



What does OOP mean for Embedded Realtime Systems?

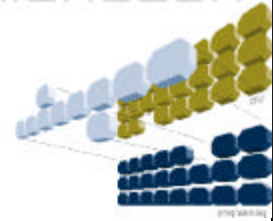
Quality by encapsulation and
use of the notation UML

Willert Software Tools GmbH
Walter van der Heiden &
Andreas Willert

Willert Software Tools ■

info@willert.de

+49 5722 9678 60



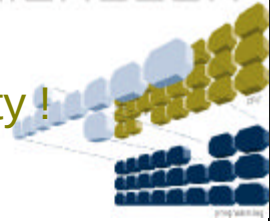
OOP of Embedded Systems with limited Resources based on UML?

Andreas Willert & Walter van der Heiden
Willert Software Tools GmbH

Willert Software Tools ■

info@willert.de

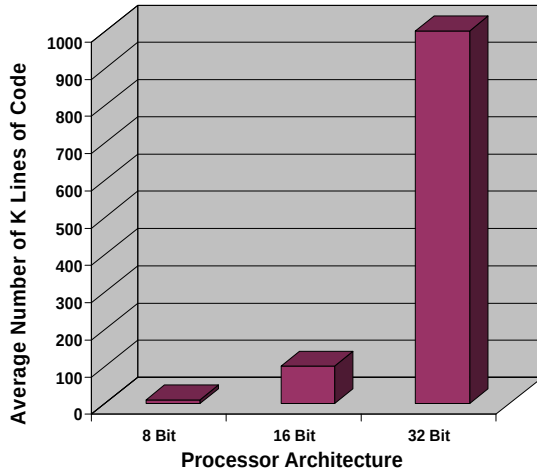
+49 5722 9678 60



Urban legends that limit productivity !

- Software is easy to change
- Software is developed for free
- Developer sitting in the same room have productive communication
- Responding to telephone calls and email just in time will increase productivity

then there's growth



- applications based on microcomputers show exponentially growth
- and are more and more complex to develop and debug!

Today's automotive applications

PWT UNIT



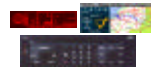
BODY GATEWAY



INSTRUMENT



TELEMATIC



MEMORY	256 KB	128 KB	184 KB	8 MB
LINES OF CODE	50.000	30.000	45.000	300.000
PRODUCTIVITY	6 LINES/DAY	10 LINES/DAY	6 LINES/DAY	10 LINES/DAY*
RESIDUAL DEFECT RATE @ END OF DEV	3000 PPM	2500 PPM	2000PPM	1000 PPM
CHANGING RATE	3 YEARS	2 YEARS	1 YEAR	< 1 YEAR
DEV. EFFORT	40 MAN-YEAR	12 MAN-YEAR	30 MAN-YEAR	200 MAN-YEAR
VALIDATION TIME	5 MONTHS	1 MONTH	2 MONTHS	2 MONTHS
TIME TO MARKET	24 MONTHS	18 MONTHS	12 MONTHS	< 12 MONTHS

Source: Wind River Systems, Inc. Proprietary

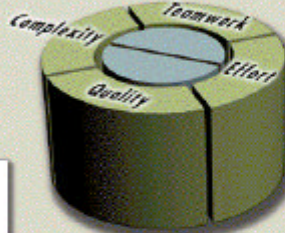
Willert Software Tools

info@willert.de

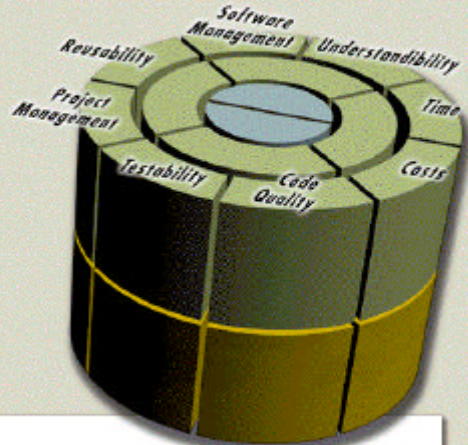
+49 5722 9678 60



64 KByte



128 KByte



256 KByte

Other Growth

- Number of employees in the projects
- Budget for Engineering measurements
- your salary

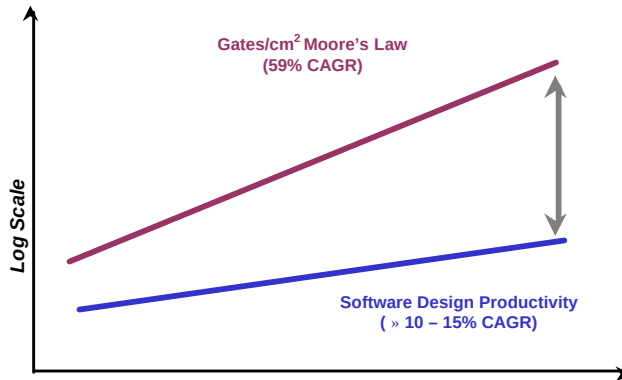


Willert Software Tools ■

info@willert.de

+49 5722 9678 60

Design productivity gap



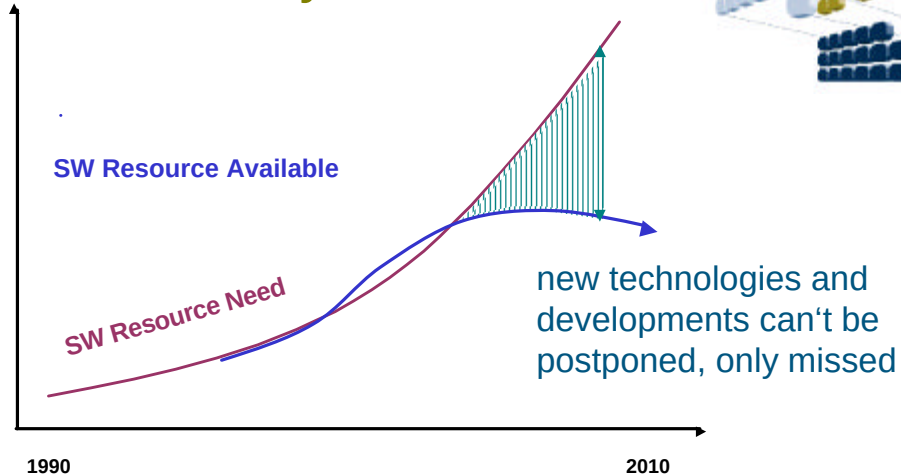
Source: Philips Semiconductor

Willert Software Tools ■

info@willert.de

+49 5722 9678 60

Availability of resources

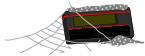


Source: Philips Semiconductor

Willert Software Tools

info@willert.de

+49 5722 9678 60

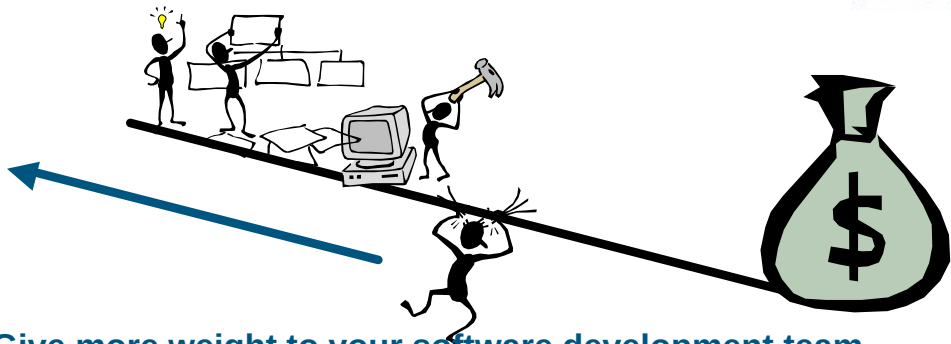


the results

- continue to add more development staff (37%) or rewrite modules (74.9%)
- reduce functionality (56%) or cancel the project (18%)
- deliver the project late (72.8%)

Or....enhance the process and make the available development staff more **PRODUCTIVE...**

The answer to the high cost country Europe (and UK)



**Give more weight to your software development team.
Get small and productive teams by use of state of the art
Methods and Tools.**

Willert Software Tools ■

info@willert.de

+49 5722 9678 60



Why new methods?

- Requirements on software development are increasing rapidly
- Efficiency in development process could not keep up with that increase
- Fill the gap with manpower is a costly matter in Europe

What do you need today?

Concepts, processes and tools for:

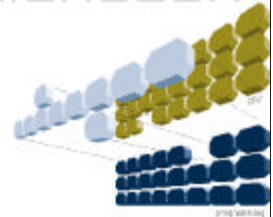
- Better communication in design teams
- Increase of changeability and reusability of software
- Early recognition and solving of Errors
- But also my be possibilities to better handling of stress by dedicated stress management?

Ergebnis einer Befragung von 18 UML Anwendern:



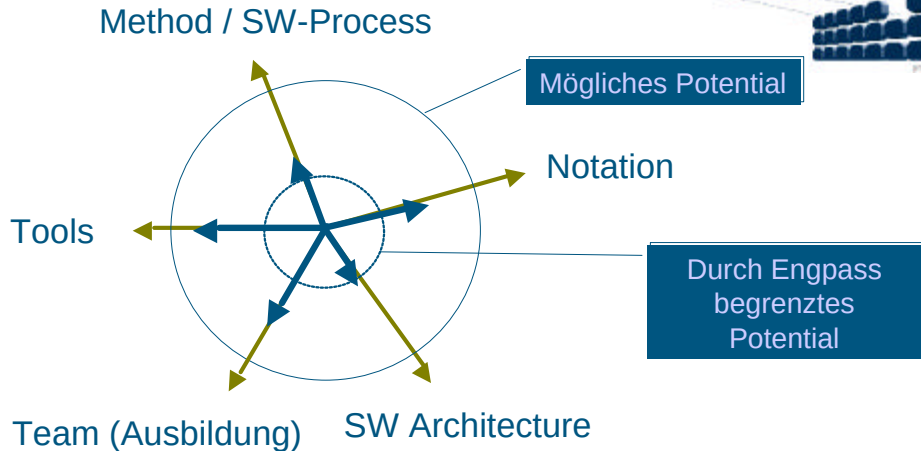
Worin sehen Sie die Hauptvorteile im Einsatz der UML?	Anzahl der Antworten
Erhöhung des Abstraktionslevels und damit bessere Übersichtlichkeit, Visualisierung von Struktur und Verhalten führt zu fehlerfreierem Code	8
Die grafische UML Darstellung vereinfacht die Kommunikation im Team und zu Aussenstehenden 'nicht SW Mitarbeitern'	7
Bessere und verständliche Dokumentation	5
Konsistenz zwischen Code - Model – Dokumentation	4
Standardisierte Notation - Sichert Portabilität - Zukunftssicheres Know-How - SW auch für 'non Experts' verständlich - Unabhängigkeit von RTOS	4
Automatische Codegenerierung erhöht Produktivität und verhindert Fehler	4
Zusammenführen von Methoden und Prozess - Einsatz der UML in allen Prozessphasen (auch Analyse) - Ermöglicht strukturiertes Arbeiten.	3
MDD (Model Driven Development) mit Animation und Simulation sind möglich	2
Klare und einheitliche Strukturierung der Software	1
Man denkt auf Modellebene statt auf Codeebene	1
Strukturierte Darstellung von Requirements	1

Anwender-Umfrage zu Produktivitätssteigerungen der Software- Entwicklung durch den Einsatz der UML (Unified Modelling Language)

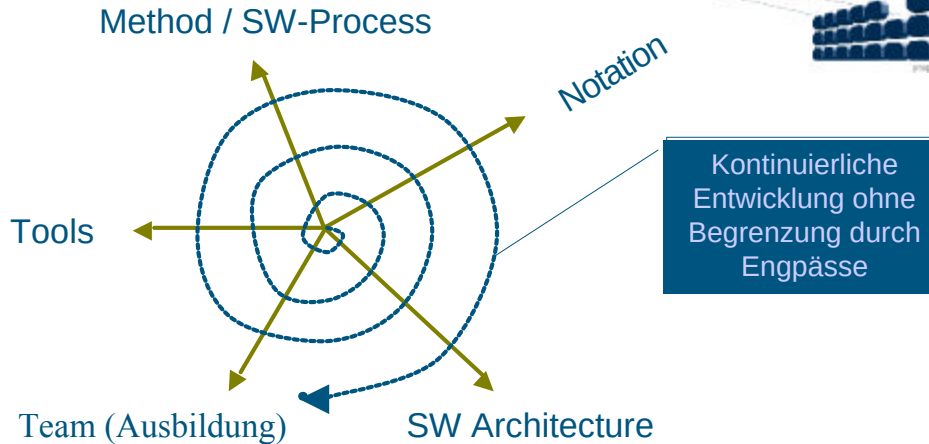


Fragen / Projekte	1	2	3	4	5	6	Durchschnitt
Wie lange hat es gedauert bis sich Produktivitätssteigerungen gegenüber der vorherigen Vorgehensweise eingestellt haben?	1	3	6	6	3	8	4,5 Monate
Wie lange setzen Sie die UML und Rhapsody ein?	24	36	50	15	18	12	25,8 Monate
Wie schätzen Sie Ihre derzeitige Produktivität (x%) gegenüber der vorherigen Vorgehensweise (100%) ein?	300	120	200	200	130	120	178,3 %
In wie weit schätzen Sie, haben Sie das Potential von UML in Kombination mit Rhapsody derzeit ausgeschöpft?	50	30	60	60	50	60	51,7 %
Wieviel Tage Schulung und Coaching haben Sie für den Einstieg in Anspruch genommen?	5	10	2	6	4	4	5,2 Tage
Was ist Ihre Target Plattform?	32 Bit	32 Bit	32 Bit	16 Bit	16 Bit	8 Bit	
Wie viele Entwickler haben beim Einstieg mit UML gearbeitet?	2	4	1	2	3	1	2,2 Entwickler
Wie viele Entwickler arbeiten derzeit mit UML?	8	2	3	4	6	2	4,2 Entwickler

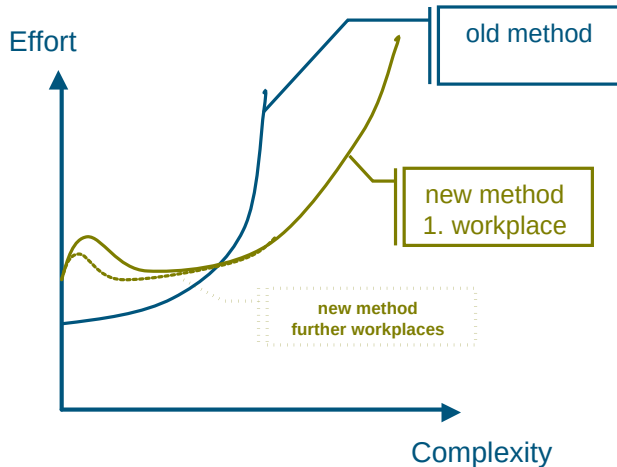
Paradigmenwechsel Engpass



Paradigmenwechsel Erfolgreich



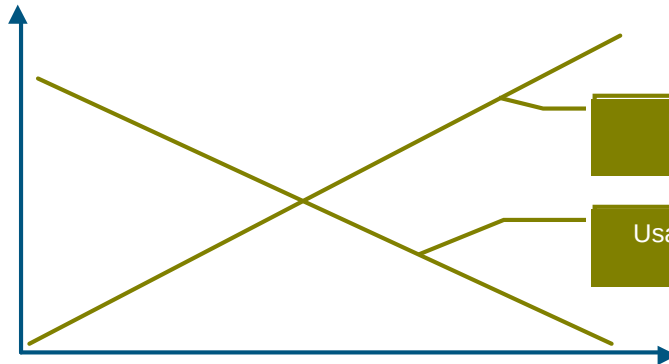
The answer to the high cost country Europe (and UK)



When complexity increases, current methods will be more and more inefficient until the time comes that a new and more efficient approach must be used. The implementation phase of a new method will generate a higher level of effort before an efficiency increase can be detected. The main question is: when will the pay-back phase be reached.

Efficiency to engineering

Efficiency



Productivity in
Development

Usage of Resources

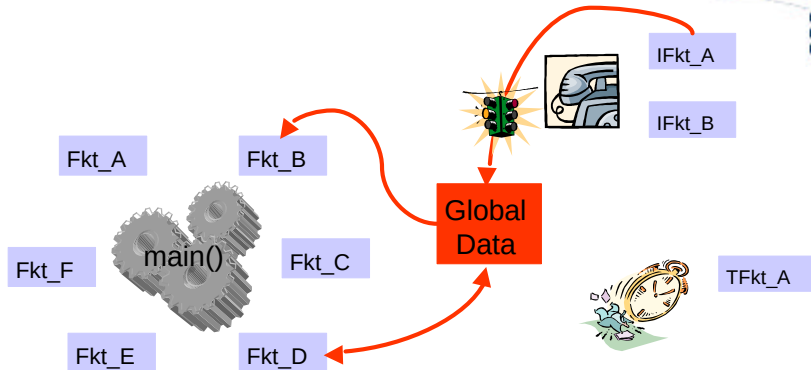
Engineering level

Willert Software Tools ■

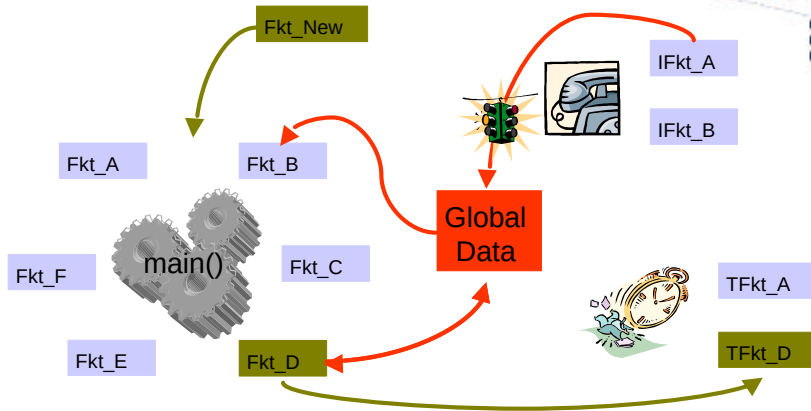
info@willert.de

+49 5722 9678 60

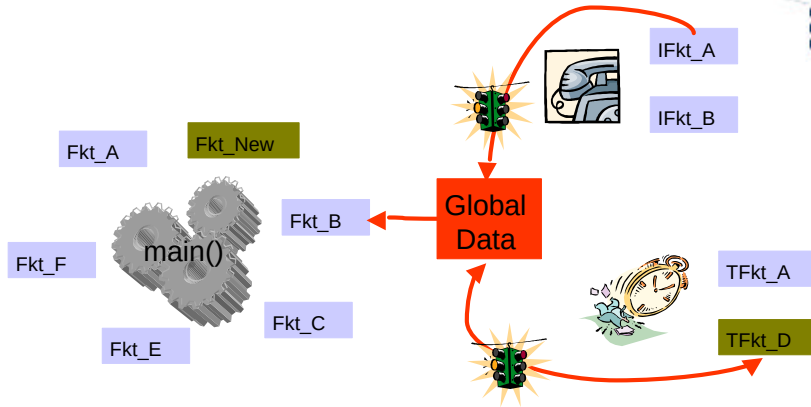
old fashioned design



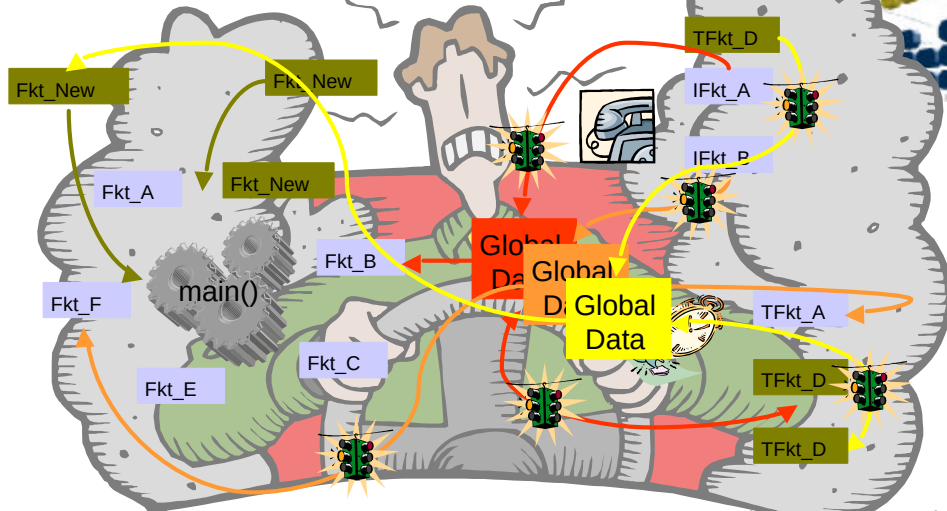
if old fashioned design grows



if old fashioned design grows



when do old fashioned designs crash?



when do old fashioned designs crash?



Problems with this approach



1. Integration of behavioral levels time, data, functionality and priority
2. Real requirements merging with architectural requirements
3. Requirements merges in the heads of the developers.

What is UML?



architect

blueprint:
376 pages of word



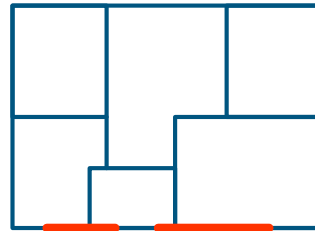
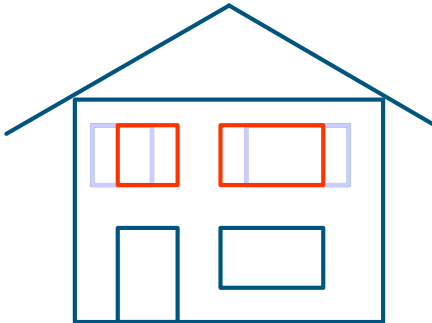
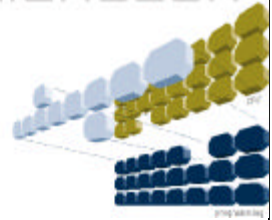
Man who wants to build

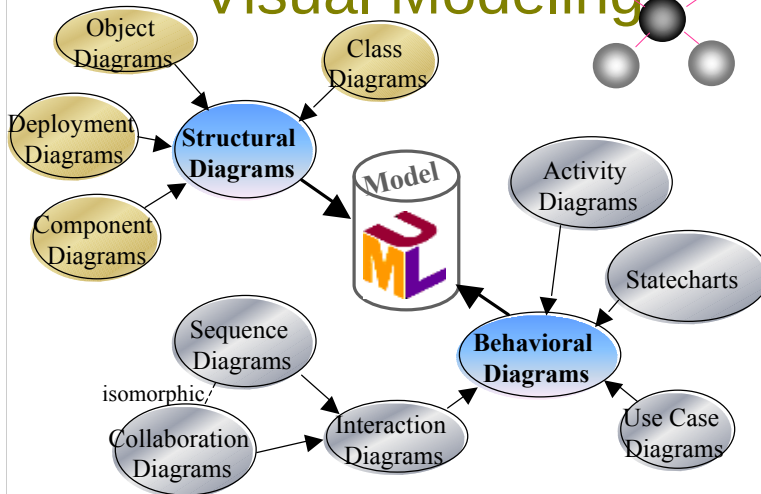
Willert Software Tools ■

info@willert.de

+49 5722 9678 60

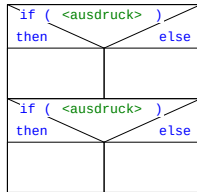
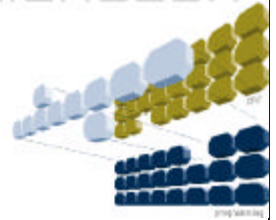
Architecture and design-errors



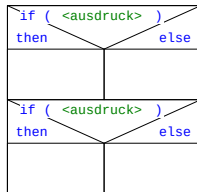


Source: I-Logix

size against complexity



K = 4

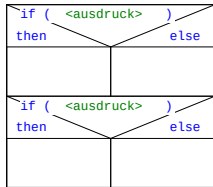
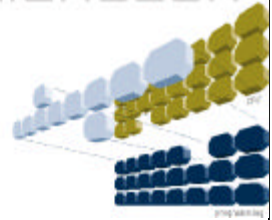
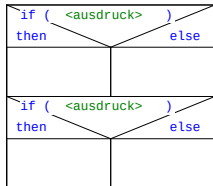


K = 4

K = 16

- complexity growth with e-function to code size

size against complexity

 $K = 4$  $K = 4$ **$K = 8$**

the trick!
OOP

First step into OOP for Embedded RT Systems is ENCAPSULAION!



- Current procedural programming dos not allow true encapsulation in an OO way.
- The elementary step into encapsulation and with that into OOP is the design of an OO runtime architecture. This is independent of the usage of C++ or UML.

ENCAPSULAION leads to OBP (Def. A.Willert)

Object Based Programming

Willert Software Tools ■

info@willert.de

+49 5722 9678 60

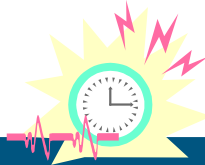
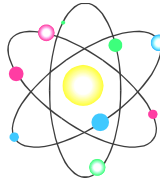
OBP löst aktuelle Probleme in der Softwareentwicklung



- Effiziente Teamarbeit -> Grundvoraussetzung für Wirtschaftlichkeit an Hochlohnstandorten
- Wiederverwendung -> in der Wiederverwendung liegt das grösste Potential für Effizienzsteigerung, jedoch nur wenn es mit geringem Aufwand möglich ist
- Verkürzung der Testzeiten -> der Aufwand für Test liegt im Schnitt bei 60% des Gesamtaufwandes
- Bessere Verstehbarkeit -> Voraussetzung für das Aufstocken des Entwicklungsteams

But where put the Interface?

- timing
- behaviour
- data flow
- priority



Fkt_New

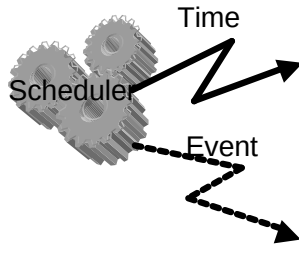
new design

Fkt_New

Fkt_New

Fkt_New

Fkt_New



Fkt_B

Fkt_C

Fkt_A

IFkt_F

Fkt_E

Message



IFkt_A



IFkt_B



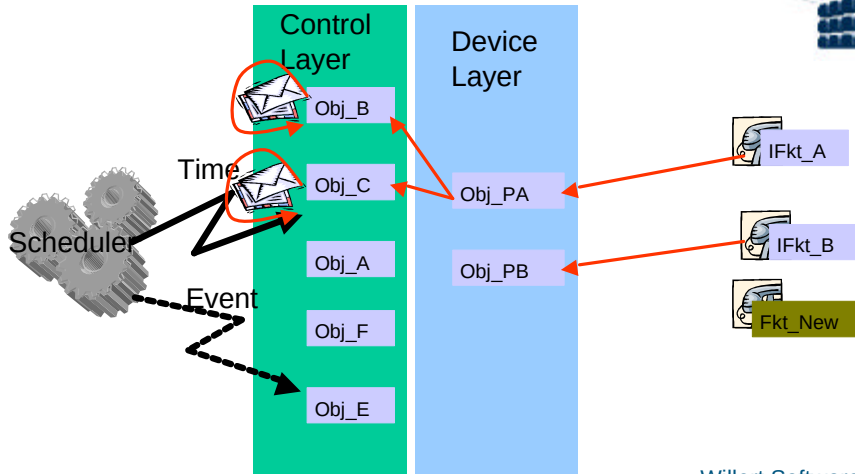
Fkt_New

Willert Software Tools

info@willert.de

+49 5722 9678 60

new design

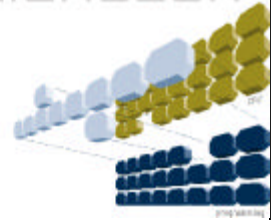


Task of the Scheduling-Pattern



- Certify the timely behavior of the single independent tasks
- Management of run-time resources while respecting priorities
- Decoupling functional behavior and communication against time behavior.

Scheduling-Pattern



Encap
sula
tion

**Allow yourself as much
encapsulation as possible!**

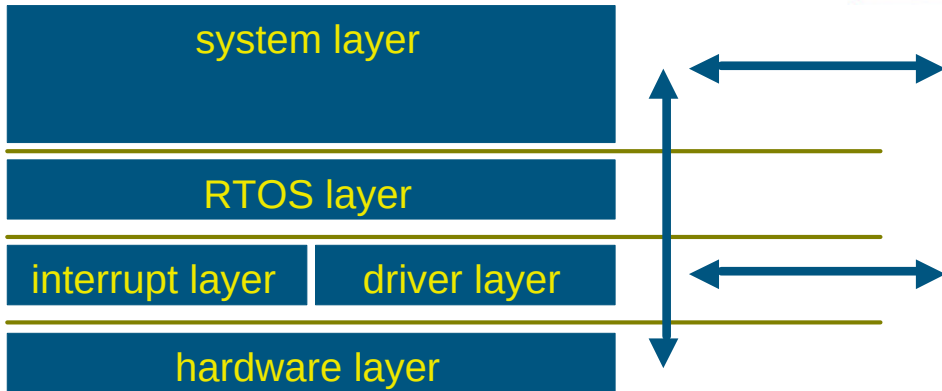
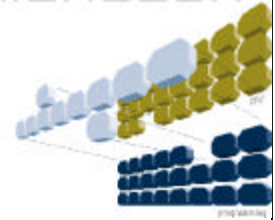
- Preemptive Multitasking
- Timesliced Multitasking
- Cooperative Multitasking
- Main Loop (Super Loop)

Tasks of the communication-Pattern

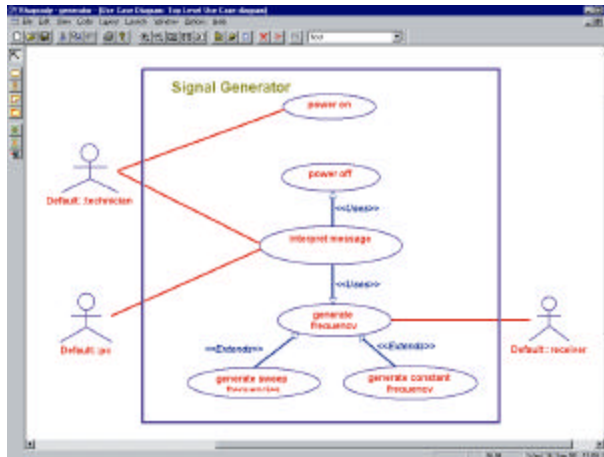
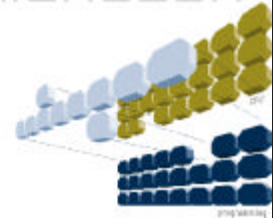


- Decoupling of the dataflow gainst runtime and functional behavior
- Communication between processes
- Communication between layers

Designpattern for Communication



Use Case

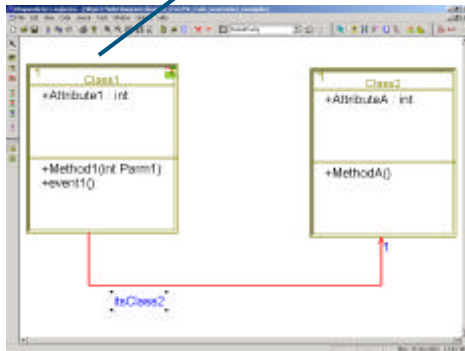


Willert Software Tools

info@willert.de

+49 5722 9678 60

Coding with UML



```

/*-----*/
/* class1.h                                     */
/*-----*/
struct Class2_t;

#define Class1Timeout_State2_id 1

/*## class TopLevel::Class1 */
struct Class1_t {
    RiCReactive ric_reactive;

    /** User explicit entries */

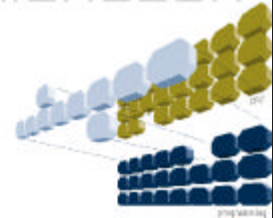
    int Attribute1; /*## attribute
Attribute1 */

    /** User implicit entries */

    struct Class2_t * itsClass2;
    /** Framework entries */

    /*states enumeration: */
    enum Class1_Enum{ Class1_RicNonState=0,
Class1_State2=1, Class1_State1=2 }
    rootState_subState,
    rootState_active;
};
  
```

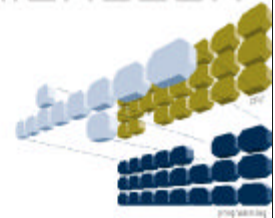
Design-Bruch 1



Implementierungen auf Basis von C benötigen zusätzliche Informationen, z.B. über die SW-Architektur und Laufzeit des Systems, die in der C-Syntax nicht enthalten sind.

Im ‚C‘ Design ist ‚rund‘ eine notwendige zusätzliche Information

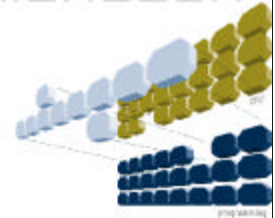
Design-Bruch 2



Die Notation UML beinhaltet bereits Informationen über die SW Architektur. Ohne ausreichende Kenntnisse und Erfahrungen entstehen Designs mit redundanter oder sogar falscher Information. Der Bruch zwischen Design und Implementation ist damit vorprogrammiert.

Im UML Design ist ‚rund‘ eine redundante Information

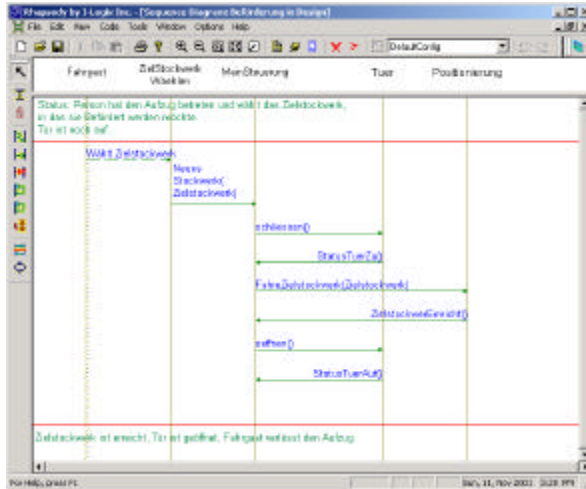
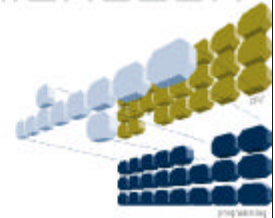
Design-Bruch 3



Eine grundlegende Voraussetzung für Return of Invest ist die Konsistenz zwischen Design und Implementation. Um diese zu sichern, ist Erfahrung oder permanentes Feedback über die Konsistenz erforderlich.

Daher braucht sie nicht mehr explizit erwähnt zu werden.

Coding with UML

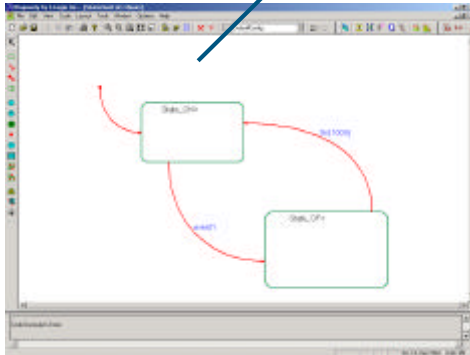


Willert Software Tools

info@willert.de

+49 5722 9678 60

Coding with UML



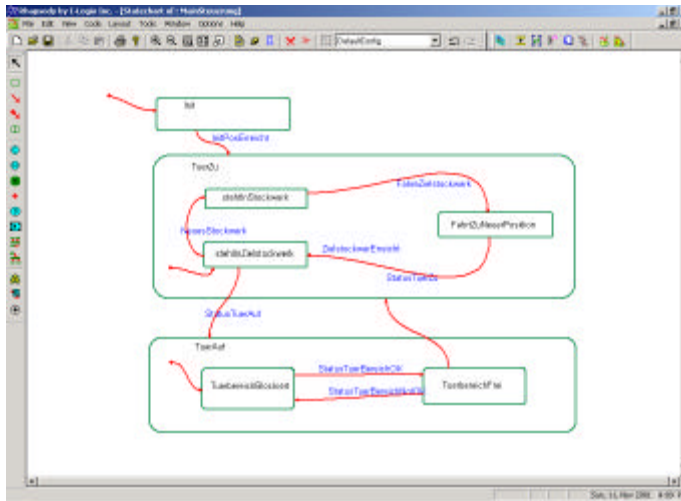
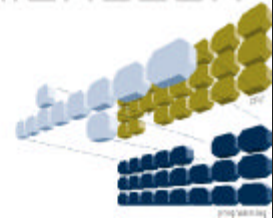
```

static int rootState_dispatchEvent(void * const void_me, short id) {
    struct Class1_t * const me = (struct Class1_t *)void_me;
    int res = eventNotConsumed;
    switch (me->rootState_active) {
        case Class1_State_ON:
        {
            if(id == event1_UML_Code_Generation_examples_id)
            {
                /*#[ state ROOT.State_OF.(Entry) */
                action2();
                /*#]*/
            }
        }
    }
    static void initStatechart(struct Class1_t * const me) {
        me->rootState_subState = Class1_RiCNonState;
        me->rootState_active = Class1_RiCNonState;
    }

    static void cleanUpRelations(struct Class1_t * const me) {
        if(me->itsClass2 != NULL)
        {
            me->itsClass2 = NULL;
        }
    }

    RiCBoolean Class1_startBehavior(struct Class1_t * const me) {
        RiCBoolean done = FALSE;
        done = RiCReactive_startBehavior(&me->ric_reactive);
        return done;
    }
  
```

Coding with UML



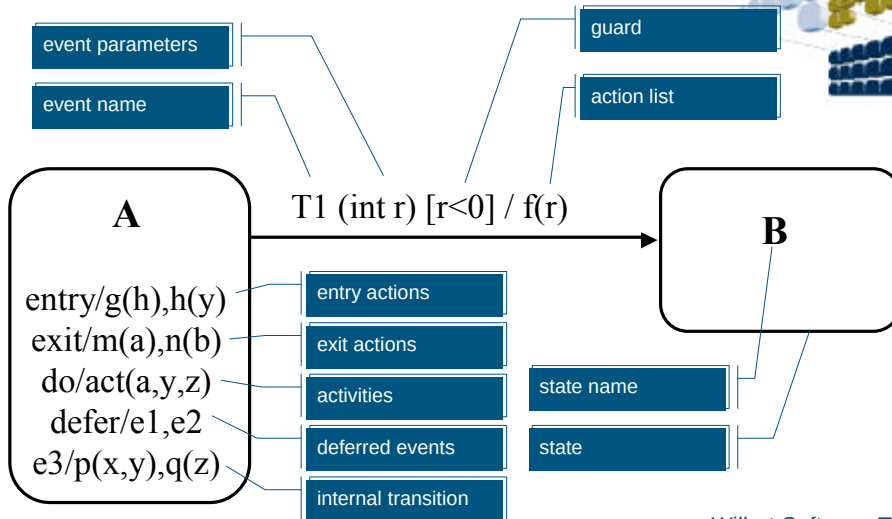
This is the diagram that is responsible for the control code generation

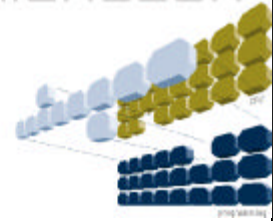
Willert Software Tools

info@willert.de

+49 5722 9678 60

state chart diagram basic syntax





Solutions

- Modelbased SW development
- Multiple views to the model
- Architectural design with Encapsulation
- Requirements are in the model
- Code – Model – Dokumentation Link

Wann ist der richtige Zeitpunkt für eine neue Methode?



- Wenn Sie beginnen Ihre vorhandene Methode mit Hilfskonstrukten zu erweitern
- Wenn Aufwandsabschätzungen zunehmend ungenau werden
- Wenn neue Mitarbeiter sehr lange brauchen um Effizient mitzuarbeiten
- Wenn Sie Emulatoren benötigen um auf Inhalte von Variablen zu tracen

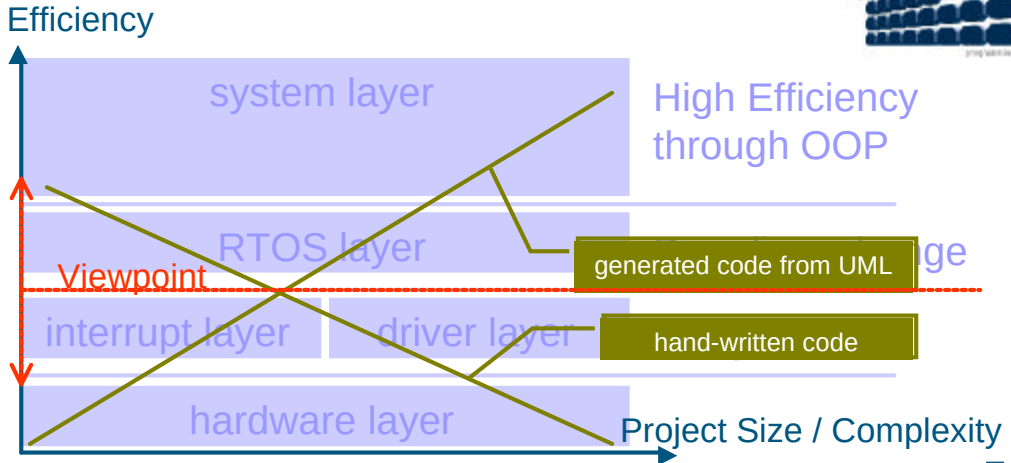
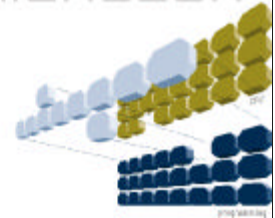


Der richtige Zeitpunkt

- Analysieren Sie Ihr eigenes Arbeitsumfeld
- Lügen Sie sich nicht in die eigene Tasche

Aufwand in Nachbesserungen, (Architekturänderungen, Fehler- Suche und Beseitigung nach Änderungen ...) oder Einsatz von Hilfskonstrukten (Zeichnungen, nachträgliche Dokumentation ...) ist weggeschmissenes Geld. Stecken sie diesen Aufwand lieber frühzeitig in einen Methodenwechsel?

why you can use UML in timecritical applications?



Willert Software Tools

info@willert.de

+49 5722 9678 60

why you can use UML in time critical applications



used resources in code size (stat.) and runtime (dyn.)

system layer

stat. 80%
dyn. 20%

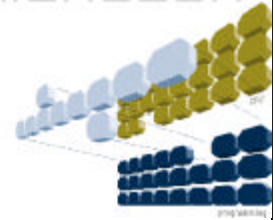
Run Time System \ RTOS layer

stat. 20%
dyn. 80%

interrupt layer

driver layer

hardware layer



Designpattern Layermodel

system layer

High Efficiency
through OOP

RTOS layer

Paradigm change

interrupt layer

driver layer

Progr. in C

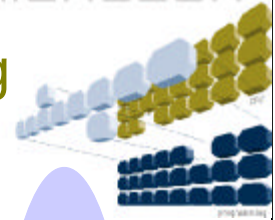
hardware layer

Willert Software Tools

info@willert.de

+49 5722 9678 60

UML Footprint and Timing



system layer

Overhead Code generation

10 to 30 % Project start
0 to -20 % during long time projects

100µs
(design min. 1 ms)

Framework OSAL

RTS \ RTOS layer

Footprint
4 – 20 KByte

10µs
(design min. 100µs)

interrupt layer

driver layer

1 us

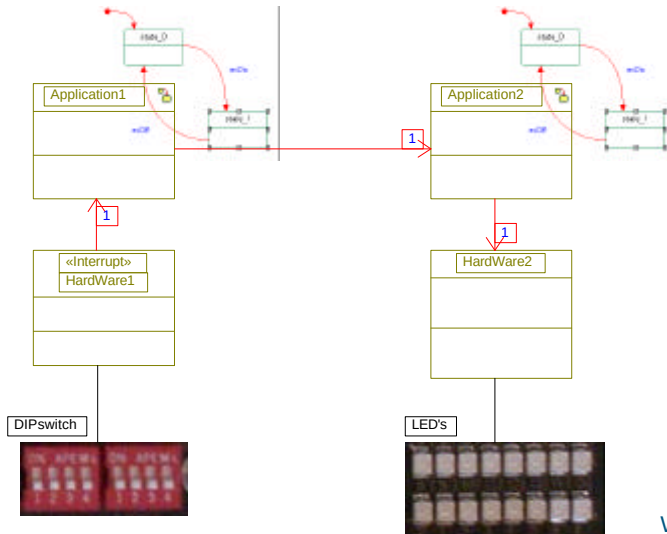
Hardware C167, 20MHz, 16 bit ext. Mem.

Willert Software Tools

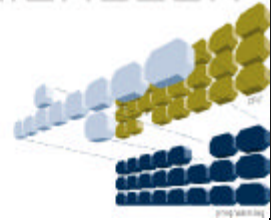
info@willert.de

+49 5722 9678 60

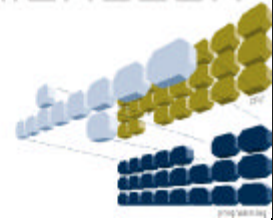
Model for Measurement



RUN-TIME-System



Avoid tight coupling between layers



- Interrupt disabling
- Methods
- Global Data (Shared Memory)
- Semaphores
- Unsuitable combinations of scheduling patterns.

Hardware dependency



CASE Tools

UML (Rhapsody in C)

UML (Rhapsody in C++)

Config.Management

Static Analyser

Nassi Shneiderman
Editor (EasyCODE)

CPU independent

- possible to use same licence, if same language
- high return of invest (ROI) SW and Tools

Framework OSAL

Run Time System

OSE

CMX RTX

OSEK

EUROS

...

More or less CPU dependent

- have to buy new RTOS licenses
- only SW compatibility if RTOS supports new HW
- high ROI in SW, no ROI Tools

Compiler c / c++

Keil C166

GNU ARM

Tasking M16C

GHS PowerPC

...

Total CPU dependent

- usually Tool from different vendor, not SW compatible
- have to buy new tool licenses
- compatibility problems occur even by switching to new version
- less ROI in SW, no ROI tools

Debugger

ROM Debugger

Emulator

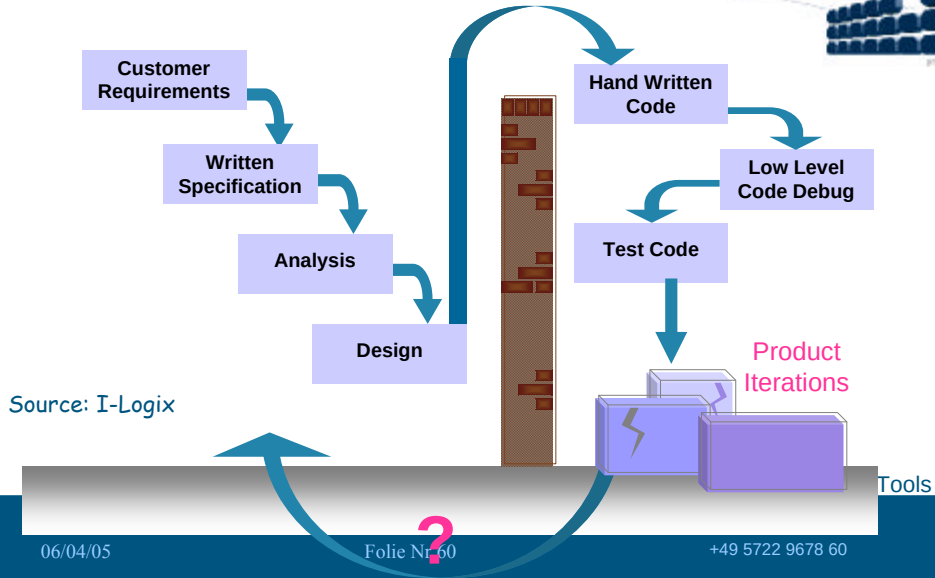
Simulator

JTAG

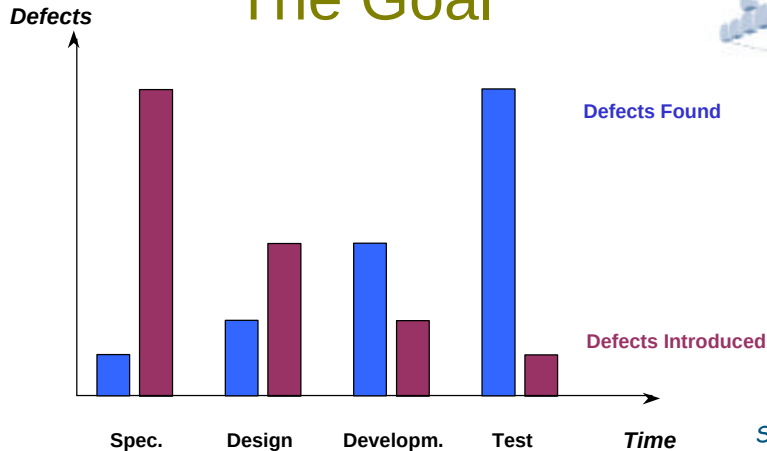
...

Willert Software Tools

The Process



The Goal



Source: I-Logix

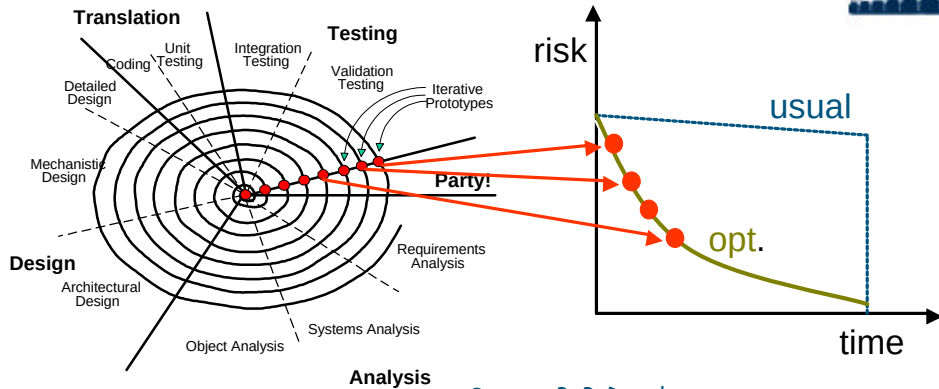
Source: "Software Engineering" Ramamoorthy et al, IEEE Computer, 10/84

Willert Software Tools

info@willert.de

+49 5722 9678 60

the ROPES process



Source: B. P. Douglas

Willert Software Tools

info@willert.de

+49 5722 9678 60

USDP (Unified SW-Development Process)

Core Workflows

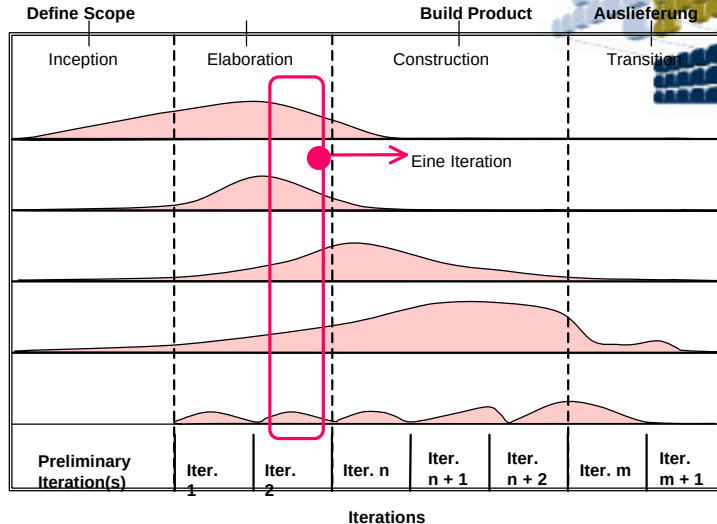
Requirements

Analysis

Design

Implementation

Test



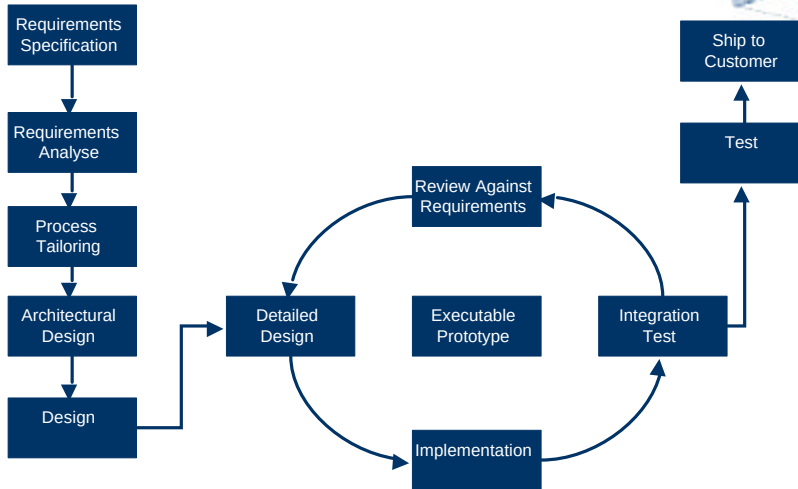
Source: Rational

Willert Software Tools

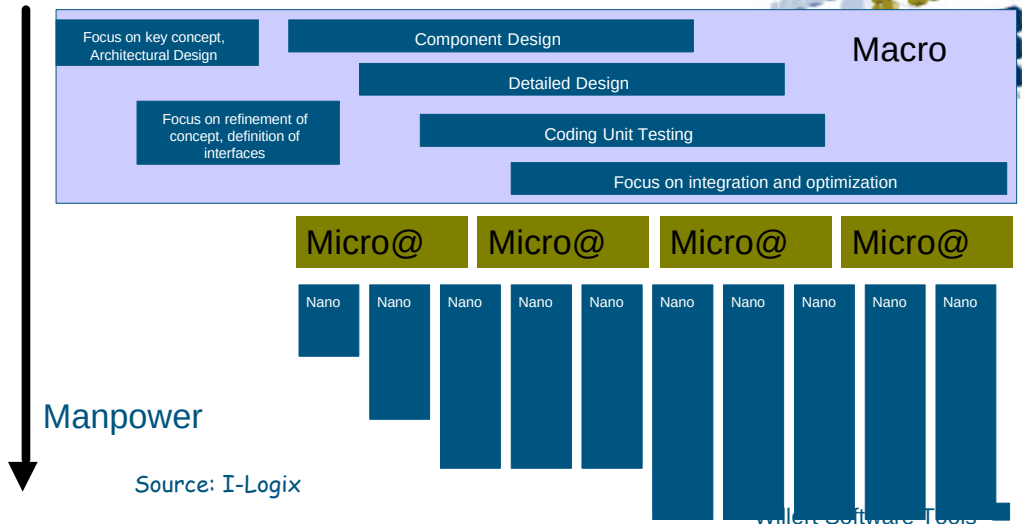
info@willert.de

+49 5722 9678 60

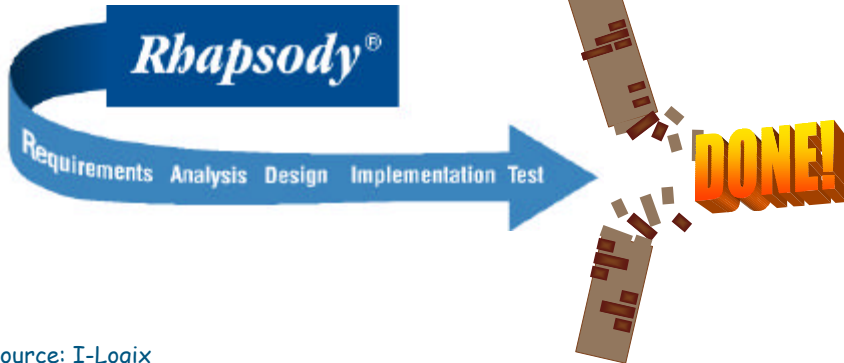
Software Process



important for teamwork

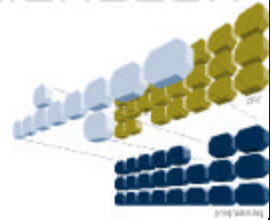


The Goal



Source: I-Logix

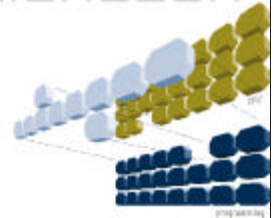
The key to higher productivity: Higher quality!



- Quality through organizational measures like ISO 9000, CMM, V-Model
- Quality through better communication, less stress and a good working environment

Peter Molzberger comments on that:

„To put it in a negative humorous way, neurotic software developers will produce neurotic software“



Return on Investment

Salary for a developer
per year incl. other
costs excl. introduction
time

Ca. 80.000,- €

Investment for the equipment of
5 workplaces with UML incl
learning measurements, excl.
Intro time.

Ca. 70.000,- €

Yearly cost after 1st year

Ca. 80.000,- € / Year

Yearly cost after 1st year

Ca. 15.000,- € / Year

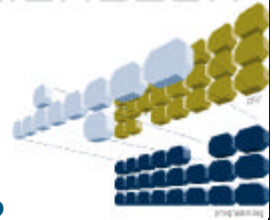


The decision!

1. UML and Rhapsody promise you
2. Footprint fits (min. 4 K Byte)
3. Timing fits (1ms)

Next step

1. UML Start-Up Training
2. Evaluation of the environment
3. Decide



- Welche Fragen haben Sie noch?
- Was ist Ihre wichtigste Erkenntnis aus dem heutigen Tag
- Unsere Wichtigsten Botschaften an Sie:
 - Lego: Sichern, das Model und Implementation konsistent bleiben
 - Jonglieren: Schrittweiser Einstieg mit schritten die wirklich die Anwendung der UML Trainieren und direkt zu return of invest führen.