München, 24. April 2017

Wie verträgt sich Digitalisierung mit vorhandener (Alt-)Software?

Objektforum München 2017
Forever young – Bestandssysteme für die Digitalisierung fit machen

Dr. Markus Pizka

OriginalHealth.net

Portrait

- **itestra GmbH**
  - Ursprung: Technische Universität München
    - Betriebssysteme, High Performance Computing
    - Software (Re-)Engineering, Qualität, Ökonomie
  - Gründung 2004, 60 Mitarbeiter, 4+ Standorte
- **Leistung**: Strategieberatung und Engineering

München

Governance & Renovierung
„Altes fit machen“

Business Solutions
„Neue Dinge“

- **Kunden:**
  - Primär namhafte Großunternehmen ~ Versicherungen, Banken, Automobil, Handel
  - In Summe ca. 100 Applikationen mit in Summe mind. 500 Mio LoC betrachtet.

Vielfältige Alt- und Neu-Technologien

itestra
Software Productivity

	BUSINESS	MAINFRAME	MID & SMALL	S.PURPOSE	Web & Mobile
Programmier-sprachen	**Java**, C#, C++, PL/SQL , ABAP	**COBOL**, PL/I, **ASM**, NATURAL, Synon, COOLGEN, SQL	RPG, VB/A Python, PHP, Tcl/Tk, Gupta	C, Ada, Lisp, MATLAB, Pearl	ObjectiveC, Java, JavaScript
Frameworks Bibliotheken	Java EE, .net, SAP ERP (MM, SD, PP(-PI), LE, FI/CO)		Typo3, vTiger, openCMS/ERP	MP/PVM, SOLR, JavaFX	Struts GWT, PrimeFaces, MyFaces AngularJS, Node.JS, Backbone.JS, EmberJS React.JS,Apache Cordova
OR-Mapping	Hibernate, TopLink				
Datenbanken	Oracle, DB2, PostgreSQL, SQL Server	DB2, IMS/DB, VSAM	MySQL, Access, MariaDB	SQLite	HTML5 Storage, MongoDB
Plattformen	Linux, Solaris, Windows SAP Web-AS 6, SAP NetWeaver7	MQ CICS, IMS/DC z/OS, Linux	Windows, Linux	LynxOS, Windows RT	Android, iOS, Windows Mobile
Schnittstellen	MQS, SOAP, REST, JSON, CTG, SAP NPI, Spring Integration uvm,				

SOFTWARE ENGINEERING

Unified Process, Scrum, V-Modell, Use Cases, OO Design, UML, Inspektionen, CM, KM, etc.

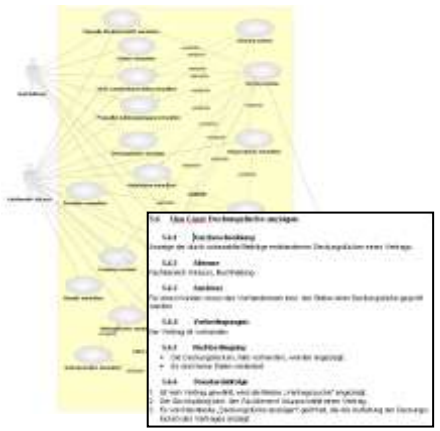
(Extrem)beispiele aktiver Alt-Systeme

itestra
Software Productivity

- **1968**
  - Großbank, Orderbuchung
  - COBOL ~ 50.000 LoC
    (GO TO, noch kein „PERFORM“)
- 1970
  - Versicherung, Leben
  - **5 Mio LoC Assembler**
- 1980+
  - Diverse Systeme proprietärer „4 GLs“
  - Coolgen, VaGen, Gupta, PowerBuilder, …
- 2000 ~ PHP, erste Java Systeme, …

Beispiel: technische Runderneuerung

Manuelles Reverse Engineering und Rewrite.



Use Cases

Geschäftsregeln

Text-UI

RPG Code

Dialoge

Informationsmodell

Java-Implementierung

Vollverantwortung, Werk/Festpreis, i.W. ohne Fachbereich

Eingangsfragen

Vor wenigen Tagen
in meiner Inbox:

*One of many reasons **younger, tech-centric firms often outpace** their larger, **established counterparts in** the **digital transformation** arena is **because legacy** systems **stand in the way** for many established firms.*

*… **digital disruptors will displace 4 out of 10** … companies over the next five years* … **impact of legacy systems on digital transformation***

Stimmt es, dass	Legacy-Systeme Digitalisierung behindern und Unternehmen zum Scheitern bringen?
Und …	was heißt modernisieren?

Vorab … es besteht Anlass zu Zweifel

itestra
Software Productivity

Wenn das so einfach wäre …

- wäre Digitalisierung etwas einmaliges
- oder die Unternehmen, die durch Dig 1.0 groß wurden, würden bei der Dig 2.0 aus dem gleichen Grund verschwinden.

Und:

- warum sollte ein Unternehmen, das noch nicht existiert, „greenfield“ Systeme schneller entwickeln als ein Unternehmen mit bereits vorhandener Software?
  (letztere können ebenfalls „greenfield“ neu entwickeln …)

Agenda für eine strukturierte Betrachtung

- Digitalisierung
  - Bedeutung und Anforderungen an die IT

- Legacy
  - Was ist das?
  - Welchen Einfluss hat Legacy und welchen nicht?

Bildarchiv Deutsche Rentenversicherung Bund

- Was tun mit „Legacy“?


OBI ;-)

- Ein Beispiel und ein Fazit

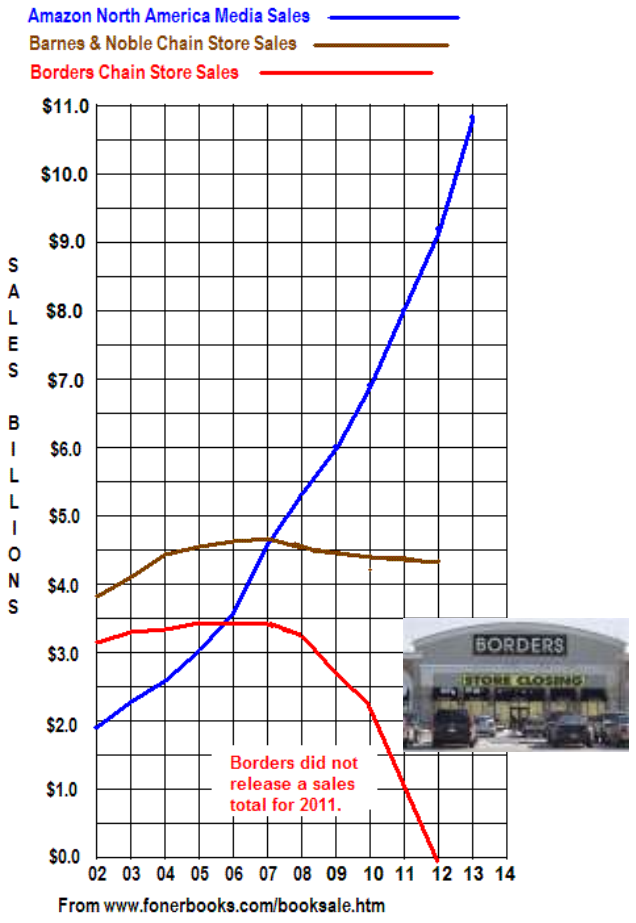
IMHO

- Radikale Veränderung von Produkten und Geschäftsprozessen
- Auf Grundlage der technischen Vernetzung und des Informationsaustausch zwischen Systemen, Geräten, Menschen.
  ➔ vernetzt, geteilt, mobil
- Jede Information ist jederzeit, ortsunabhängig, ohne Verzögerung, nahezu kostenlos verfügbar, änderbar, generierbar.
  Möglich durch: Rechenleistung, Speicherkapazitäten, schnelle (Funk)netze, mobile Geräte, Sensoren, wearables, IoT, AI, …
- Hohe Änderungsgeschwindigkeit, hohe Reichweite (PoS!), anonym/keine Markenloyalität, rasche Monopolbildungen

Die Wirkungen sind beobachtbar …

Wikipedia: „früher einer der weltweit bedeutendsten Hersteller für fotografische Ausrüstung“

Obwohl Kodak Pionier der digitalen Fotografie war (schon 1975)!



Unter Top-100 Unternehmen heute: Apple (1976/97), Google (1998), Facebook (2004), Amazon (1994)

Und erfasst weitere Branchen … auch uns

itestra
Software Productivity

SAP

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

itestra
Software Productivity

Print, Versicherungen, Banken, TV, Taxi, …

Software, Gesundheitswesen, Justiz, …

Was haben IT und Legacy damit zu tun?

itestra
Software Productivity

1) **Wenig**: in erster Linie sind Geschäftsinnovationen notwendig

2) **Viel**:
   - Technik ist Enabler und muss Treiber / aktiver Mitgestalter werden
   - Der Bedarf an neuen Lösungen ist enorm,
     besonders bzgl. **Customer Experience (UI)** und **Vernetzung**
   - Entscheidend:
     - **Time-to-System**
       Kapazitäten, Qualität,
       Produktivität

Geld
Bsp. PKV Oscar:
32,5 Mio $ von Google

3) **Und Legacy?** … kommt darauf an, ob
   für neues Produkt relevant, tatsächlich weniger produktiv, etc.

Schärfung des Begriffs „Legacy“

- Eigentlich sind Vermächtnisse kostbar →
- Im Kontext von Software:

Altsystem *(englisch legacy system) … etablierte, historisch gewachsene Anwendung … Vermächtnis … Altlast.*

*a **legacy system** is an old … application … a previous or **outdated** computer system.*

- **Der Begriff sagt sehr wenig aus** ~ “alt”!
  → Raum für **Spekulation** und **Emotion**
- Häufig assoziiert mit **älteren Technologien**, und missbraucht zur Diskreditierung.

Ältere Systeme sind Zeitkapseln

19. Jhdt.

Ca. 1998?

Zeitkapsel Software-System 1980

Kultur: proprietär	Open Source, Soc. Network
Kundenprozess:	
vor Ort in der Filiale, zwischen 8 und 16 Uhr	7x24
Papier, Bearbeitungszeit in Tagen	mobile
Elektronische Datenverarbeitung	BPM
Auftragsbezogene Programmierung	agil/enabler
Separate Anwendungen –UI bis DB	SOA
Daten Im-/Export	Interoperabilität
Nachtverarbeitung	sofort und online
Include, Call-Parameter	Vererbung, Aspekte
COBOL, PL/I u.a. from scratch	Java/JS/C#, Fwks
Felder, 1-fach, Listen, „Bubblesort“	Heap, Hash, …
Testlauf erfordert Anmeldung	Roundtrip lokal
Main Memory: kB / MB	GB / TB
Bildschirme mit 80x25 Zeichen	4K
Lizenzkosten ~ 5.000 DM	0 € ?

Technisch + fachlich + organisatorisch alt

itestra
Software Productivity

- **Können diese Systeme beibehalten werden?**
- **Ist ein Abbau der „technischen Schuld“ (z.B. Verletzung von Richtlinien) sinnvoll?**
- **Erscheint es sinnvoll, Systeme dieser Art unmodifiziert „zu wrappen“ und z.B. aus neuen UIs aufzurufen?**
- **Ist Übersetzen des Codes (z.B. COBOL→Java) einer solchen Zeitkapsel eine Option zur Modernisierung?**

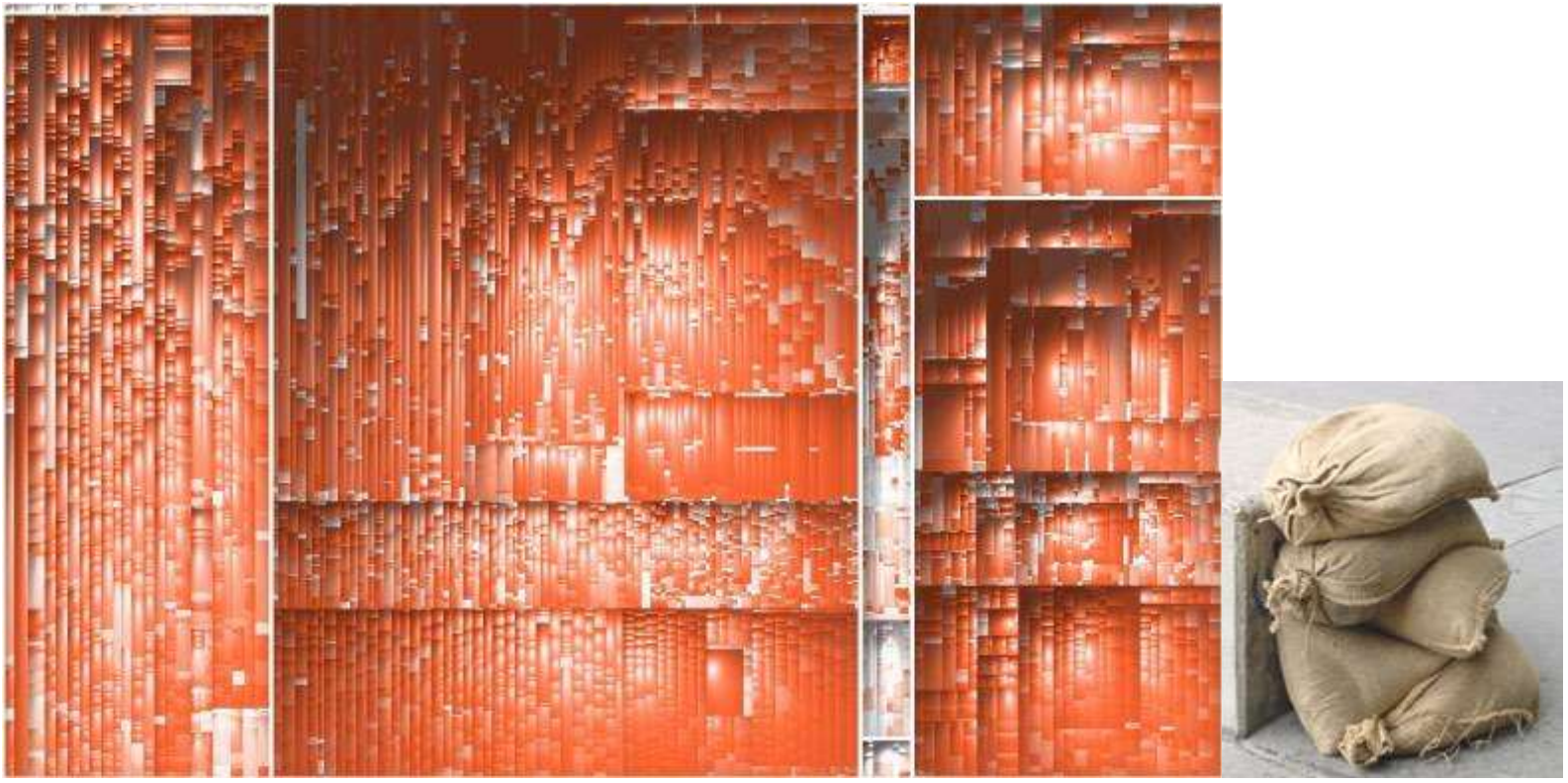
➔ nein, oder nur bedingt

„Technische Schuld“ gibt es natürlich auch

itestra
Software Productivity

Meist sehr groß und enormer Ballast

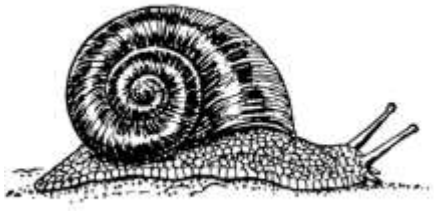
Copy + Paste: (3% best), 50% average, 90% worst



… zusätzlich nicht erreichbar („*dead code*“) häufig 30% + unused (?%) + …

Und äußerst inperformant!

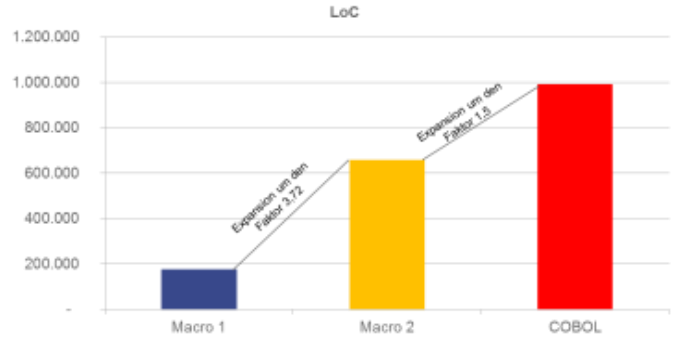
```
FOR Z = 1 TO ANZ  // für alle Elemente
CUR = FIRST;
WHILE (CUR != NULL && NOT LAST)
 IF CUR->NEXT != NULL
    THEN CUR = CUR->NEXT;
 ELSE
    LAST = '1';
 END;
... // APPEND ELEMENT
NEXT Z
```



Insert in O(n) statt O(1) (→ Listenaufbau: $O(n^2)$)
> 100.000 € p.a. Mehrkosten im Rechenzentrum!

Andere typische Mängel

- „Separation of Concerns“ fehlt
- Workarounds
- Dateien oder nicht normalisierte DB
- Keine/ungeeignete Datenzugriffsschicht
- Niedrige Datenqualität
- Dokumentation ist Pseudo-Code
- Kopfmonopole
- Keine Versionierung, kein Debug, …
- Programmgeneratoren, 3GL/4GL
  - SWT, TAA, VaGen, CoolGen, XCOBOL
  - Powerbuilder



Konsequenz

Geringe Effizienz

- Hohe **Kosten**: Entwicklung, Wartung, Betrieb
- Verlangsamung der Fortentwicklung ~ **Time-to-System**
- Evtl. auch Instabilitität

- Wettbewerbsfähigkeit geht verloren

Aber: nicht anhand Technologie pauschalieren

itestra
Software Productivity

- Es existieren auch Systeme:
  - COBOL o.ä. Sprache
  - z/OS mit IMS oder CICS
  - **keine „Zeitkapsel“!**

- 2 Beispiele: Partnerdaten- und Kernbankensystem
  - ✓ Implementieren aktuelle Fachprozesse
  - ✓ Service-Orientierte Architektur
  - ✓ Umfangreiche, normalisierte Datenhaltung
  - ✓ Testautomatisierung
  - ✓ **Niedrige Kosten und Time-to-System**

- **Back-End Systeme**, keine UIs! (vgl. „2-Speed IT“)

„Neue“ Systeme sind oft nicht besser

itestra
Software Productivity

- Neue **Technologien** = neue Möglichkeiten = neue Fehlermuster
  - Ungeeignete Eclipse Konfiguration (>300 Einzelprojekte?)
  - Zahllose (unnötige) Schichten und Indirektionen (>10)
  - Falsche Anwendung von Design-Patterns
  - Überflüssige aber aufwendig gepflegte Tests
  - Fehlende/falsche Thread-Synchronisation
  - Ungeeignetes Fehlerhandling (Exceptions, Return Codes, …)
  - Weiterhin: Duplikation, hard-coded, Kopfmonopole, ungeeignete Datenmodelle
    … und schon wieder **veraltete** Frameworks (Struts, etc.)

- Konsequenzen: erneut **sehr hohe Kosten** und **Verzug/mangelhafte Time-to-System**
- **Zwei Beispiele:**
  - Java, Projekt mit 200 MA, System: 1 Mio Zeilen, Kosten pro Zeile ca. 150 € ➔ Faktor 10 erhöht
  - Java, 0,9 Mio Zeilen, wenig technische Schuld, 8 Jahre, 70 Mio € ➔ Faktor 5-6 erhöht, gestoppt

Schärfung des Begriffs „Legacy“, die 2te

itestra
Software Productivity

1. Alt ≠ schlecht
2. Neu ≠ gut
3. Technologie alt ⇏ System veraltet
   (wenngleich gewisse Wahrscheinlichkeit ;-)

Fachlich, technisch, organisatorisch alt

Neu aber schlecht gemacht

Diverse Mängel
(aus aktueller Sicht)

Hohe Kosten, lange Time-to-System ≠ *Digitalisierung*

Was tun?

itestra
Software Productivity

Ein besserer Umgang

mit alten und neuen

„Legacy“-Systemen.

Weniger Emotion, Ironie und Ignoranz

„Nintendos“ vs. „Hosties“

Damit verbunden: weniger Tool- und Produkt („Standard“)-Gläubigkeit

Strukturierte Strategieentwicklung

itestra
Software Productivity

1. Standortbestimmung
„wo stehen wir?“

2. Zielbild
„wo wollen wir hin?“

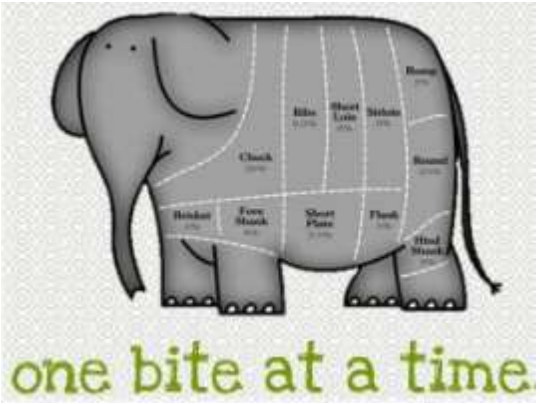
3. Roadmap
„welche Wege gibt es“?

Fachlich + Technisch
Ökonomisch + Organisatorisch

Fachliche Prios
Tech. Abhängigkeiten
Zeit, Personal

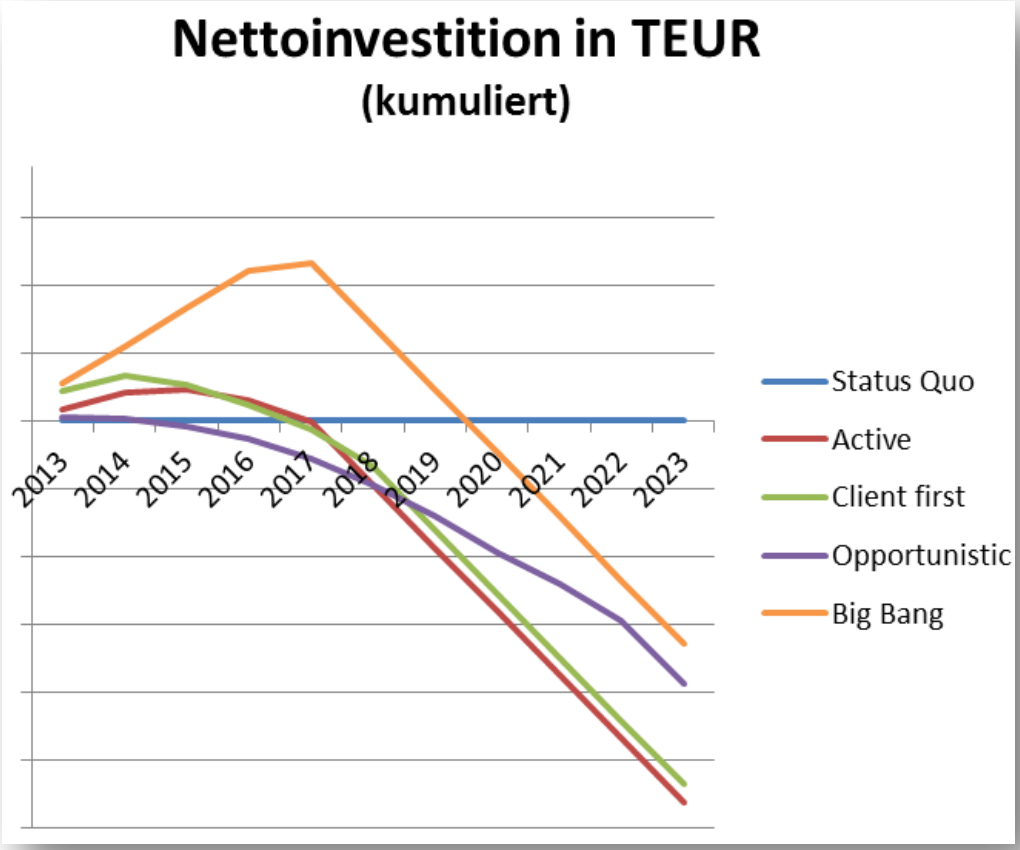
Roadmap-Schablone: schrittweise

- Aufgrund Umfang meistens notwendig
  - Schrittweise Erneuerung
  - Mischbetrieb alt/neu
  - über mehrere Jahre ~ 5-10
- Hierdurch
  - Keine parallele Doppelpflege
  - Kein fachlicher Freeze
  - Früher Nutzen
  - Steuerbarkeit
- Schnittlinien: in der Regel fachliche Einheiten (falls Schicht: UI)
- Integration auf Ebene der Daten (falls Schicht UI: Dienste)

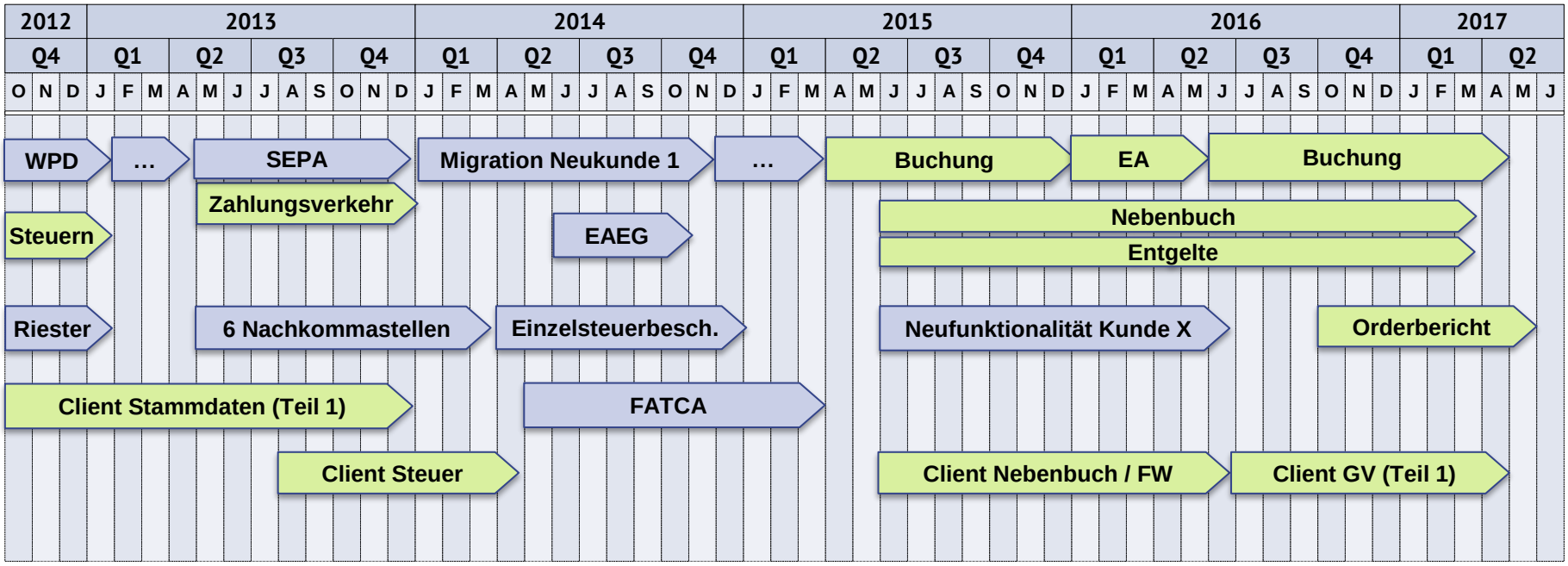


Fallbeispiel

- Bank
- Geschäftskritisch, hohes Volumen
- 1,6 Mio SLOC
- Von COBOL Monolith nach Java Services
- Verschiedene Roadmaps
  - Unterschiedlicher Invest
  - Andere Ergebnisse und Nutzenverläufe



Erfolgreiche Umsetzung



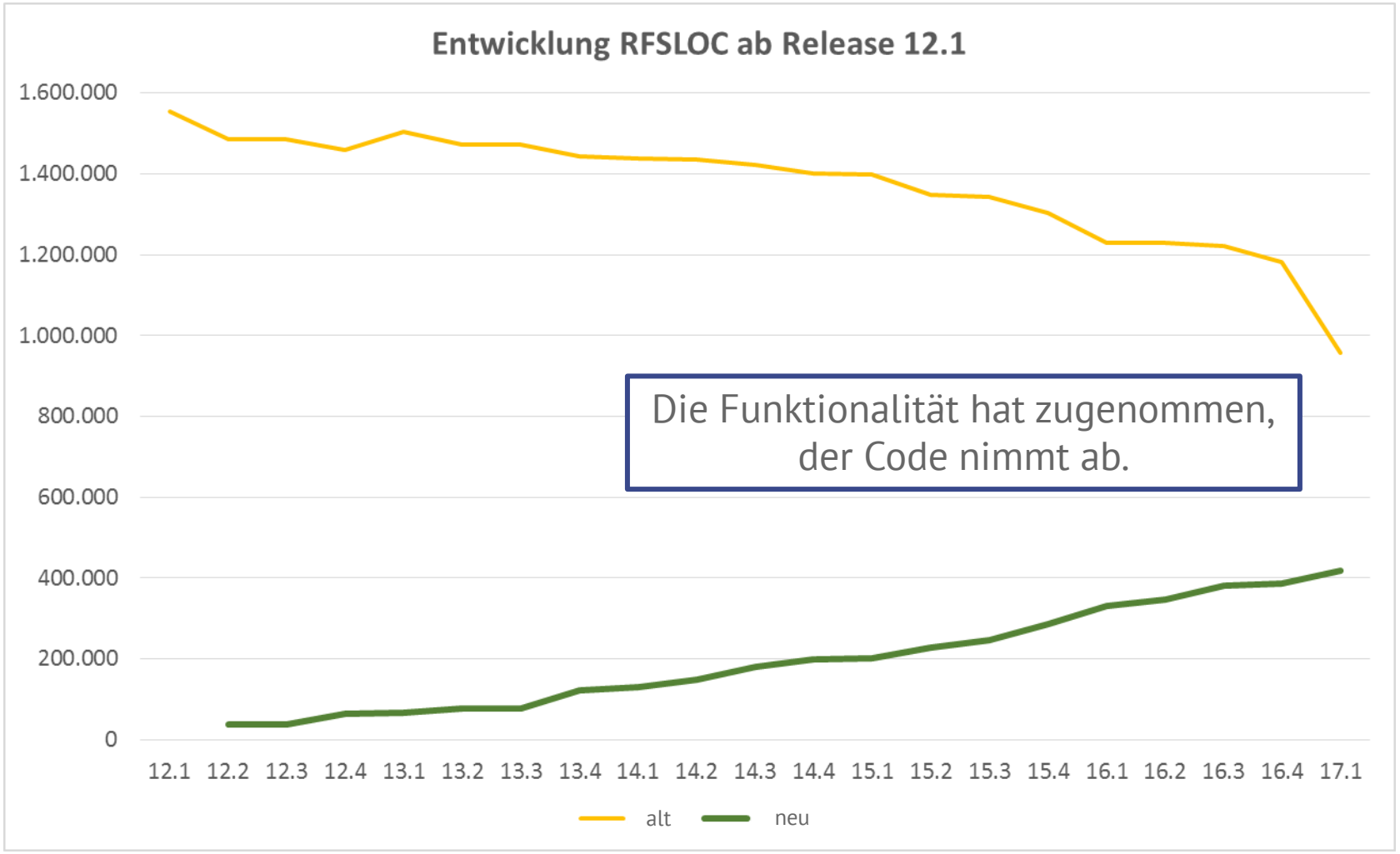
Modus „opportunistic“ → **Modus „active“**

Fachliche / gesetzliche Themen

Modernisierung

➔ weitere 2-3 Jahre zu erwarten.

Code-Wachstum und -Abnahme



Fazit

itestra
Software Productivity

- Fähigkeit zur Digitalisierung („fit für die Zukunft“) hängt ab von
  - **Zeit** und **Kosten** und damit
  - **Budget**, **Kapazitäten** und **Qualität**.

- Das Alter der Technologie ist nur ein Einflussfaktor
- Werkzeuge und „*tech. debt*“ geben (bisher) keine gute Antwort
- Wichtig sind:
  - Standortbestimmung → Zielvorstellung → Roadmap
  - **Geduld und Ausdauer!**
- Blick in die Zukunft:
  - Milliarden Lines of Code in COBOL
  - Milliarden Lines of Code in „Legacy“ Java / C# / ABAP
  - Hoffentlich automatisierte Modernisierung? ☺

itestra
Software Productivity

VIELEN DANK!