BSSE System and Software Engineering

Automatische Software Produktion
Was ist realistisch?

MicroConsult Praxisforum „Embedded Quality“
Neue Dimensionen der Qualitätssicherung

Version 1

Zürich 10.10.2002

Dr. Rainer Gerlich
Auf dem Ruhbühl 181
88090 Immenstaad
Germany

Tel. +49/7545/91.12.58
Fax +49/7545/91.12.40
Mobil +49/171/80.20.659
e-mail gerlich@t-online.de

Inhalt

- Warum Automation?
- Anwendungsfälle
- Strategie und Prinzip von ASaP

  ASaP = Automatische Software Produktion und Test
- Vorteile
- Ergebnisse
- Einstieg in ASaP

ASaP - Automation in der Softwareentwicklung

- die Vorteile von ASaP
  - die Komplexität für den Entwickler sinkt
  - der manuelle Aufwand wird reduziert
  - Aufwand, Entwicklungszeit und Risiko werden minimiert
  - gleichbleibend hohe Qualität wird garantiert
- der ASaP Ansatz
  - vollständige Automatisierung des Entwicklungszyklus
    automatische Umsetzung der Definition in ausführbares System
  - Skalierbarkeit - Abdeckung verschiedenster Systemanforderungen
  - automatische Verifizierung und Validierung, Testen
  - Definition nur durch Namen und Zahlen, Typen oder Templates
  - Identifikation des Rationalisierungspotenzials
  - organisatorische Maßnahmen zur Begrenzung des Aufwandes

Warum Bedarf für Innovation?

- Anforderungen aus der Raumfahrt
  - Zuverläsigkeit

    Vorgang nicht wiederholbar, z.B. Landung auf Planeten

    „fail-operational“
  - Machbarkeit
    - Architekturevaluierung: Zentralrechner vs. verteiltes System
    - Energiebedarf und Masse vs. CPU- und Bus-Last
    - Kosten und Komplexität
- ähnliche Problemstellungen außerhalb Raumfahrt
  - Entwicklung von isolierten Komponenten zum System
  - steigende Komplexität und Intelligenz bei hoher Zuverlässigkeit
  - z.B. Automobiltechnik, Medizintechnik, Robotik, Biotechnik
  - Auswirkungen von Softwarefehlern nehmen zu
  - Synergie durch Know-how aus der Raumfahrt

Prinzipielle Anwendungsfälle

- neues System: Automation von Produktion und V&V
  - vollständige, automatische Generierung aus Systeminformation
  - ggf. Integration von manuell entwickeltem oder anderem Code
  - automatische Verifikation und Validierung, Testautomation (V&V)
- existierendes System: Testautomation und V&V

  Wartung, Erweiterung, höhere Qualität
  - automatisches Testen:

    vorhandene Information auswerten und automatisch in Testfälle umsetzen
  - automatische Verifikation und Validierung

    automatische Instrumentierung für Überwachung und Berichtserstellung

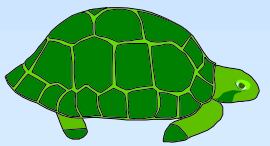
Warum Automation der Produktion?

- Demonstration der Machbarkeit bei wenig Aufwand
- durch Simulation?
  - Vereinfachung → Reduktion von Kosten und Zeit ☺ ✓
  - genügend repräsentativ? ☹ **Nein!**
  - keine Lösung bei ereignisgesteuerten Systemen im Gegensatz zu kontinuierlichen Systemen
- durch „Automation“!
  - schnelles Feedback
  - geringe Kosten
  - reproduzierbares Produktionsverfahren
  - reproduzierbare Qualität

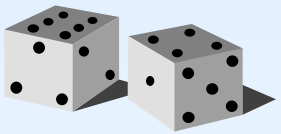
Warum Testautomation?

- Beispiel:
  - Verlust von Marktanteilen wegen Qualitätsproblemen auf gesättigtem Mark, finanzieller Verlust
    - Evolution der Produkte
    - steigende Komplexität
    - stark wachsende Anzahl der Testfälle
  - Status
    - 10 .. 20 % Testabdeckung
    - mit bisher ca. 1 .. 3 MJ
    - → 5 .. 30 MJ = 700 .. 4,000 kEuro noch notwendig

      unmöglich, durch manuelles Testen Schritt mit der Entwicklung zu halten



  - trotzdem Argumentation gegen Testautomation
    - wir müssen nur anfangen, richtig zu arbeiten ...
    - brauchen wir wirklich so viel Qualität?

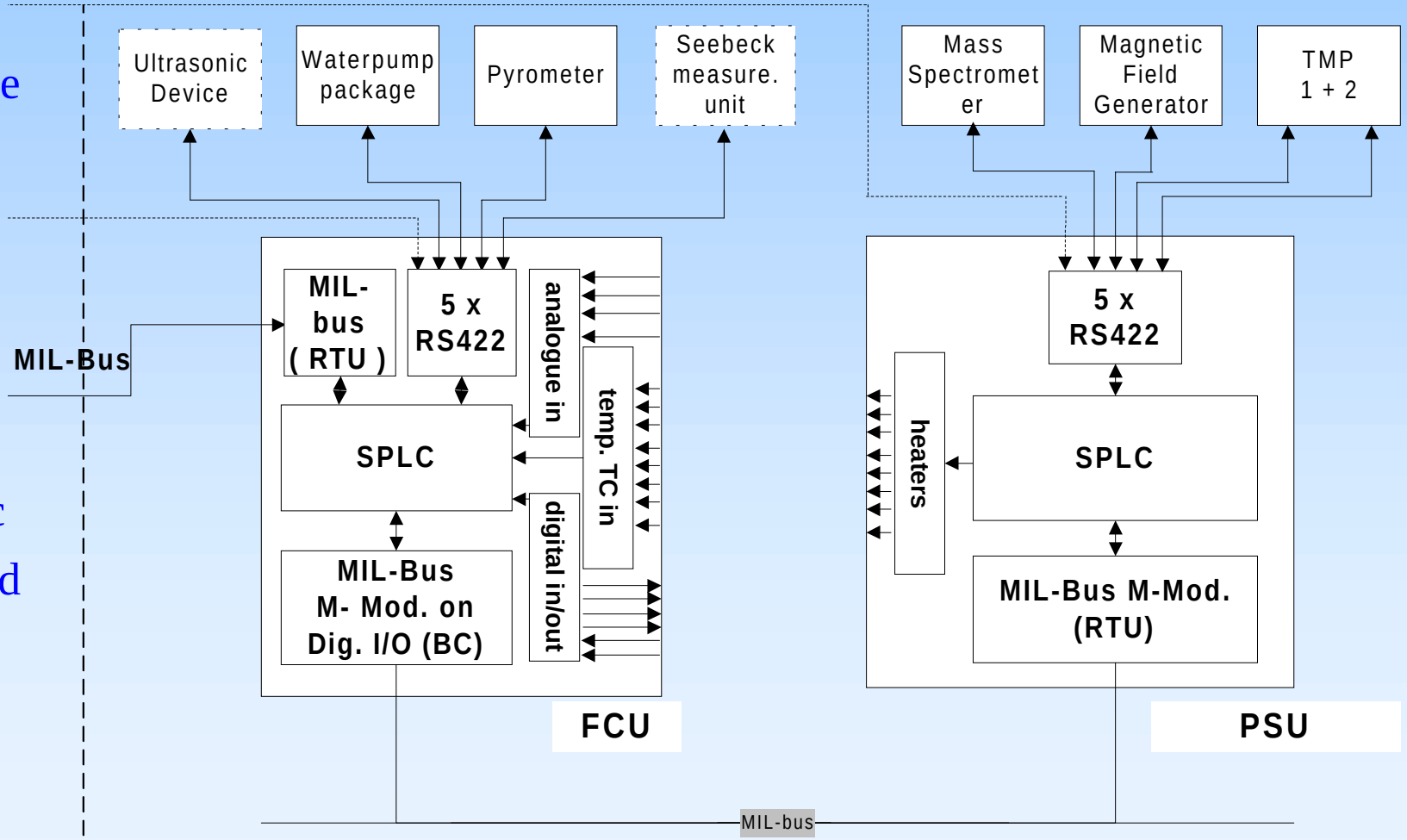


BSSE System and Software Engineering

Beispiel: Verteiltes Echtzeitsystem (events + synchron)

Peripherie

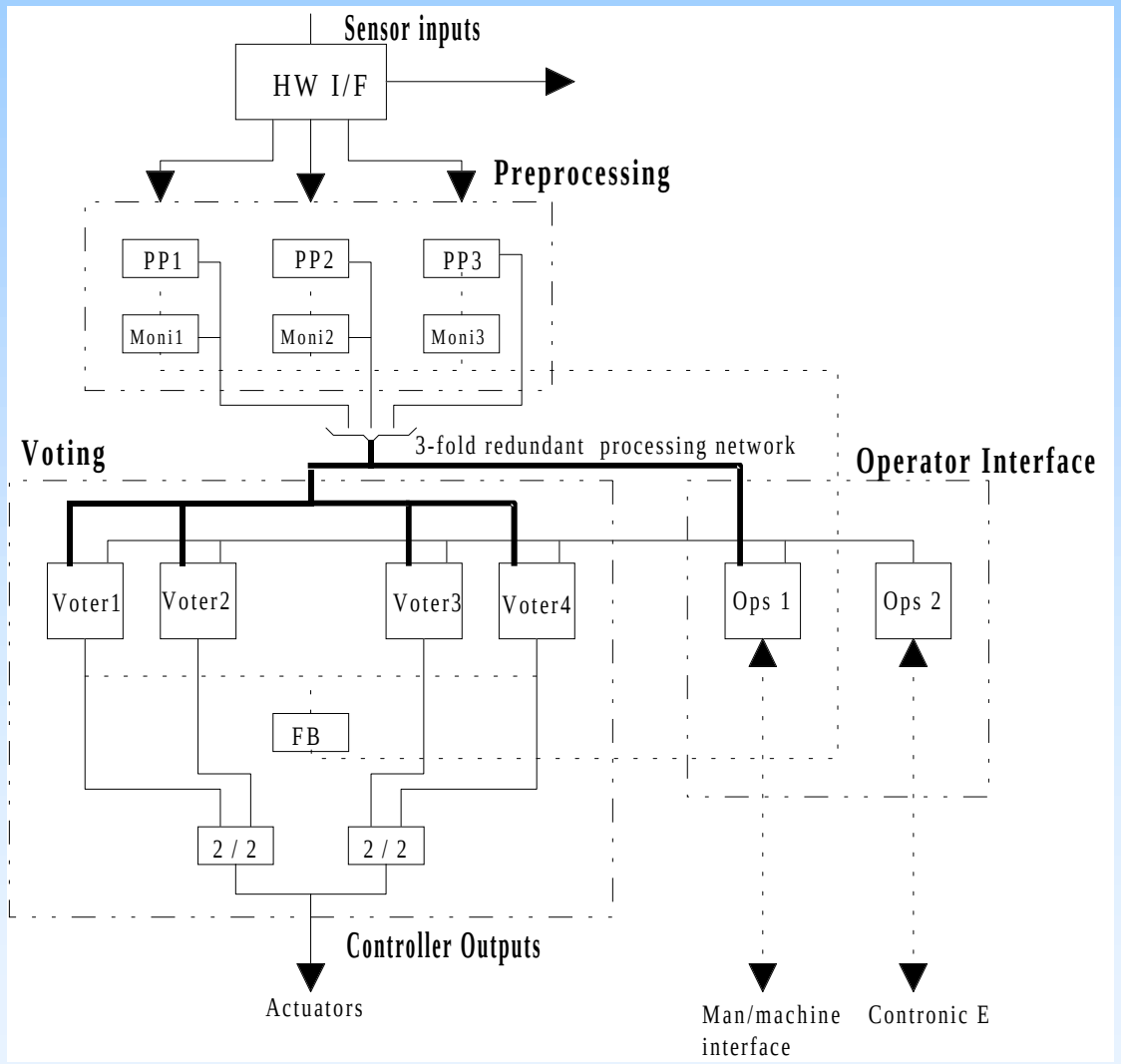
2 x Sparc embedded



Beispiel: Verteiltes synchrones Kontrollsystem

Überwachung

Redundanz



zyklische Datenakquisition

Vorverarbeitung

Filterung

Voting

Potentielle Anwendungsbereiche
Beispiel Automobilbau

- Bedarf: steigende Komplexität und Zahl der Komponenten
  - Systemhierarchie (System und Untersysteme), Koordinierung
  - Bedarf an zuverlässiger, benutzerfreundlicher Software
  - fail-stop (Servo) $\rightarrow$ fail-safe $\rightarrow$ fail-operational (drive-by-wire)
  - Trade-Off für Systemarchitektur (z.B. zentral - dezentral)
  - Untersuchung und Behandlung von Fehlersituationen
- Fahrzeug
  - Aggregate, Datenerfassung, -verarbeitung, -überwachung, Regelung
- Fahrerunterstützung
  - Anzeigesysteme, Leitsysteme, Komfort
- Fahrzeugfertigung
  - Roboter
    Programmverifikation, automatischer Vergleich mit CAD-Soll-Daten

Automatische Softwareproduktion vs. Automatische Codegenerierung

- Prinzipielles Problem
  - uns fehlen Sensoren zur Identifikation von Softwarefehlern
  - viel leichter, Systembeschreibung oder viel Code zu erstellen ...
    ... als zu verifizieren und zu validieren
- ASaP Lösung

  durch Automation
  - Synergie zwischen Generierung, Verifizierung und Validierung
  - korrekten Code und lauffähiges System garantieren
  - schnelles Feedback vom System durch Visualisierung
  - automatisch und früh Probleme identifizieren
  - manuellen Aufwand begrenzen
- Automatische Codegenerierung
  - deckt nur einen Aspekt von sehr vielen durch Automation ab

Was ist Automation?

Automation = Erfahrung + Organisation

Automation = korrekte Transformation „Vorgabe → Ergebnis“

Automation = Umsetzung von Regeln in Produktionsprozess

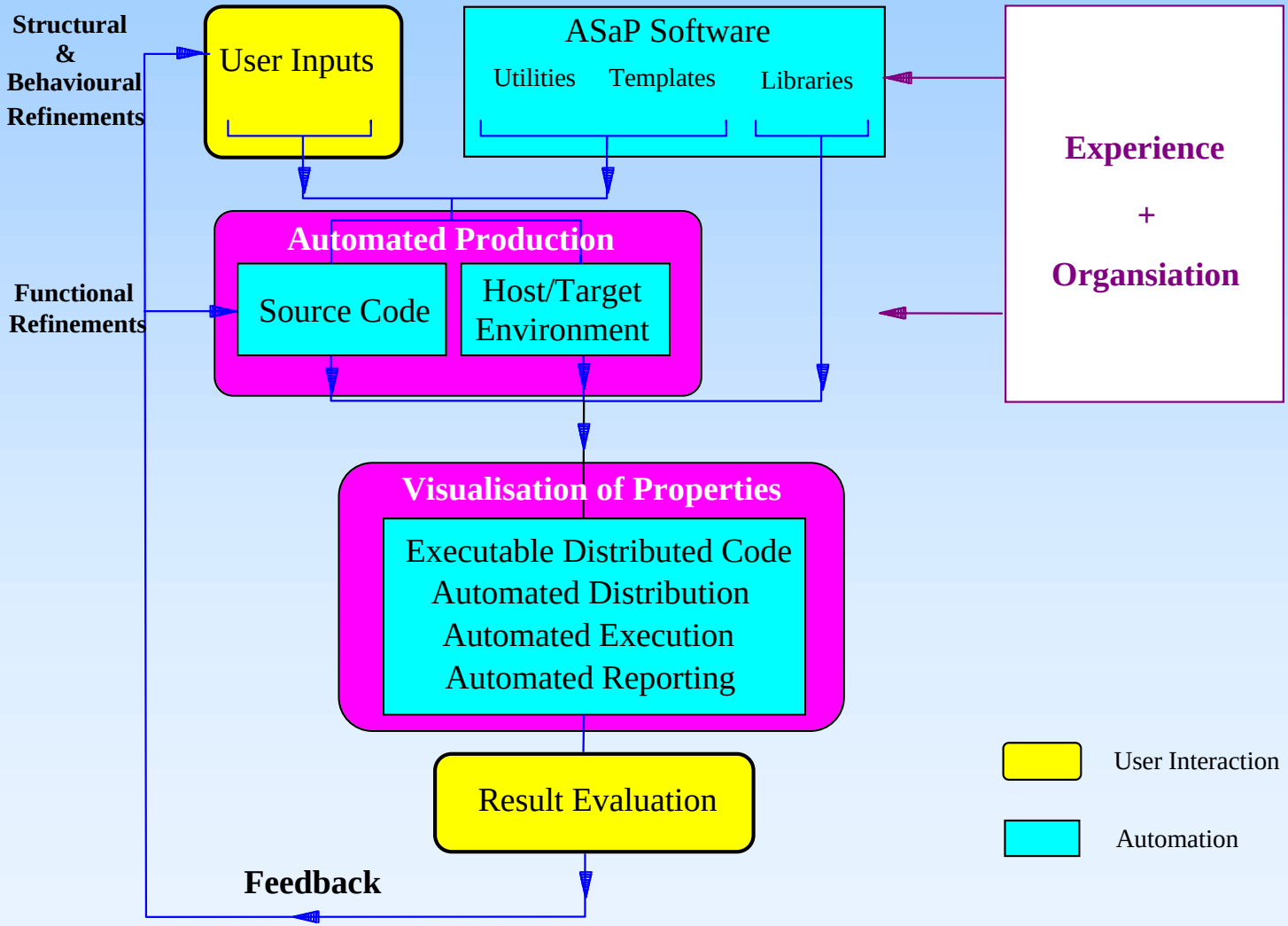
direkt Erfahrung umsetzen, ...

... statt Einhaltung von Standards nachträglich zu kontrollieren versuchen

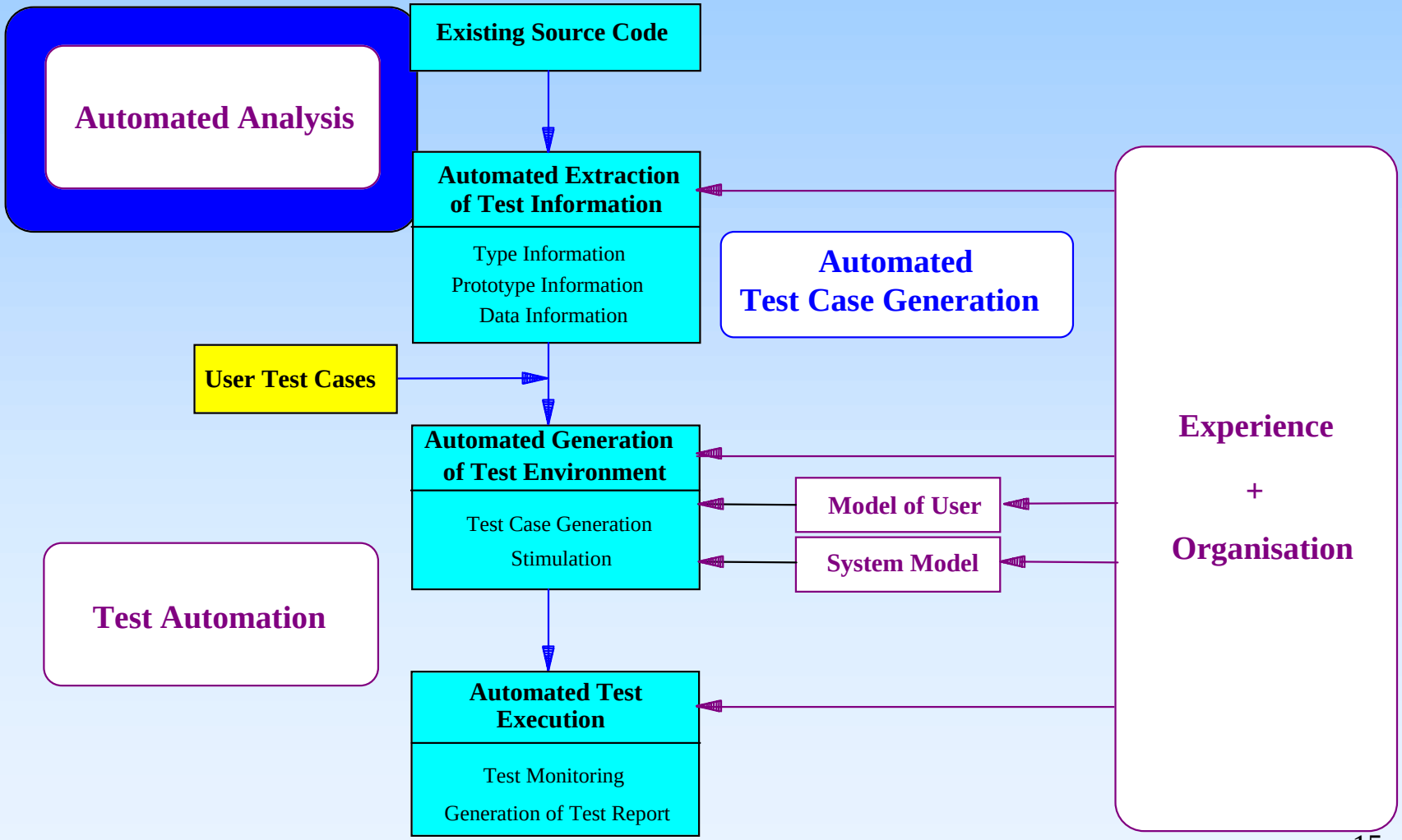
Vorteile der Automation

- Konfigurieren statt Implementieren
- mehr Effizienz durch Spezialisierung
  - Spezialisierung auf Teilbereich ohne Beschränkung der Allgemeinheit
    z.B. verteilte und/oder Echtzeitsysteme (großer Teilbereich)
  - Teilbereich $\rightarrow$ Konkretisierung $\rightarrow$ Produktions- / Prüfregeln
  - diametral zu gegenwärtigen Trend!
- mehr und frühzeitigere Prüfungen bei weniger Aufwand
- wesentlich mehr Gestaltungsfreiheit bei weniger Risiko

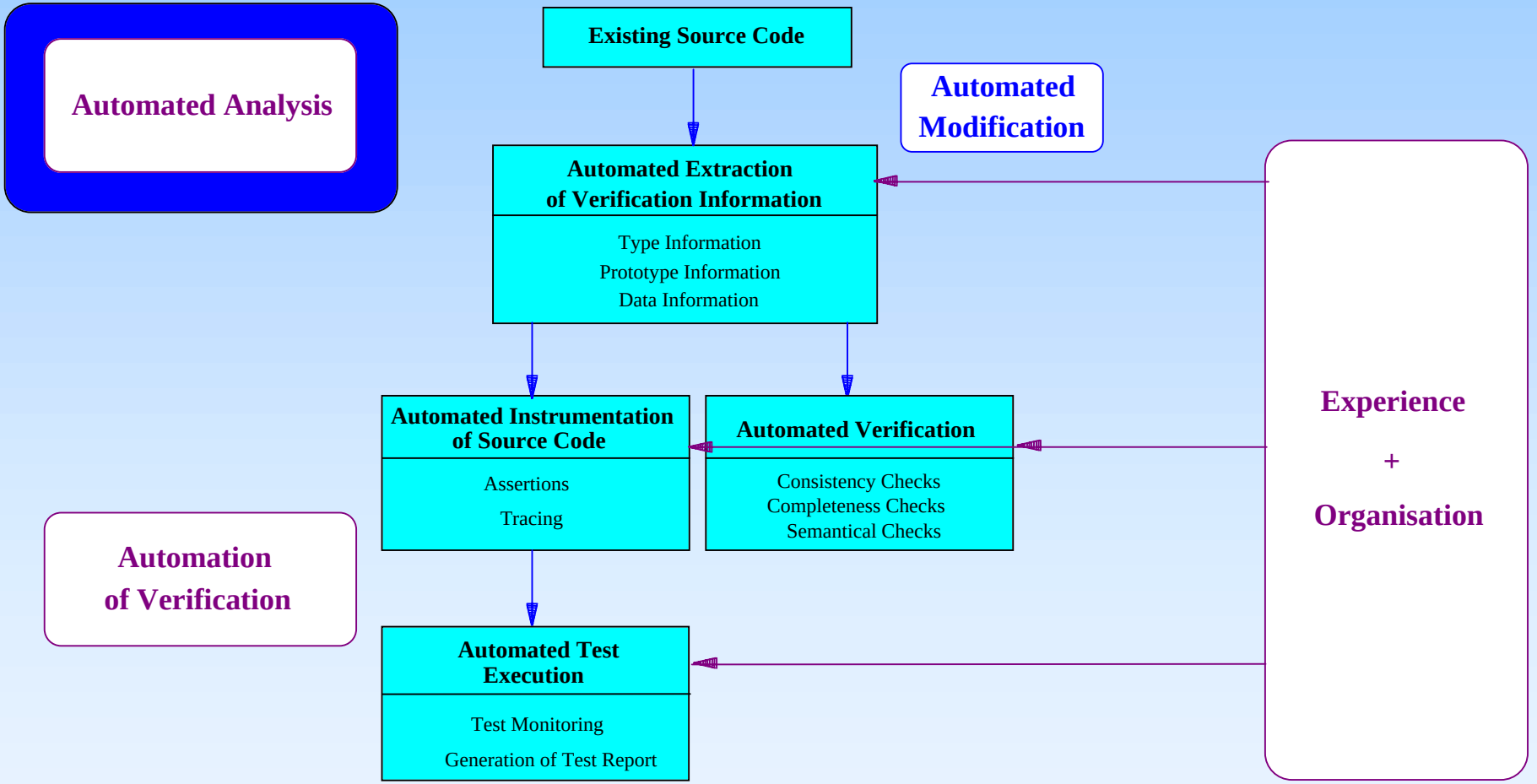
System-Entwicklungs-Zyklus von ASaP



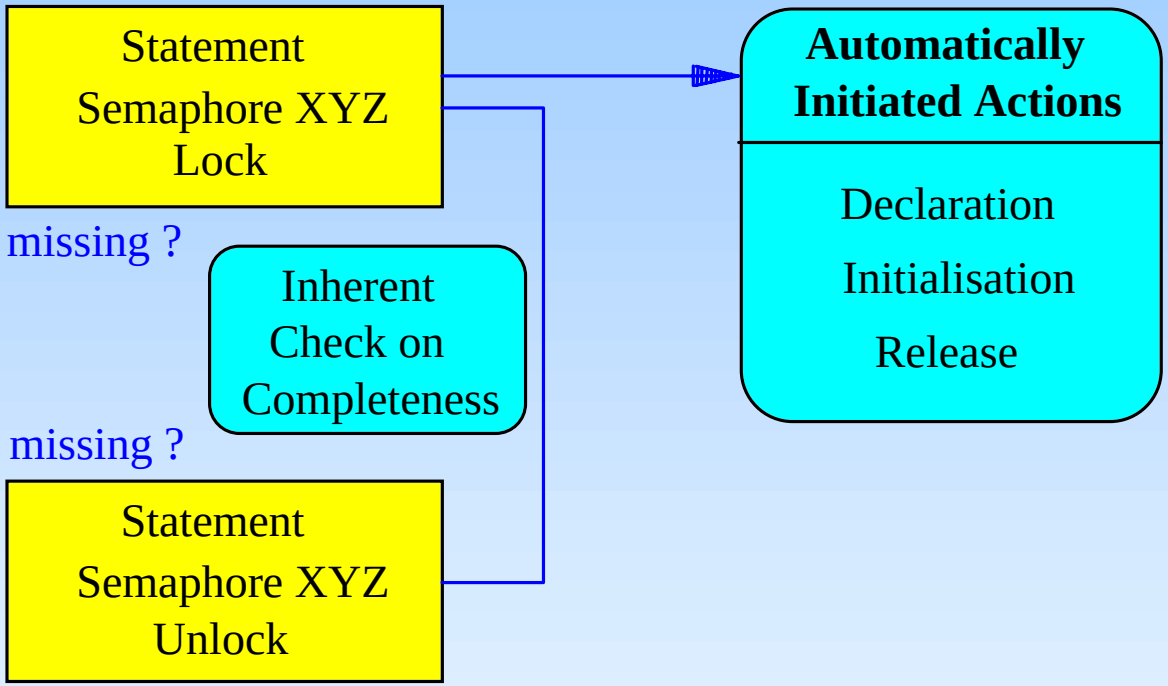
Automatisches Testen



Automatische Verifikation



Beispiel 1: Synergie durch Spezialisierung

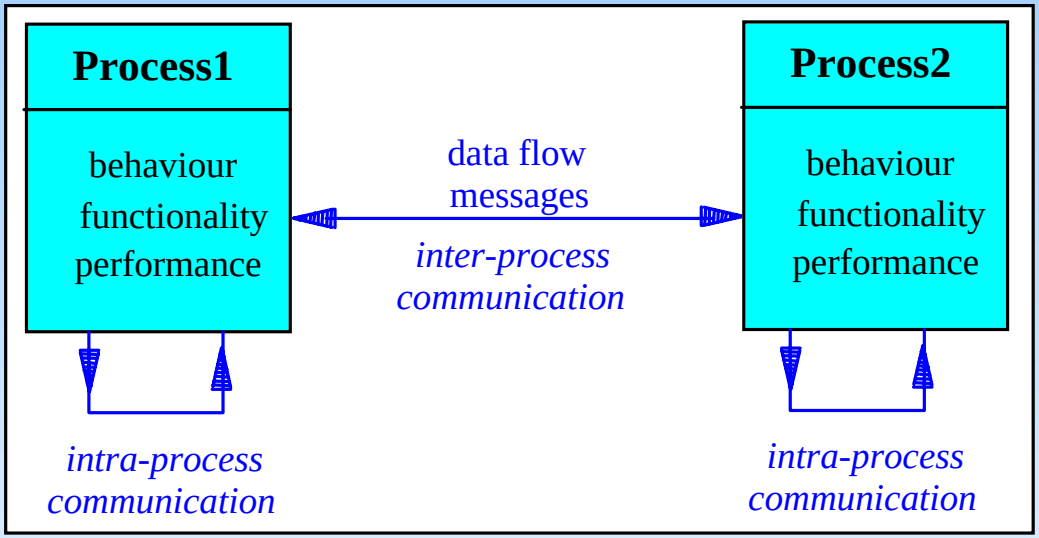


Semantic: each pair only once

Occurence of a statement: **check on complementary statement**
automated generation of complementary code

Beispiel 2: Konfigurationsoptionen

Principal Configuration Options of a Distributed / Real-Time System



Intra-Process Activities
Finite State Machines
synchronous processing
asynchronous / sporadic processing
time-out
exception handling

Distribution
Mapping Process - Processor
Communication Channels
Topology
Fault Tolerance
OS

Vorteile im Detail

- geringeres Entwicklungsrisiko und höhere Planungssicherheit
  - schnelles Feedback bzgl. kritischer Größen (Performance, ...)
  - mehr Flexibilität für Änderungen
- Sicherung der Unternehmenserfahrung
  - kontinuierliche Optimierung des Produktionsprozesses
  - Unabhängigkeit von „Tagesform“
- höhere Qualität
  - Produktion nach festen, erprobten Regeln
  - Qualität quantitativ nachweisen
  - Abdeckung von Fehlereinspeisung, Stress Testing, Überlast
- automatische Verifikation und vollständigerer Test
  - automatische Extraktion der notwendigen Information
  - automatischer Aufbau der Testumgebung
- Visualisieren durch Automatisieren
  - automatisch Bedingungen prüfen und Berichte erstellen
  - „Antworten erhalten, ohne fragen zu müssen“

Weniger Aufwand, weniger Fehler ...

- Ingenieure nur in den Definitionsprozess einbinden
  - Umsetzen von Ideen
  - nur Ergebnisse validieren und Ansatz verbessern / korrigieren
  - z.B. Kommunikation, Verhalten, Topologie nur durch Namen und Zahlen
- Abhängigkeit vom Definitionsumfang reduzieren
  - linear statt höhere Potenzen
  - Beispiel Netzwerk: nur Knoten (**o(n)**) statt Verbindungen (**o(n^2)**)
- Aufwand begrenzen
  - endlicher (manueller) Aufwand für unendliche Menge von Möglichkeiten
  - u.a. diametral zum Klassenkonzept $\rightarrow$
    für jeden neuen Datentyp fällt manueller Änderungsaufwand an
  - ASaP
    einmaliger Aufwand nur für Basistypen pro Methode / Funktionalität
    z.B. char, short, int, long, float, double

Quantifizierbarer Fortschritt bei Automation

- Quantifizierung des Erfolgs möglich
  - Produktivität um x % erhöht
  - Entwicklungszeit um y % verkürzt
  - Fehlerrate um z % verringert

    durch sofortige Verfügbarkeit des Systems und der Berichte
- **Garantie** für
  - Korrektheit des Systems durch Überprüfung der Benutzereingaben
    - aus Minimum an Information
    - aber alles, was für automatische Transformation erforderlich
  - Einsparung von Kosten und Entwicklungszeit
  - (Verringerung der) Fehlerrate

Durch Automation erzielte Produktivität und Qualität

- Produktivität
  - in 30 Minuten das Äquivalent von ca. 5 Mann-Jahren (MJ)
    - verteiltes Echtzeitsystem
    - in 30 Minuten das Äquivalent von ca. 1 MJ
    - verteilte synchronisierte Datenbank
  - in 1 Minute das Äquivalent von ca. 2 MJ
    - Operationen zu Datentypen, Interfaces etc.
- Fehlerrate
  - verteiltes Echtzeitsystem 2 = 2.5 .. 1 * 10^{-5}
  - verteilte Datenbank 0
- Literatur
  - üblich: 10^{-2} sehr gut: $<10^{-3}$

Grenzen und Integrationsfähigkeit

- Aktuelle Grenzen / Verfügbarkeit
  - am Anfang einer Entwicklung in Richtung Automation
  - Identifikation weiterer Gebiete durch Diskussion mit Kunden
  - Identifikation von organisatorischen Maßnahmen
- Prinzipielle Grenzen
  - Komplexität der Anwendung im Vergleich zur Automatisierungserfahrung
  - aber: kontinuierlich wachsende Erfahrung
  - Grenzen werden immer weiter hinausgeschoben
- Komplementarität
  - Integration existierender Automatisierungsansätze
    - ✡ z.B. für kontinuierliche Anwendungen MathLab, Scade
  - Schnittstellen für (automatische) Integration

Was behindert die Einführung der Automation?

- Konservative Ausbildung und Prägung
  - Skript SS 2002 über Status der Softwareentwicklung: künstlerischer, kreativer Prozeß, Handarbeit im Vordergrund
- Haltung der Entwickler
  - suchen Herausforderung in Implementierung und nicht in Gestaltung
- mangelnde Innovationsfähigkeit
  - „das ist bei uns ganz anders“
  - „das geht doch nicht“
  - „jetzt nicht, wir sind in Projekten unter Druck“
- inflexible Organisation und fehlendes Benchmarking
  - „unsere Standards sehen das nicht vor“
  - auf „manuelle“ Entwicklung ausgerichtete Standards
    Berichte und Reviews für Phasen, die bei Automation entfallen
    wer fordert Bericht / Review für eine Compilerphase an?

Ziele setzen!

- Beispiel SMD (Surface Mounted Devices)
  - Fernsehapparat 1960
    Preis ca.100 Mh, wenig Funktionalität, hohe Ausfallrate
  - Fernsehapparat 2000
    Preis ca. 20 Mh, hohe Funktionalität, sehr geringe Ausfallrate
- Verbesserung durch neue Technologie
  - durch Automation der Fertigungstechnik erheblicher Sprung in Funktionalität, Produktivität und Qualität (↗) und Kosten (↘)
- wichtig: Ziele setzen und realisieren
  - vollständige Änderung des Fertigungsprozesses und der Werkzeuge
  - drastische Änderung der Form der Bauelemente
  - firmenübergreifende Standardisierung
  - Software: interne Adaption vollkommen ausreichend

Einstieg in die Automation

- Mitwirkung des Kunden
  - Erfahrung mit Beschreibungselementen des Anwendungsbereiches
  - ggf. Mitarbeit bei Identifikation des Automatisierungsansatzes
- Einsatz für Systemgenerierung
  - Idee für Realisierung des gewünschten Systems
- Einsatz der Automatisierung für Test und Verifikation
  - Verfügbarkeit von Quellcode und/oder Dokumentation
- Empfehlung
  - gemeinsame Startphase in einem Projekt
  - schlüsselfertig: erste Iteration auf Anwenderplattform zum Starten
- Erfahrung
  - nach ca. 1 Monat vollkommen eigenständiges Arbeiten

Wie ist Automatisierung bei mir möglich?

- Analyse der „Herstellungsabläufe“ von Software
  - vergleichbar REFA-Ansatz
  - für ein möglichst breites Anwendungsgebiet wie
    Echtzeitsysteme, verteilte Systeme,
    Client-Server Anwendungen
    GUIs, Datenbanken, Prozess technische Systeme, ...

- Festlegen der optimalen Anwenderschnittstelle
  - Identifikation der Systemparameter („Gestaltung“)
    z.B. Prozesse, Topologie, Informationsaustausch
  - Einbettung in vorhandene Entwicklungsumgebung
    vorhandene Werkzeuge, ...
- Optimierung der Abläufe
  - Anpassung von Standards an Automation
  - Kontrolle von Produktivität und Qualität durch kontinuierliches Benchmarking