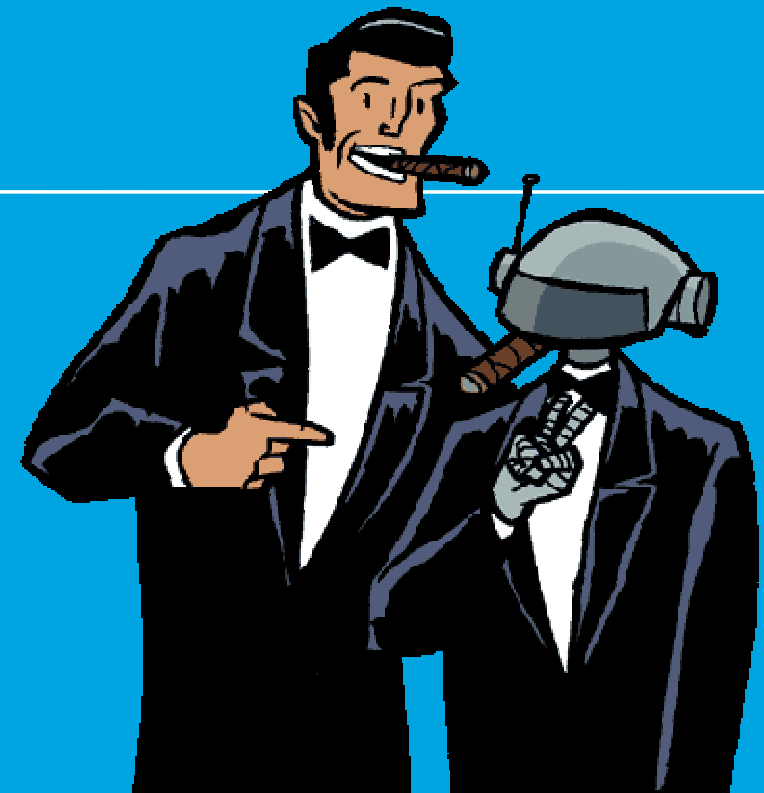


7+1 heiße Tipps für die Programmierung



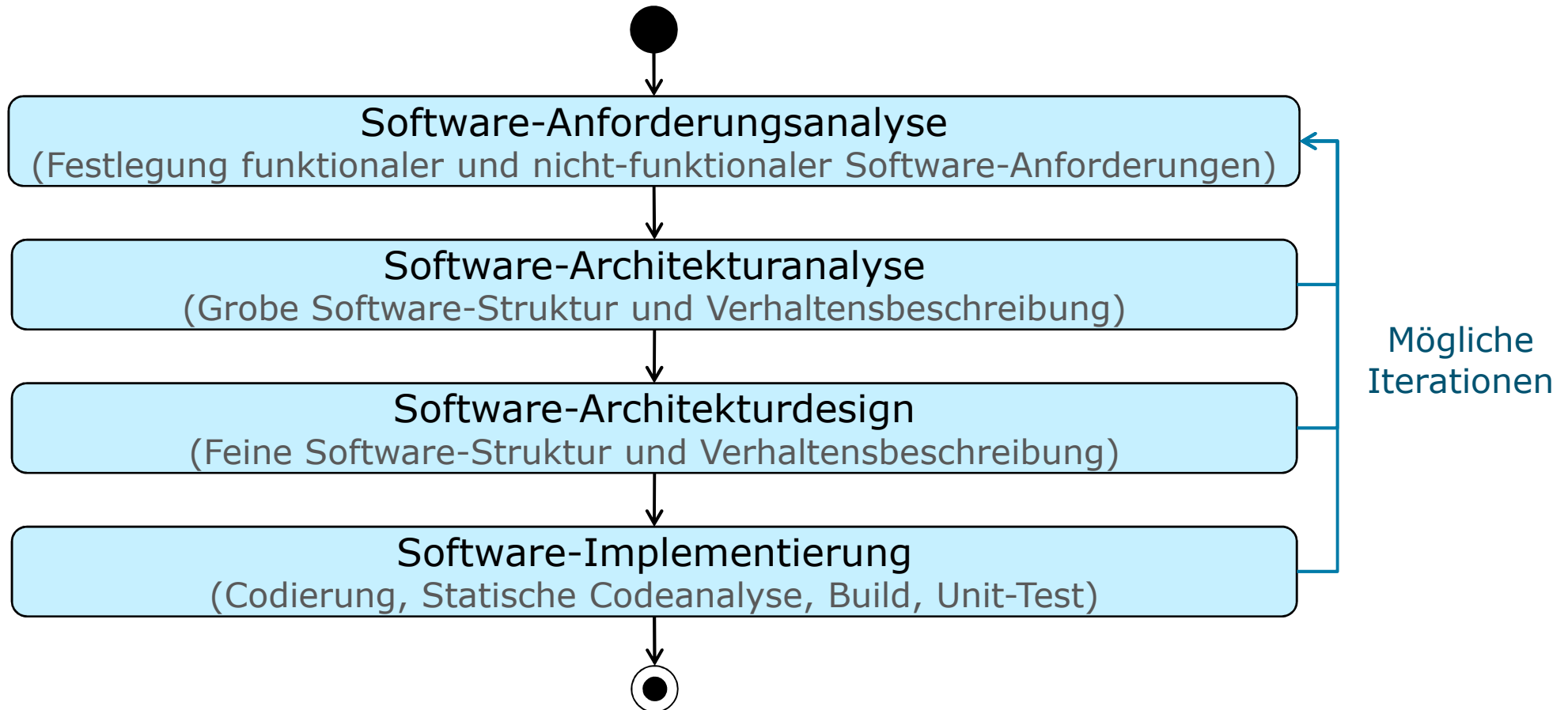
Warum muss ich Programmcode **nicht** kommentieren?

```
// function counts and shows
// min, max and count value
void op_1(void)
{
    // check count boundary
    if (a >= b && a < c)
    {
        //increment count value
        a = a + 1;
        printf("Min value    = %i", b);
        printf("Max value    = %i", c);
        printf("Count value = %i", a);
    }
    else
        //reset count value
        a = 0;
}
```

```
35: void Counter_count_show(void)
36: {
37:     if (count >= min && count < max)
38:     {
39:         count = count + 1;
40:         printf("Min value = %i ", min);
41:         printf("Max value = %i ", max);
42:         printf("Count value = %i\n", count);
43:     }
44:     else
45:         count = 0;
46: }
```

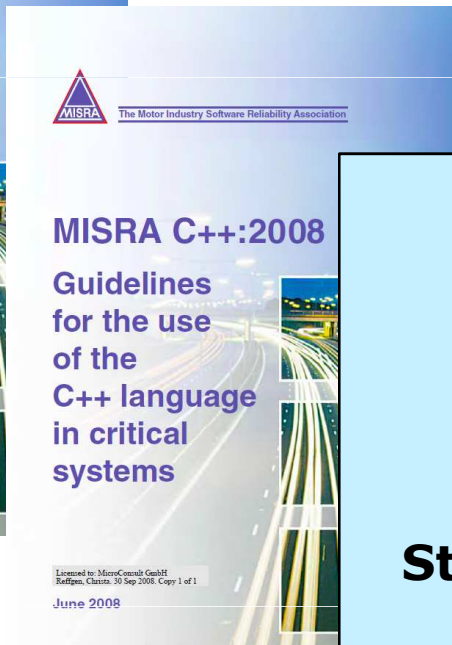
Wann immer Sie versucht sind Ihren **Programmcode** zu kommentieren, versuchen Sie ihn so umzucodieren, dass der **Kommentar überflüssig** ist!

Welche wichtigen Entwicklungsschritte muss ich vor der Codierung durchlaufen?

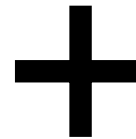


Entwickeln Sie auf Basis von Software-Anforderungen eine **Software-Architektur** und verfeinern Sie die Architekturelemente - erst dann codieren!

Wie kann ich die **Qualität** meines **Programmcodes** mit **geringem Aufwand verbessern**?



**Firmen-
interner
Coding-
Style-Guide**

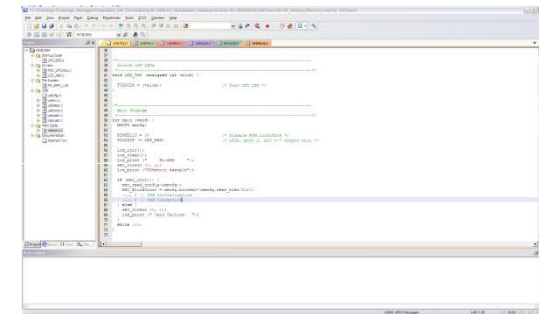
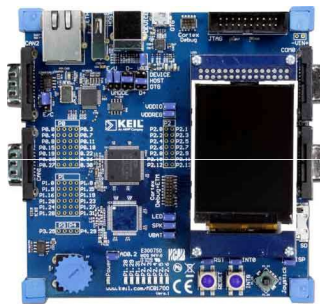


Statische Codeanalyse-Tools:

- PC-Lint (Gimpel Software)
- PC-Splint (Freeware)
- QA C/C++ (QA-Systems)

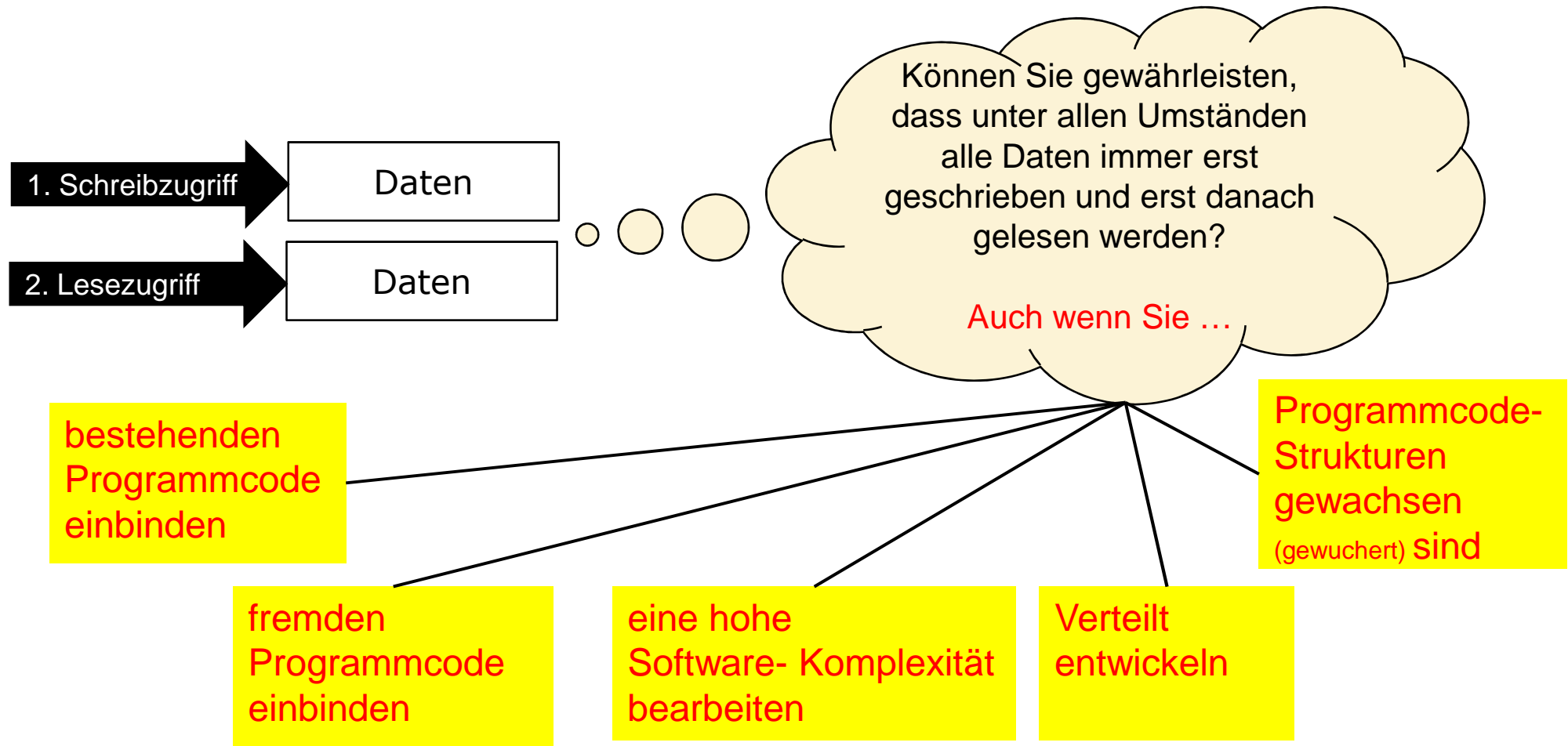
Codieren Sie nach Codier-Regeln und prüfen Sie die Einhaltung dieser automatisch durch eine Tool-gestützte statische Codeanalyse ab!

Wie kann ich **Software unabhängig von der konkreten Hardware, dem konkreten Betriebssystem und der konkreten Toolkette programmieren?**



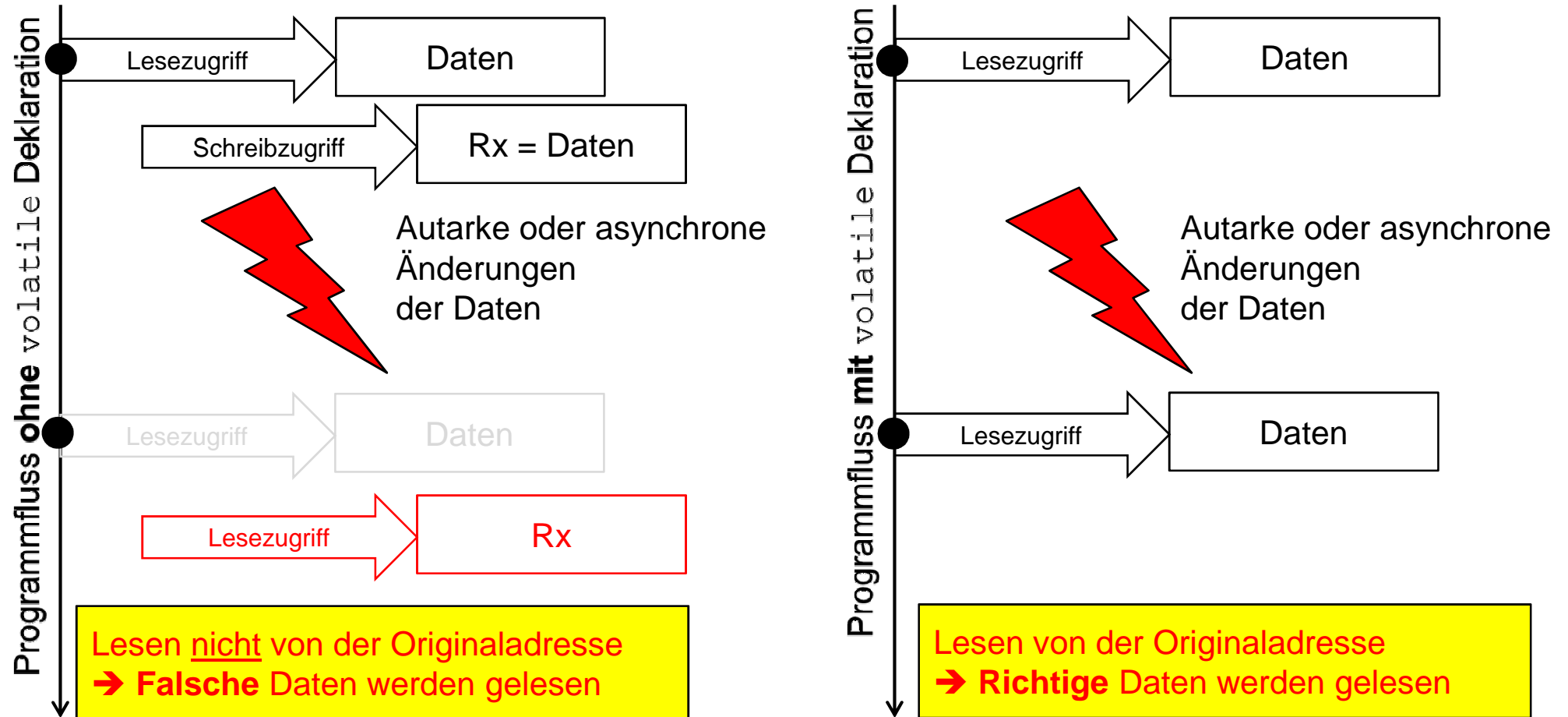
Entkoppeln Sie die **Applikation** von den konkreten Elementen durch kanalisierte Adaptionen!

Warum muss ich alle **Variablen** im **Speicher vorinitialisieren**?



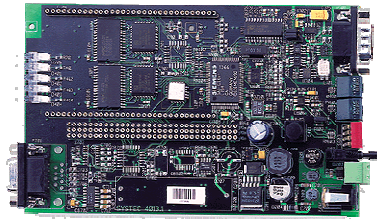
Das **Risiko**, insbesondere bei **sicherheitskritischer Software** ist sonst zu hoch - **LEBENSGEFAHR**, Sie würden grob-fahrlässig handeln!

Wozu benötige ich den Daten-Modifizierer „**volatile**“?

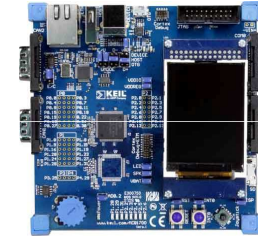


Deklariieren Sie Daten/Zeiger immer dann als **volatile**, wenn sie sich ohne Kenntnis des Compilers autark ändern können → **keine Compiler-Optimierung!**

Warum soll ich Hochsprache vor Assembler verwenden?



Target Wechsel



Applikation in
Target A Assembler

Komplett neu codieren

Applikation in
Target B Assembler

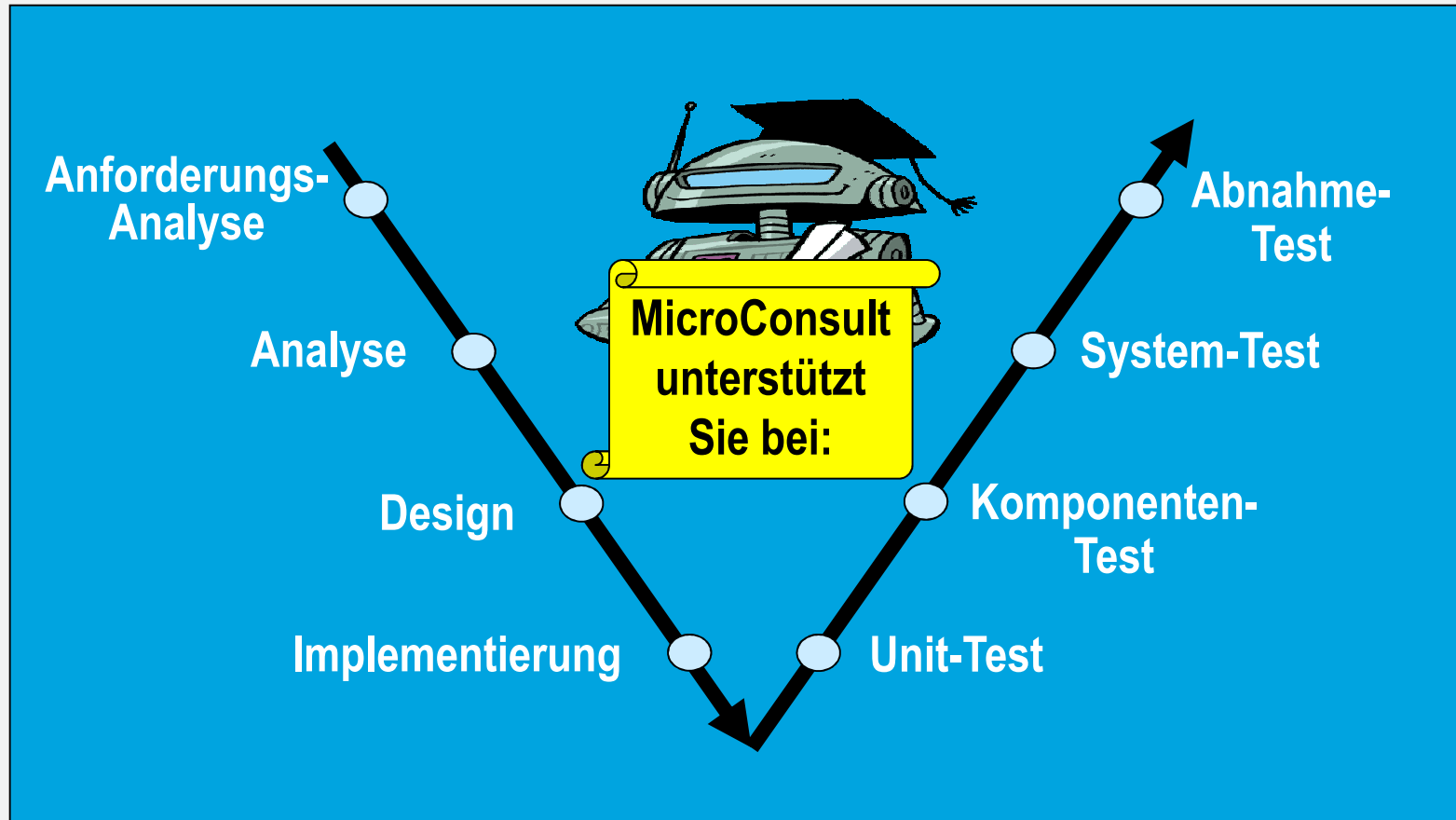
Applikation in
Hochsprache

Neue Toolkette für
Hochsprache und Target B

Applikation in gleicher
Hochsprache

**Hochsprache ist unabhängig von der Hardware,
Assembler ist Hardware-spezifisch und damit nicht portierbar!**

Beratung, Training, Workshops, Coaching, Projektarbeit



HW-/SW-Technologien, Tools, Methoden, Prozess, Team