ADventas
Consulting

ABAP and Event-Based Components

Hamburg, July 2013

… Work in Progress …

ADventas Consulting

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:

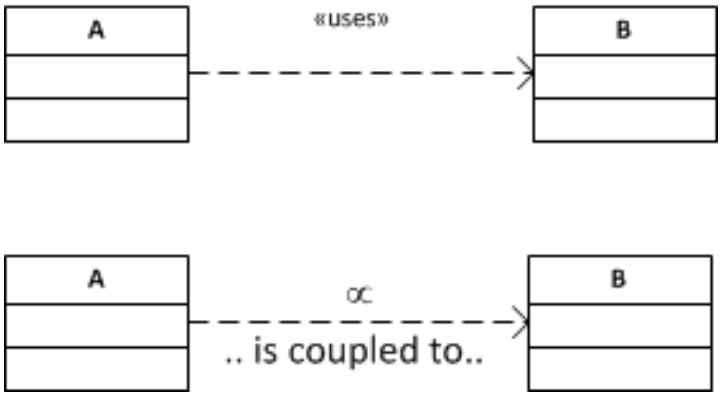
- Low degree of coupling of it‘s components
- Less complexity
- Better to understand, test and maintain
- Enhance reusability

ADventas Consulting

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.



ADventas Consulting

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.

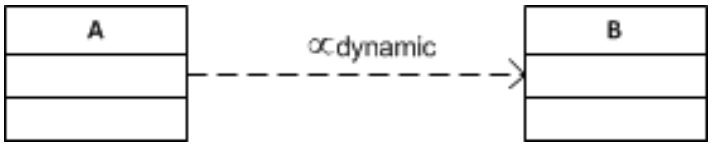
To produce A's executable code from source code, some part of B must be present

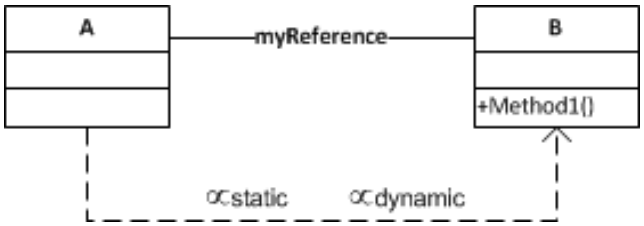
A contains references to symbols in B, e.g. a constant, variable or method.

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

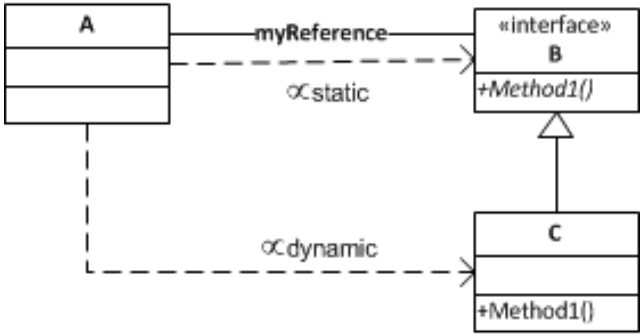
ADventas Consulting

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!



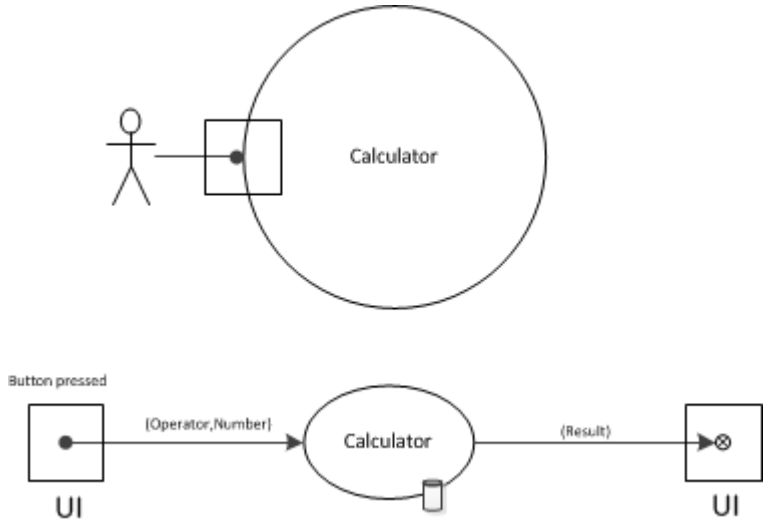
ADventas Consulting

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 }
26
```

Wire-up

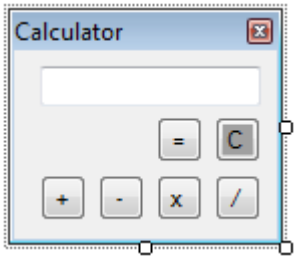
∝

```
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
18             if (@operator == "C")
19             {
20                 prev_operator = "";
21                 result = 0;
22             }
23             else
24             {
25                 if (prev_operator == null || prev_operator == "=")
26                 {
27                     prev_operator = @operator;
28                     result = value;
29                 }
30                 else
31                 {
32                     switch (prev_operator)
33                     {
34                         case "+": result = result + value; break;
35                         case "-": result = result - value; break;
36                         case "/": result = result / value; break;
37                         case "x": result = result * value; break;
38                         default: break;
39                     }
40                     prev_operator = @operator;
41                 }
42             }
43             On_result(result);
44         }
45     }
46 }
```

Event

∝

```
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 calulate_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         }
```



Event

ZPROGRAM4

```
 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( ).
```

ADventas Consulting

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.
```

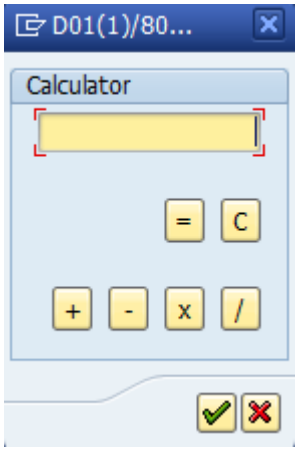
ZCL_CALCULATOR4_UI

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!



```
type-pool zebc .

define zebc_wire_up.
  &1-object = &2.
  &1-method = &3.
end-of-definition.
```

```
types:
  begin of zebc_event,
    object  type ref to object,
    method  type seocpdname,
  end of zebc_event.
```

```
report  zprogram5.

type-pools:
  zebc.

data:
   c type ref to zcl_calculator5_ui,
   r type ref to zcl_calculator5.

start-of-selection.

  create object c.
  create object r.

  zebc_wire_up c->on_operator r 'CALCULATE'.
  zebc_wire_up r->on_result c 'DISPLAY'.

  c->run( ).
```

∝

ZCL_CALCULATOR5

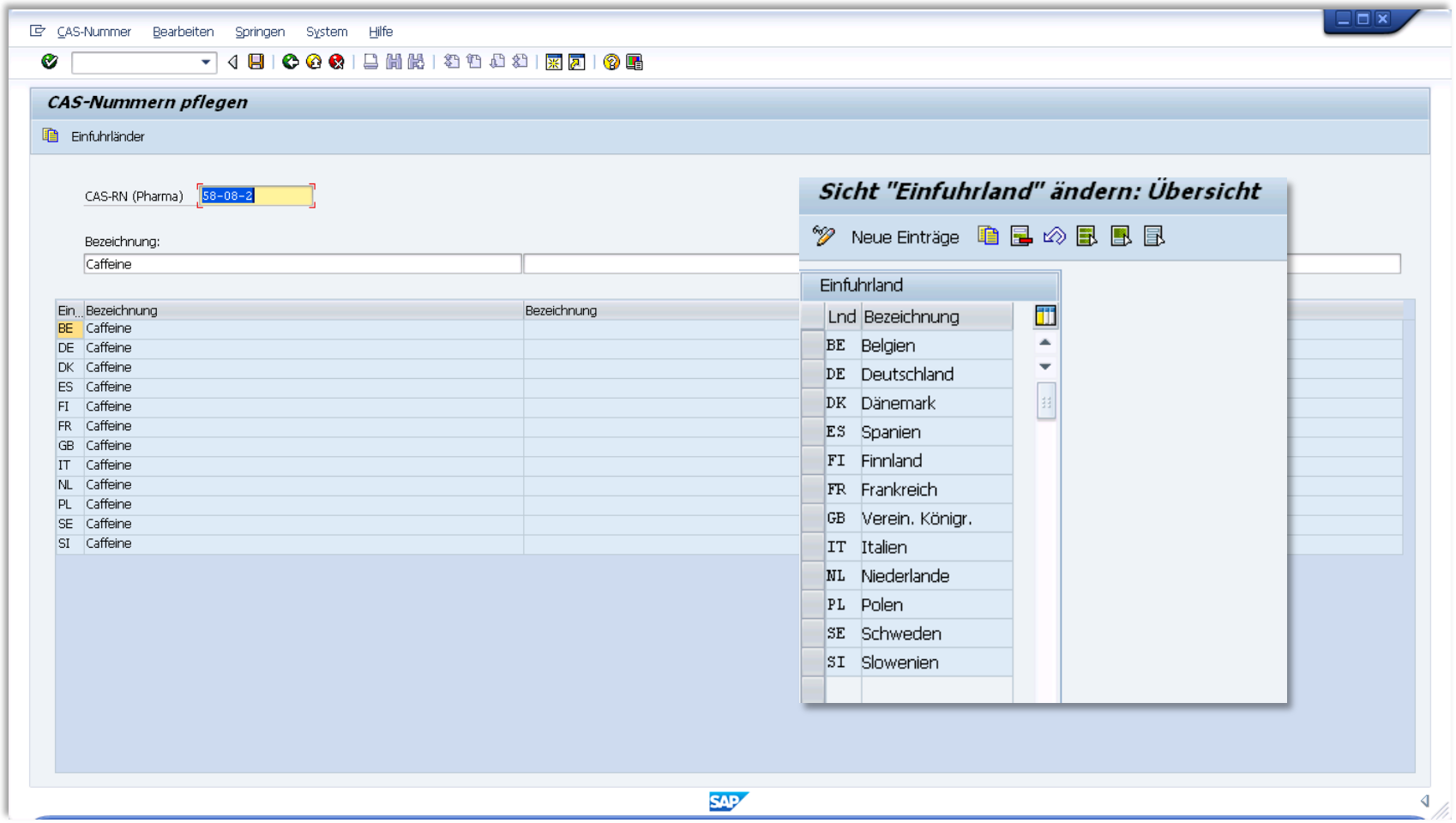
```
method calculate.
 if operator = 'C'.
    clear: prev_operator, result.
  elseif prev_operator is initial or
         prev_operator = 'EQUAL'.
    prev_operator = operator.
    result = number.
  else.
    case prev_operator.
      when '+'.
        result = result + number.
      when '-'.
        result = result - number.
      when '*'.
        result = result * number.
      when '/'.
        result = result / number.
    endcase.
    prev_operator = operator.
  endif.


  call method on_result->(on_result-method)
    exporting
      result = result.
*  RAISE EVENT on_result
*    EXPORTING
*      result = result.
endmethod.
```

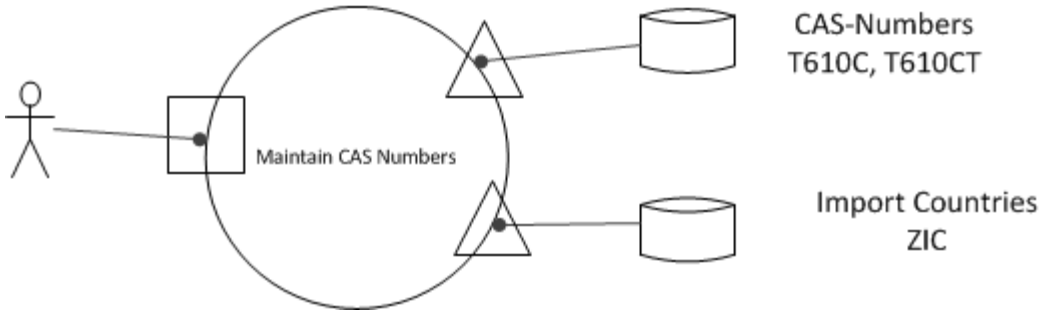
∝

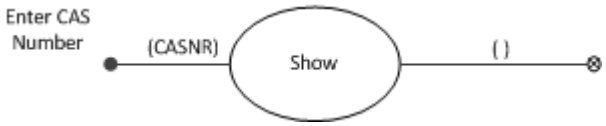
ZCL_CALCULATOR5_UI

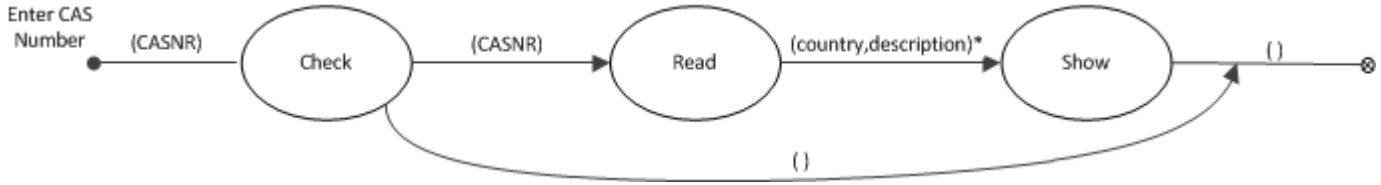
```
method display.
   me->result = result.
endmethod.
```

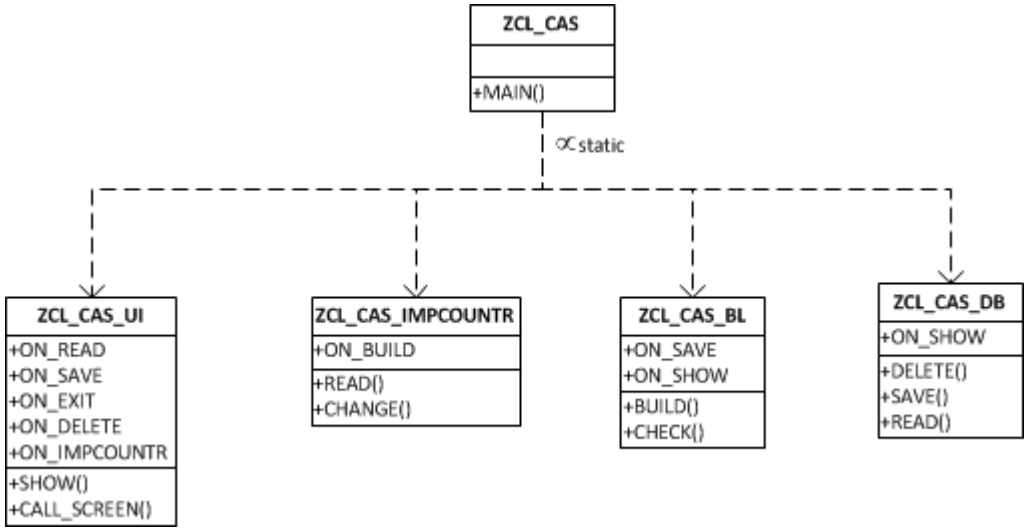
```
method calculate.
  call method on_operator->(on_operator-method)
    exporting
      operator = operator
      number   = number.
*  RAISE EVENT on_operator
*    EXPORTING
*      operator = operator
*      number = number.
endmethod.
```


CAS-Nummer
Bearbeiten
Springen
System
Hilfe
CAS-Nummern pflegen
Einfuhrländer
CAS-RN (Pharma)
58-08-2
Bezeichnung:
Caffeine
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
Einfuhrland
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


CAS-Numbers
T610C, T610CT
Maintain CAS Numbers
Import Countries
ZIC


Enter CAS
Number
(CASNR)
Show
( )

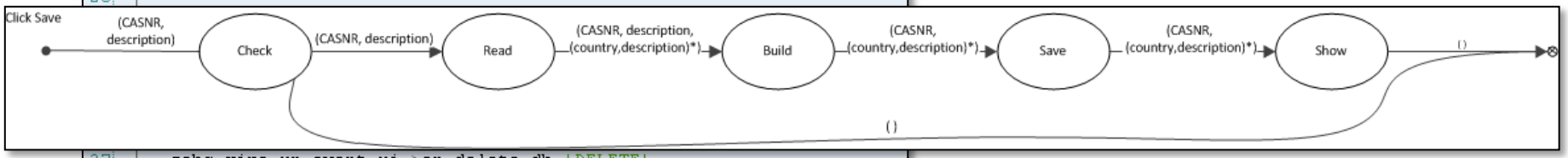

Enter CAS
Number
(CASNR)
Check
(CASNR)
Read
(country,description)*
Show
( )
( )


ZCL_CAS
+MAIN()
static
ZCL_CAS_UI
+ON_READ
+ON_SAVE
+ON_EXIT
+ON_DELETE
+ON_IMPCOUNTR
+SHOW()
+CALL_SCREEN()
ZCL_CAS_IMPCOUNTR
+ON_BUILD
+READ()
+CHANGE()
ZCL_CAS_BL
+ON_SAVE
+ON_SHOW
+BUILD()
+CHECK()
ZCL_CAS_DB
+ON_SHOW
+DELETE()
+SAVE()
+READ()

ADventas Consulting

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

```
method main.
  create object ui.
  create object imp_countr.
  create object bl.
  create object db.

* Enter - Read and show import countries for a given CAS number.
  zebc_wire_up1 event ui->on_read bl 'CHECK' parameter 'CAS' ui->cas.
  zebc_wire_up1 event ui->on_read db 'READ' parameter 'CAS' ui->cas.
  zebc_wire_up event db->on_show ui 'SHOW'.

* Save - Save text for a given CAS number and show the changes
  zebc_wire_up event ui->on_save bl 'CHECK'.
  zebc_wire_up event ui->on_save imp_countr 'READ'.

  zebc_wire_up event imp_countr->on_build bl 'BUILD'.

  zebc_wire_up event bl->on_save db 'SAVE'.
  zebc_wire_up event bl->on_show ui 'SHOW'.
```




```
  zebc_wire_up event ui->on_delete db 'DELETE'.

 ui->call_screen( ).
endmethod.
```

ADventas Consulting

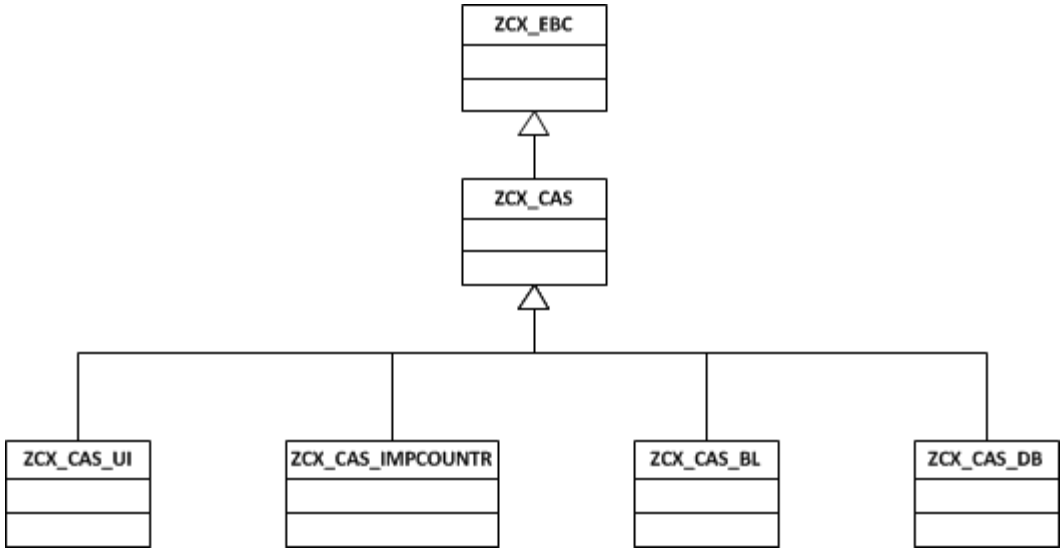
Each user command raises an event, which can have more than one handler. All events have the same signature.

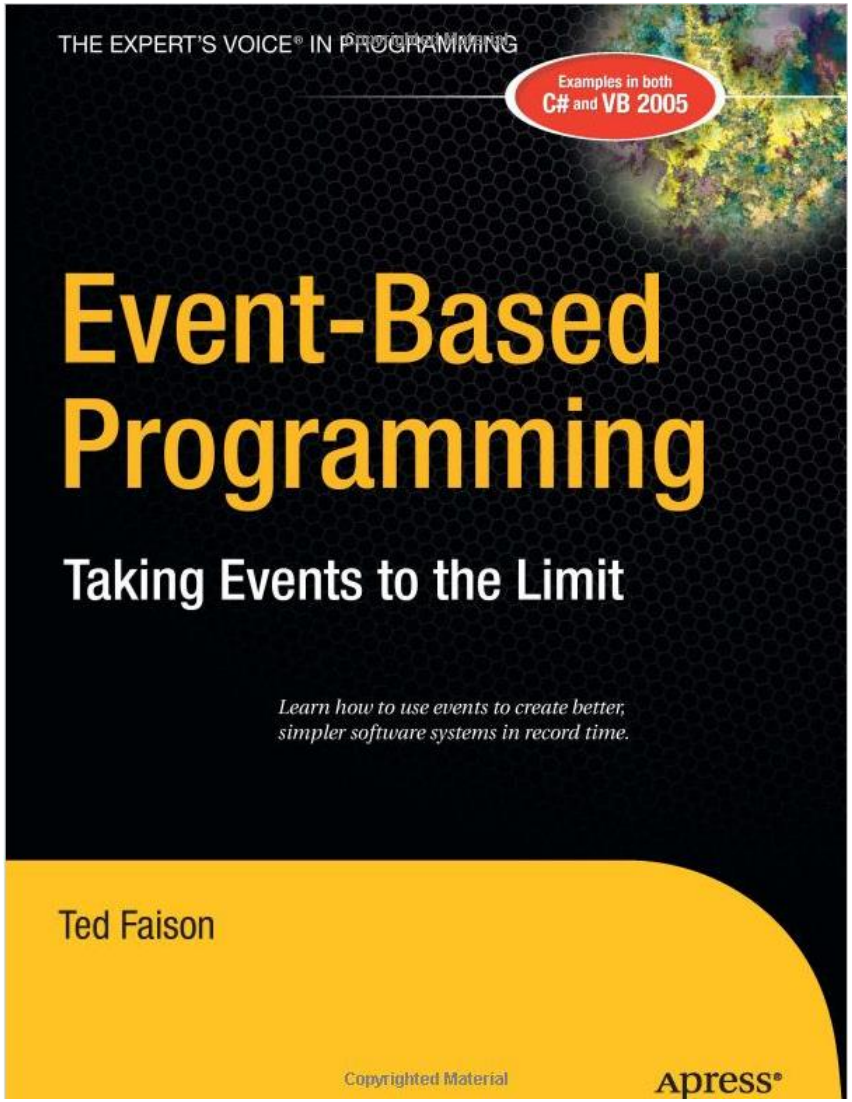
```
module user_command_1000 input.
  try.
      case ok_code.
        when 'ENTER' or 'SHOW'.
          me->raise( me->on_read ).
        when 'SAVE'.
          loop at me->on_save into me->event.
            call method me->event-object->(me->eve
              exporting
                cas = me->cas.
          endloop.
          message s001.
        when 'DELE'.
          loop at me->on_delete into me->event.
            call method me->event-object->(me->eve
              exporting
                cas = me->cas.
          endloop.
        when 'IMPCOUNTR'.
          loop at me->on_impcountr into me->event.
            call method me->event-object->(me->event-method)
              exporting
                cas = me->cas.
          endloop.
        when 'COPY'.
          call transaction 'ZDM25'.
        when others.
      endcase.
    catch zcx_cas into exception.
      text = exception->get_text( ).
      message text type exception->msgty.
  endtry.
endmodule.                    " USER_COMMAND_1000  INPUT
```

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

Methode RAISE aktiv

```
method raise.
  if events is not initial.
    loop at events into event.
      if event-object is not initial and event-method is not initial.
        try.
            call method me->event-object->(me->event-method)
              parameter-table event-parameter.
        endtry.
      endif.
    endloop.
  endif.
endmethod.
```


ZCX_EBC
ZCX_CAS
ZCX_CAS_UI
ZCX_CAS_IMPCOUNTR
ZCX_CAS_BL
ZCX_CAS_DB


THE EXPERT'S VOICE® IN PROGRAMMING
Examples in both
C# and VB 2005
Event-Based Programming
Taking Events to the Limit
Learn how to use events to create better, simpler software systems in record time.
Ted Faison
Copyrighted Material
Apress®

ADventas
Consulting

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