

DELTA
software
technology

Legacy-Modernisierung

Mehr als nur Kosten sparen

Frankfurt, 10.03.2016

The Perfect Way to Better Software

Entstanden

- In ca. 30 Jahren
- Nach damaligen Paradigmen: Monolith
- Mit damaligen Programmiersprachen: COBOL, PL/I,...
- Seit dem immer wieder erweitert, umstrukturiert, angebaut,...
 - ❖ Neue Technologien hinzugefügt ohne die alten (vollständig) zu ersetzen

Ergebnis

- Unflexibel, unübersichtlich, monolithisch
 - Teuer
- ⇒ Wahrgenommen als „Klotz am Bein“

Eigentlich sind die Anwendungen

- (Passgenaue) Unterstützung der Geschäftsprozesse
- Know How und Entwicklungsarbeit vieler Jahrzehnte
- Groß, komplex, unternehmenskritisch

„Wegschmeißen und Neuschreiben“

- Hohes technisches und finanzielles Risiko
- Finanzieller Verlust: Investitionen vieler Jahre werden weggeworfen

=> Benötigt wird eine Modernisierung

Beispiele für Modernisierungsaufgaben

- Bereinigung
 - ❖ Überflüssigen und/oder toten Code entfernen
- Plattform-Migration
 - ❖ Hardware
 - ❖ Compiler
- Umstrukturierung, Refaktorisierung in Service-orientierte und flexible Architektur
- Ersetzen alter Technologien durch modernere, bzw. Konsolidierung der Technologien
 - ❖ Datenbanken
 - ❖ Sprachen
- Kombination aus verschiedenen Punkten

Unsere Automatisierungsthese



Peopeware

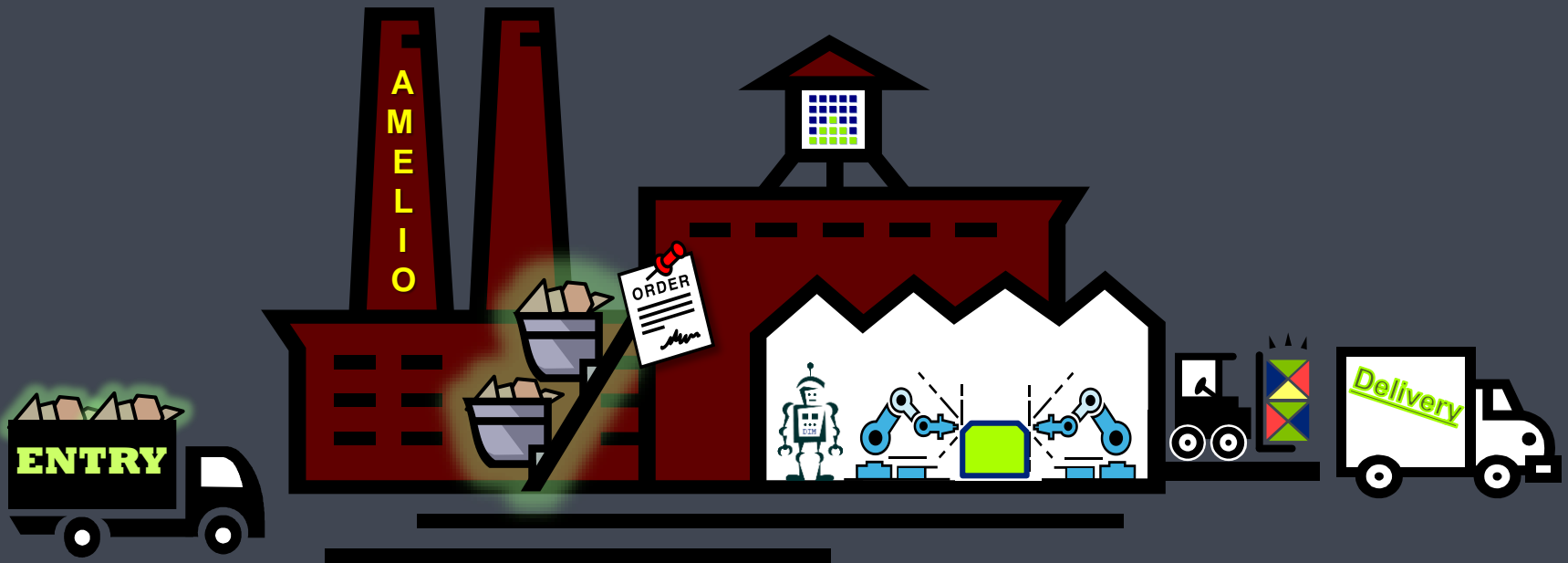
If you can tell the rules to somebody somewhere, you should be able to define them for a computer.

If you don't know the rules, you would better hesitate to do anything at all.

Hard- and Software



AMELIO: Modernisierungswerk



Vorteile eines Factory-Ansatzes

Passgenau/ Kundenspezifisch – kein „one-size-fits-all“

Modellgetrieben und regelbasiert

- Hohe Abstraktion
- Wiederholbar, reproduzierbar, nachverfolgbar
- Absolut gleichförmig
 - ❖ Kein persönlicher Stil, keine Tagesform
- CleanRoom-Konzept
 - ❖ Vollständig automatisierte und kontrollierte Prozesse
 - ❖ Keine „Kontamination“ durch manuelle Eingriffe

Kein Footprint, kein Vendor-Lockin

Schrittweise Modernisieren

Besser schrittweise, als „Big Bang“

- Z.B.: Plattformwechsel → Code-Bereinigung → Refaktorisierung → partielles Neuschreiben



Schrittweise Modernisieren

Vorteil

- Risikominimierung durch überschaubare Schritte
- Keine Vermischung von Impacts
- Strategiewechsel nach jedem Schritt möglich
 - ❖ Bei Automation: Sogar innerhalb eines Schritts möglich
 - ❖ Extrem später „Point of no return“
- Nach jedem Schritt kann das Ergebnis produktiv genutzt werden
 - ❖ Jeder Schritt ist ein Gewinn
 - ❖ Früherer „Return on Invest“

Plattformwechsel und Architekturtransformation

- Transformation einer Unisys-Anwendung nach Windows
 - ❖ Im 24/7-Betrieb
- Austausch von DBMS und Benutzerschnittstelle
- Architekturtransformation: Einführung von Services
 - ❖ Automatische Erzeugung von Datenservices
- Schrittweise
 - ❖ Schritt 1: Einführung von Services
 - ❖ Schritt 2: Plattform-Migration
 - ❖ Schritt 3: Austausch des DBMS

Zahlen

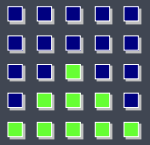
- 11.000 Module, 14 Millionen LOC
- 98,66% Automationsgrad
- 1,3 Millionen Änderungen
- 49 Fehler (0,00385%)

Modernisierung

- ⇒ Erzeugt passgenaue und flexible Unterstützung der Unternehmensprozesse
- ⇒ Hilft die Jahrzehnte getätigten Investitionen effizient weiter zu verwenden
- ⇒ Reduziert das Risiko eines „Wir schreiben alles neu“-Ansatzes

Das Plus

- Bereinigung und Refaktorisierung verbessert
 - ❖ Les- und Wartbarkeit
 - ❖ Verständnis der Anwendung
 - ❖ Anpassbarkeit an neue Anforderungen
- Konsolidierung von Technologien vereinfacht das Handling
- Architekturtransformation, von Monolith zu Service-Orientierung
 - ❖ bringt Flexibilität
 - ❖ ermöglicht Eingliederung in agilen Prozess
- Schrittweises Neuschreiben (falls noch notwendig/gewünscht) von überschaubaren Einheiten wird möglich



DELTA
software
technology



www.delta-software.com/amelio