softwarearchitekturen mit Unified Modeling Language (UML):

- **De-facto Standard** in industrieller Modellierung. **Große Anzahl von Entwicklern** in UML ausgebildet.
- Einfache, intuitive **Notation,** relativ **genau definiert.**
- **Weitgehende Werkzeugunterstützung** (auch für Code-Generierung, Analyse, Reverse Engineering, Simulation, Transformation).

Warum UML ?

- **Ziel**: Übertragen von Resultaten aus **Forschung** in formalen Methoden in **praktische Softwareentwicklung.**
- Verwendung durch Entwickler (nicht in formalen Methoden ausgebildet), um
  - Sicherheit von erstellten Komponenten zu prüfen.
  - vorhandene Sicherheits-Komponenten korrekt im Systemkontext einzusetzen.
  - Systemumgebung betrachteter Komponente zu analysieren.

UML-Erweiterung UMLsec

Erweiterung für Entwicklung **sicherer Systeme:**

- Auswertung von UML-Spezifikationen für Sicherheitsanalyse während Entwicklung.
- Beinhaltet **etablierte Sicherheits-Regeln** als **Checkliste.**
- **Nicht** auf sichere Systeme **spezialisierte** Entwicklern zur Verfügung gestellt.
- Berücksichtigt Sicherheitsanforderungen **von Beginn** der System-Entwicklung und im -**Kontext**.
- **Sicherer Entwurf durch Modellanalyse**.
- Sichere Implementierung durch modellbasierte **Testgenerierung**.
- Macht Zertifizierung **kostengünstiger**.

UMLsec: Wie?

- Wiederkehrende **Sicherheitsanforderungen**, **Angriffs-Szenarien**, und Sicherheits-Konzepte als Stereotypen und Tagged Values angeboten.
  [Stereotypen: Markierungen an UML-Modellelementen, z.B. <<call>>.
  Tagged Values: Annotationen der Form (tag=value), mit denen weitere Informationen übergeben werden. → vgl. Abschnitt 3.2]
- Verwenden Constraints, um Spezifikationen unter Verwendung automatischer Werkzeuge zu **verifizieren** und mögliche Schwächen anzuzeigen.
  [Stereotypen können mit vordefinierten Constraints verbunden werden.]
- Garantiert, dass UML Spezifikation gewünschtes Niveau der Sicherheitsanforderungen **liefert**.
- Verbindung zum Code über Round-Trip.

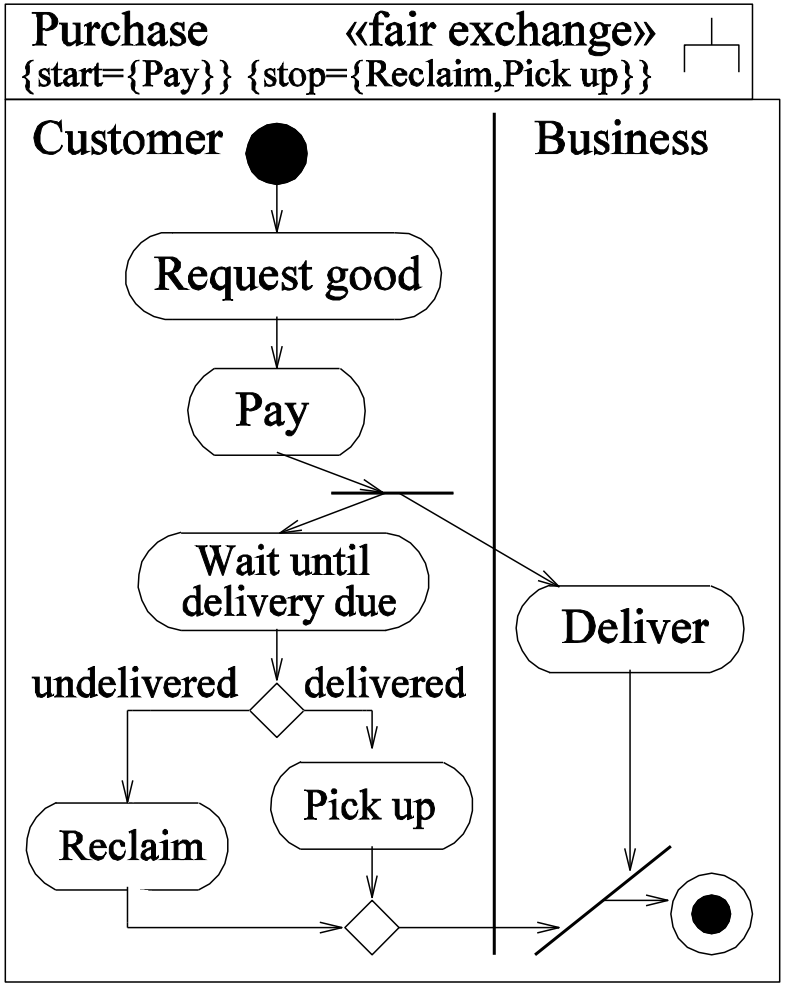
Herausforderungen

- **Sichere Systeme** aus (un)sicheren Komponenten ?
- Sicherheit als **pervasive Eigenschaft**: vs. Zuverlässigkeit, Programm-Analyse, formale Methoden, Software Engineering, Programmiersprachen, Compiler, Computer-Architekturen, Betriebssysteme, reaktive Systeme, ..., ...
- Problem: **keine Integration / Kohärenz**.
- Wie bekommt man diese Aspekte im Kontext der Entwicklung sicherer Systeme zusammen? **Notwendig** für Sicherheit (Angriffe auf Grenzen zwischen Ansichten / Aspekten / Ebenen ...).

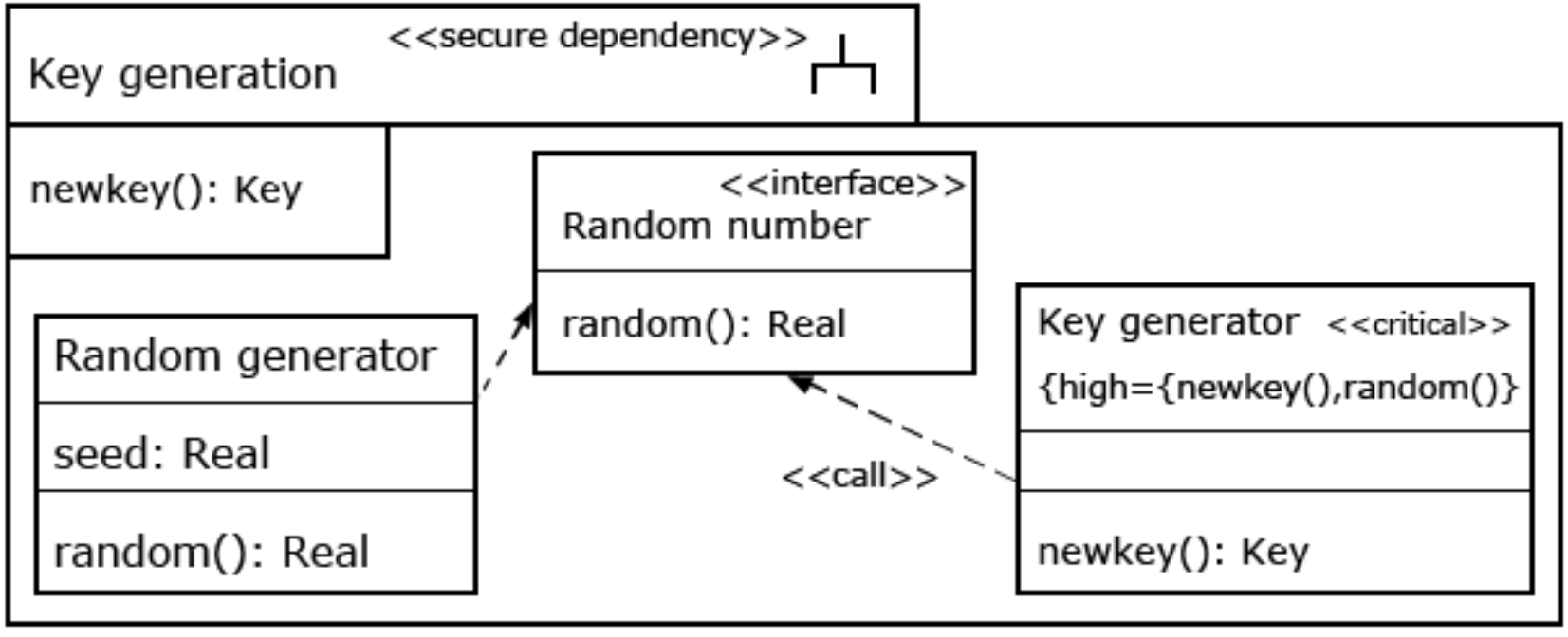
- IT-Sicherheits-Risiken
- Schwierigkeiten bei der Softwareentwicklung
- Modellbasierte Entwicklung sicherer Systeme
- Beispiele

Beispiel: Sichere e-Transaktionen

- Sicherheit von GP z.B. bei e-Transaktionen.
- Hier: Kunde kauft Ware beim Händler.
- Nach Bezahlung bekommt Kunde Ware **ausgeliefert** oder kann Bezahlung **zurückfordern**.

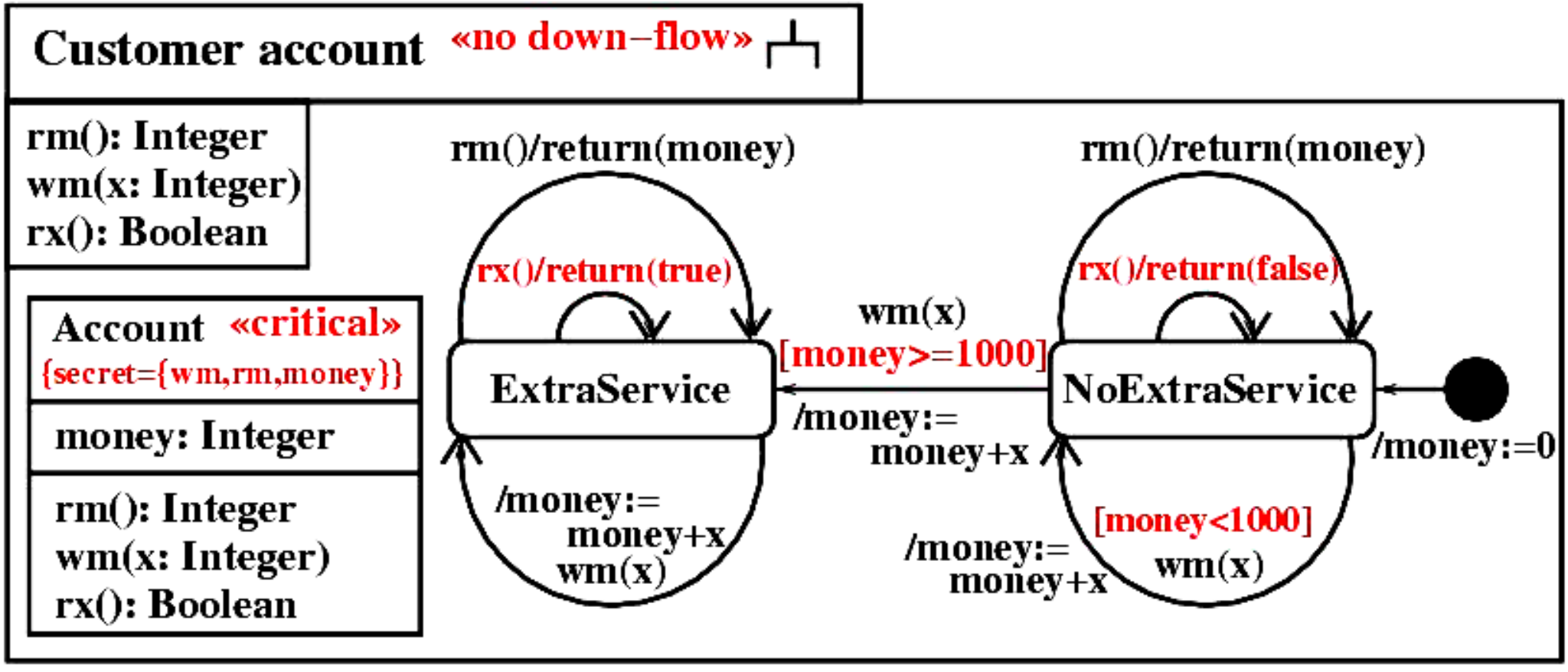


Beispiel: Durchgängige Datensicherheit



Sicherheitslevel definieren. **Konsistenzanalyse**.

Beispiel: Versteckte Informationsflüsse

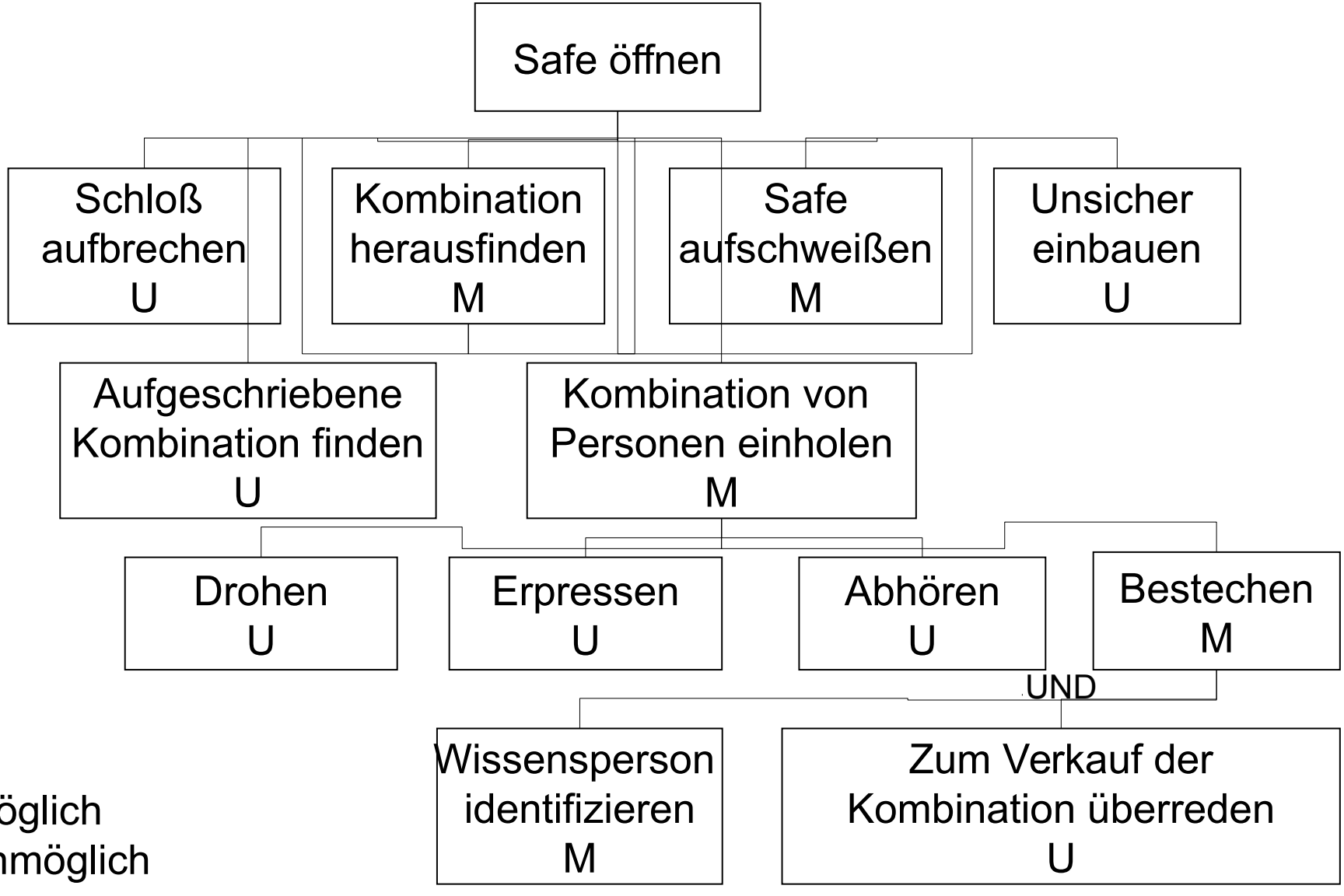


Können **vertrauliche Daten heraussickern** ?

Ohne Werkzeugunterstützung oft nicht ersichtlich.

Technik für Sicherheitsanalysen: Angriffsbäume

[Weck 2003]



M = Möglich
U = Unmöglich

Zusammenfassung

Dieser Abschnitt:

- IT-Sicherheits-Risiken
- Schwierigkeiten bei der Softwareentwicklung
- Modellbasierte Entwicklung sicherer Systeme
- Beispiele

Nächster Abschnitt: Hintergrund IT-Sicherheit