

ABAP and Event-Based Components

Hamburg, July 2013



... Work in Progress ...

If we want to build evolvable software, then we must find a way to control resp. reduce the coupling of its components.

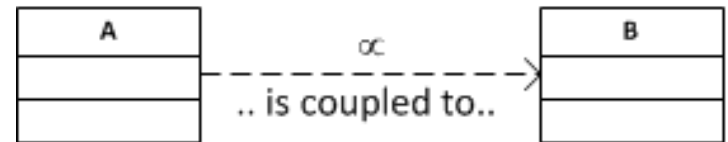
High software quality:

- Low degree of coupling of its components
- Less complexity
- Better to understand, test and maintain
- Enhance reusability

Coupling is a measure and a dedicated symbol is used to represent coupling in diagrams.

- Coupling – A measure of the strength of association established by connection from one component to another.
- Cohesion – The degree of connections among the elements of a single components.

A component is a self-documenting entity containing classes and/or objects.



Coupling is bad, but not all forms of coupling are equal!

Coupling affects...

- Compile time, run time or both
- User defined types

Types of Coupling

- Worst: Duplicated Logic -> use the DRY principle (Don't Repeat Yourself)
- Next worse: Coupling that impacts compile time -> Static Coupling
- Most benign form: Coupling, that occurs only at run time -> Dynamic Coupling

Coupling is called static, if the coupling between two objects effects build time.

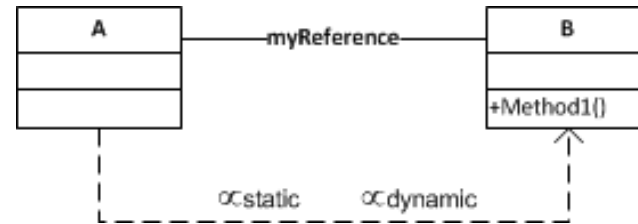
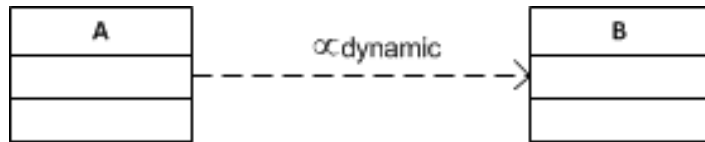
To produce A's executable code from source code, some part of B must be present

A contains references to symbols in B, e.g. a constant, variable or method.

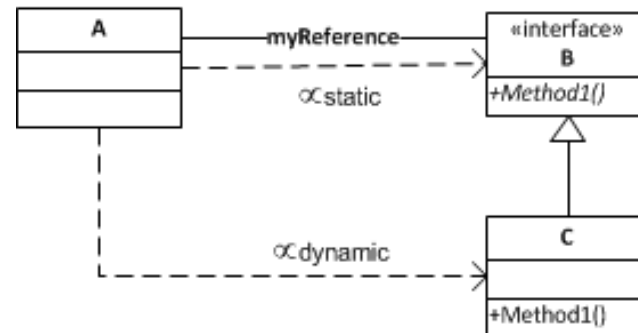
- Declaration and Implementation:
C#, VB, Java, ABAP – Global Class
- Declaration:
C++, Object Pascal, ABAP – Local Class



Coupling is called dynamic, if the coupling between two objects effects run time.



Using Interfaces you can transform a static coupling into a dynamic one!

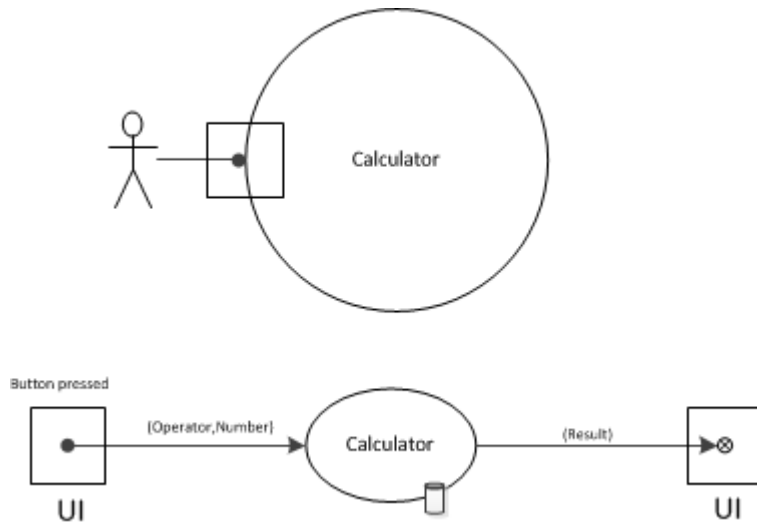


The Idea of Event-Based Components is to use events instead of coupling components directly with each other.

Rules

- The more complex a class or component is, the more it should be decoupled
- Coupling should be introduced into simpler classes and components first
- Reduce the overall coupling in a system to the lowest level possible
- Put the coupling into desirable places

Calculator Example



| INPUT | | OUTPUT |
|--------|----------|--------|
| Number | Operator | Result |
| 2 | + | 2 |
| 3 | * | 5 |
| 4 | = | 20 |
| 2 | + | 2 |


```

6 namespace Calculator
7 {
8     static class Program
9     {
10         [STAThread]
11         static void Main()
12         {
13             Application.EnableVisualStyles();
14             Application.SetCompatibleTextRenderingDefault(false);
15
16             var r = new Caluculator();
17             var c = new CalculatorUI();
18
19             c.On_operator += r.Calculate;
20             r.On_result += c.display;
21
22             Application.Run(c);
23         }
24     }
25 }

```

Wire-up

∞

```

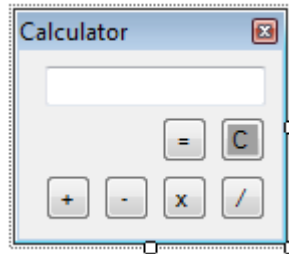
6 namespace Calculator
7 {
8     class Caluculator
9     {
10         string prev_operator;
11         int result;
12
13         public event Action<int> On_result;
14
15         public void Calculate(string @operator, int value)
16         {
17             if (@operator == "C")
18             {
19                 prev_operator = "";
20                 result = 0;
21             }
22             else
23             {
24                 if (prev_operator == null || prev_operator == "=")
25                 {
26                     prev_operator = @operator;
27                     result = value;
28                 }
29                 else
30                 {
31                     switch (prev_operator)
32                     {
33                         case "+": result = result + value; break;
34                         case "-": result = result - value; break;
35                         case "/": result = result / value; break;
36                         case "x": result = result * value; break;
37                         default: break;
38                     }
39                     prev_operator = @operator;
40                 }
41             }
42             On_result(result);
43         }
44     }
45 }

```

```

10 namespace Calculator
11 {
12     public partial class CalculatorUI : Form
13     {
14         public event Action<string, int> On_operator;
15
16         public CalculatorUI()
17         {
18             InitializeComponent();
19         }
20
21         private void calculate_Click(object sender, EventArgs e)
22         {
23             Button button1 = sender as Button;
24             On_operator(button1.Text, int.Parse(input_output.Text));
25         }
26
27         public void display(int result)
28         {
29             input_output.Text = result.ToString();
30             input_output.Focus();
31         }
32     }
33 }

```



Event

Event

```

9 report zprogram4.
10
11 data:
12   c type ref to zcl_calculator4_ui,
13   r type ref to zcl_calculator4.
14
15 start-of-selection.
16
17 create object c.
18 create object r.
19
20 set handler r->calculate for c.
21 set handler c->display for r.
22
23 c->run( ).
    
```

ZCL_CALCULATOR4

Method: CALCULATE
Description: Berechnen

Visibility: Public, Protected, Private
Method: Static, Instance

Abstract
 Final
 Event handler for

Class/interface: ZCL_CALCULATOR4_UI
Event: ON_OPERATOR

```

1 method calculate.
2   if operator = 'C'.
3     clear: prev_operator, result.
4     elseif prev_operator is initial or
5       prev_operator = 'EQUAL'.
6       prev_operator = operator.
7       result = number.
8   else.
9     case prev_operator.
10      when '+'.
11        result = result + number.
12      when '-'.
13        result = result - number.
14      when '*'.
15        result = result * number.
16      when '/'.
17        result = result / number.
18    endcase.
19    prev_operator = operator.
20  endif.
21
22  raise event on_result
23  exporting
24    result = result.
25 endmethod.
    
```

ZCL_CALCULATOR4_UI

Method: DISPLAY
Description: anzeigen

Visibility: Public, Protected, Private
Method: Static, Instance

Abstract
 Final
 Event handler for

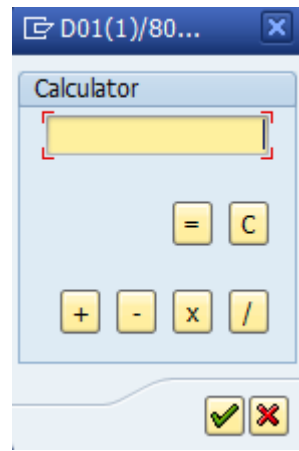
Class/interface: ZCL_CALCULATOR4
Event: ON_RESULT

```

1 method display.
2   me->result = result.
3 endmethod.

1 method calculate.
2   raise event on_operator
3   exporting
4     operator = operator
5     number = number.
6 endmethod.
    
```

This destroys everything!



```

1 type-pool zebc .
2
3 define zebc_wire_up.
4     &1-object = &2.
5     &1-method = &3.
6 end-of-definition.

7
8 types:
9     begin of zebc_event,
10    object type ref to object,
11    method type seocpdname,
12 end of zebc_event.

9 report zprogram5.
10
11 type-pools:
12     zebc.
13
14 data:
15     c type ref to zcl_calculator5_ui,
16     r type ref to zcl_calculator5.
17
18 start-of-selection.
19
20 create object c.
21 create object r.
22
23 zebc_wire_up c->on_operator r 'CALCULATE'.
24 zebc_wire_up r->on_result c 'DISPLAY'.
25
26 c->run( ).
    
```

ZCL_CALCULATOR5

```

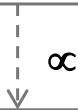
1 method calculate.
2     if operator = 'C'.
3         clear: prev_operator, result.
4     elseif prev_operator is initial or
5         prev_operator = 'EQUAL'.
6         prev_operator = operator.
7         result = number.
8     else.
9         case prev_operator.
10            when '+'.
11                result = result + number.
12            when '-'.
13                result = result - number.
14            when '*'.
15                result = result * number.
16            when '/'.
17                result = result / number.
18            endcase.
19            prev_operator = operator.
20        endif.
21
22        call method on_result-object->(on_result-method)
23            exporting
24                result = result.
25
26        * RAISE EVENT on_result
27        * EXPORTING
28        *     result = result.
29    endmethod.
    
```

ZCL_CALCULATOR5_UI

```

1 method display.
2     me->result = result.
3 endmethod.

1 method calculate.
2     call method on_operator-object->(on_operator-method)
3     exporting
4         operator = operator
5         number = number.
6 * RAISE EVENT on_operator
7 * EXPORTING
8 *     operator = operator
9 *     number = number.
10 endmethod.
    
```



CAS-Nummern pflegen

Einfuhrländer

CAS-RN (Pharma)

Bezeichnung:

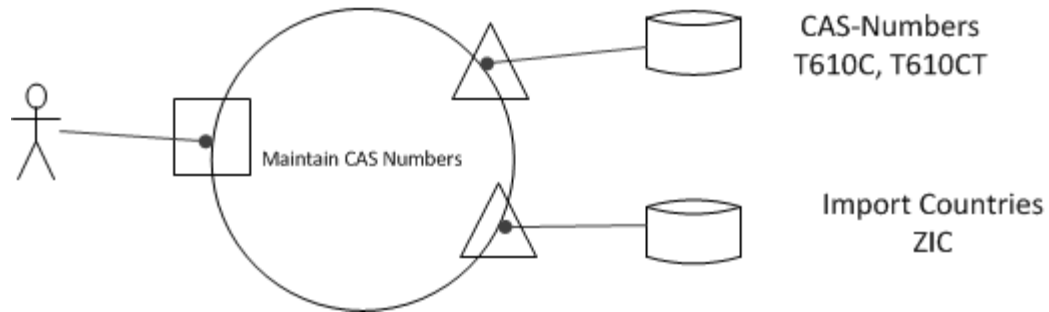
| Ein | Bezeichnung | Bezeichnung |
|-----|-------------|-------------|
| BE | Caffeine | |
| DE | Caffeine | |
| DK | Caffeine | |
| ES | Caffeine | |
| FI | Caffeine | |
| FR | Caffeine | |
| GB | Caffeine | |
| IT | Caffeine | |
| NL | Caffeine | |
| PL | Caffeine | |
| SE | Caffeine | |
| SI | Caffeine | |

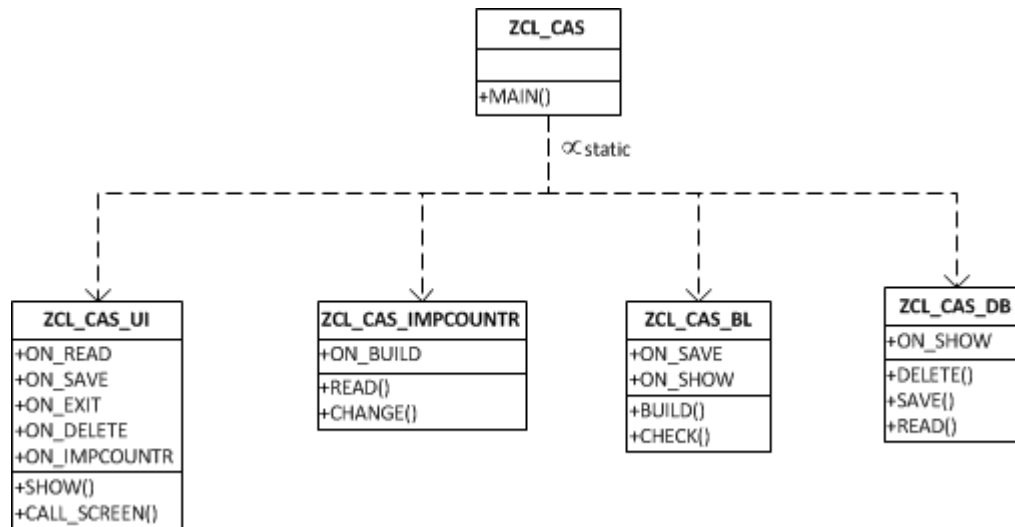
Sicht "Einfuhrland" ändern: Übersicht

Neue Einträge

| Lnd | Bezeichnung |
|-----|-----------------|
| BE | Belgien |
| DE | Deutschland |
| DK | Dänemark |
| ES | Spanien |
| FI | Finnland |
| FR | Frankreich |
| GB | Verein. Königr. |
| IT | Italien |
| NL | Niederlande |
| PL | Polen |
| SE | Schweden |
| SI | Slowenien |

SAP



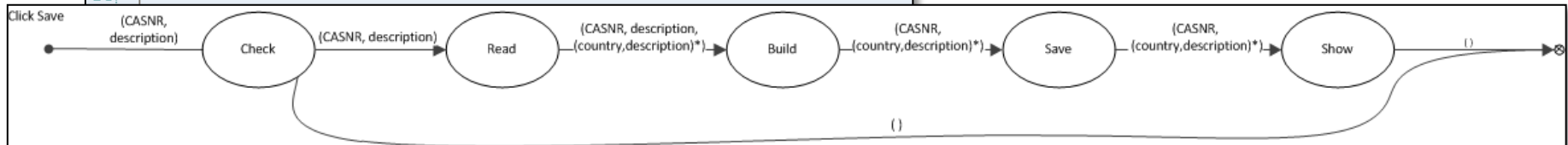


The transactions calls the method MAIN, where the binding is done and finally the first screen is called.

```

1  method main.
2  create object ui.
3  create object imp_countr.
4  create object bl.
5  create object db.
6
7  * Enter - Read and show import countries for a given CAS number.
8  zebc_wire_up1 event ui->on_read bl 'CHECK' parameter 'CAS' ui->cas.
9  zebc_wire_up1 event ui->on_read db 'READ' parameter 'CAS' ui->cas.
10 zebc_wire_up event db->on_show ui 'SHOW'.
11
12 * Save - Save text for a given CAS number and show the changes
13 zebc_wire_up event ui->on_save bl 'CHECK'.
14 zebc_wire_up event ui->on_save imp_countr 'READ'.
15
16 zebc_wire_up event imp_countr->on_build bl 'BUILD'.
17
18 zebc_wire_up event bl->on_save db 'SAVE'.
19 zebc_wire_up event bl->on_show ui 'SHOW'.
20

```



```

27 zebc_wire_up event ui->on_delete db 'DELETE'.
28
29 ui->call_screen( ).
30 endmethod.

```

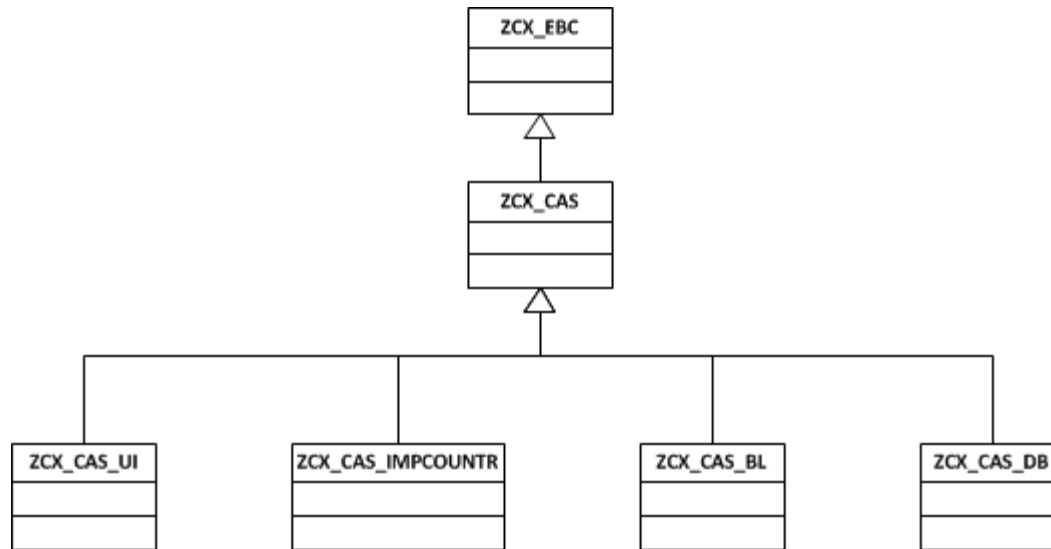
Each user command raises an event, which can have more than one handler. All events have the same signature.

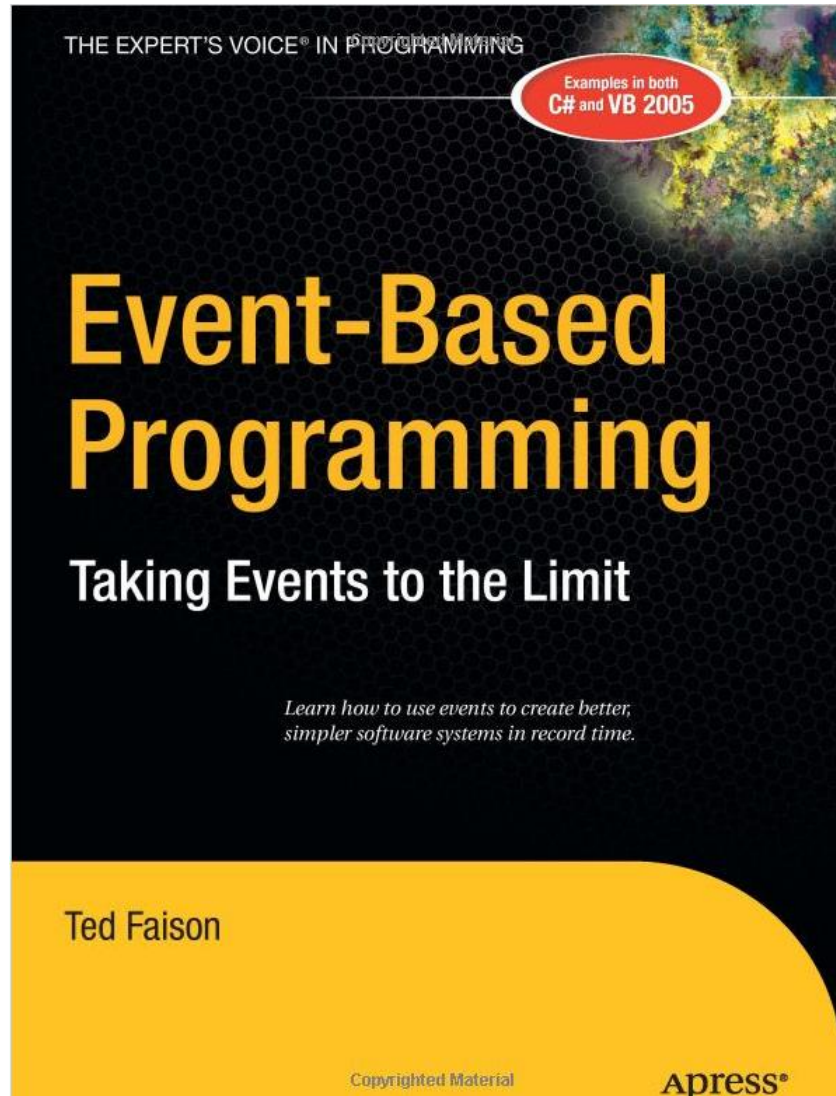
| Art | Parameter | Typisierung | Beschreibung |
|-----|-----------|------------------|------------------------|
| ▶ | EVENTS | TYPE ZEBC_EVENTS | |
| ⚡ | ZCX_EBC | | Event-Based Components |

| Methode | RAISE | aktiv |
|---------|--|-------|
| 1 | method raise. | |
| 2 | if events is not initial. | |
| 3 | loop at events into event. | |
| 4 | if event-object is not initial and event-method is not initial. | |
| 5 | try. | |
| 6 | call method me->event-object->(me->event-method) | |
| 7 | parameter-table event-parameter. | |
| 8 | endtry. | |
| 9 | endif. | |
| 10 | endloop. | |
| 11 | endif. | |
| 12 | endmethod. | |


```

7  module user_command_1000 input.
8  try.
9  case ok_code.
10 when 'ENTER' or 'SHOW'.
11   me->raise( me->on_read ).
12 when 'SAVE'.
13   loop at me->on_save into me->event.
14     call method me->event-object->(me->event-method)
15     exporting
16       cas = me->cas.
17   endloop.
18   message s001.
19 when 'DELE'.
20   loop at me->on_delete into me->event.
21     call method me->event-object->(me->event-method)
22     exporting
23       cas = me->cas.
24   endloop.
25 when 'IMPCOUNTR'.
26   loop at me->on_imp countr into me->event.
27     call method me->event-object->(me->event-method)
28     exporting
29       cas = me->cas.
30   endloop.
31 when 'COPY'.
32   call transaction 'ZDM25'.
33 when others.
34   endcase.
35 catch zcx_cas into exception.
36   text = exception->get_text( ).
37   message text type exception->msgty.
38 endtry.
39 endmodule.                " USER COMMAND 1000 INPUT
  
```



Thank you for your attention!

ADventas Consulting
Peter Langner
Kattjahren 8
22359 Hamburg

Tel. 040 60 55 94 01
Fax 040 60 55 94 00
Mobile 0151 12 21 48 67

Peter.Langner@adventas.de
<http://www.adventas.de>