

**specify.** **simplify.**  
**explore.**

with **ComplexValues**

counseling developer **Thomas J. Schrader**

Smalltalked Visuals GmbH **Christian Haider**

data workflows **fragile**,  
systems **complicated**,  
maintenance **difficult**

**?**

conventional OO misses something

Values - functional style  
taps the full potential of OO

„values | objects“ !

[MacLennan, 1982]

Value is **abstract**  
concept

**42**

abstraction  
no lifecycle  
stateless  
context-free

# Why?

## Adequate

- Objects and Values ARE different
- There are Values – why use Objects?

## Solid

- No side effects
- No cycles
- No illegal Values

## Pretty

- Readable at a glance
- Concise
- Understandable

## Practical

- Record and replay
- Print and transportable
- Supported by tools (references, refactoring, formatter, syntax highlighter)
- Trivial

Value is **literal** -  
understand objects  
at a glance

**Time**

**h: 16 m: 30**

# ComplexValue is immutable composite

top  
down  
tree

Text

```
string: 'ComplexValue...'  
style: (Textstyle  
  font: #{Helvetica}  
  size: 60)
```

ComplexValues are  
real objects  
but without identity

Value = behavior + content

same class & content = same Value



# ComplexValue is generated from a specification

aValueClass>>localSpecification

<constant: #... class:  
#{aClass}>

<optional: #... class #{aClass}  
default: aDefaultValue>

<sequence: #...>

<map: #...>

# ComplexValues: configure in Smalltalk

```
Store class>>publicCincom
  ^PostgreSQL
    name: #publicCincom
    source: #psql_public_cst
    environment:
'store.cincom...'
  user: (User
    name: 'guest'
    password: 'guest')
```

# Standard Values

## Immediates

SmallInteger 42 Character \$a

## Literal

Float 13.5 Symbol #none

String 'abc' Array #(1 'xyz' #one)

## Value like

Point 1 @20 Association #abc -> 42

Date Time Rectangle (0@0 extent: 5@5)

ColorValue (ColorValue red: 1 green: 0 blue: 0)

# Complex Values

## ChartText

```
style: (Textstyle
  color: (CmykColor
    cyan: 1
    magenta: 0.3
    yellow: 0
    black: 0.3
    rgb: #[0 101 157])
  font: #{SmallCharts.Helvetica}
  size: 12)
string: 'This is a ' , self name asString
position: 5 @ 10
```

# Defining a Value

localSpecification

<constant: #constant class: #{Symbol}>

<optional: #optional class: #{Symbol} default: '#a'>

<sequence: #array>

<map: #dictionary>

# Constructor

constant: **const** optional: **opt** array: **arr** dictionary: **dict**

| inst |

inst := self new.

inst

initializeConstant: **const**

optional: **opt**

array: **arr**

dictionary: **dict**.

^inst

# Optional Constructors

constant: **const**

| **inst** |

**inst** := **self** new.

**inst** initializeConstant: **const** optional: **nil** array: **nil** dictionary: **nil**.

**^inst**

constant: **const** optional: **opt** (...)

constant: **const** optional: **opt** array: **arr** (...)

constant: **const** optional: **opt** dictionary: **dict** (...)

constant: **const** array: **arr** (...)

constant: **const** array: **arr** dictionary: **dict** (...)

constant: **const** dictionary: **dict** (...)

# Initializer

initializeConstant: **const** optional: **opt** array: **arr** dictionary: **dict**

**constant** := **const**.

(**opt** notNil and: [self optional ~= **opt**]) ifTrue: [  
    **optional** := **opt**].

(**arr** notNil and: [**arr** notEmpty]) ifTrue: [  
    **array** := (**Array** withAll: **arr**) beImmutable].

(**dict** notNil and: [**dict** notEmpty]) ifTrue: [  
    | **od** |  
    **od** := **OrderedDictionary** new.  
    **dict** keysAndValuesDo: [:**key** :**value** | **od** at: **key** put: **value**].  
    **dictionary** := **od** beImmutable].

self beImmutable



# Accessors

constant

"<Symbol>"

**^constant**

optional

"<Symbol>"

**^optional** ifNil: [*#a*]

array

"<Array>"

**^array** ifNil: [*#()*]

dictionary

"<Dictionary>"

**^dictionary** ifNil: []

**Dictionary** new  
beImmutable]

# Printer

printvalueWith: **printer**

| args |

args := **OrderedCollection** new.

args add: (**printer** constant: 'constant' value: self constant).

args add: (**printer** optional: 'optional' value: **optional**).

args add: (**printer** array: 'array' value: self array).

args add: (**printer** dictionary: 'dictionary' value: self dictionary).

**^printer** printvalue: self arguments: args

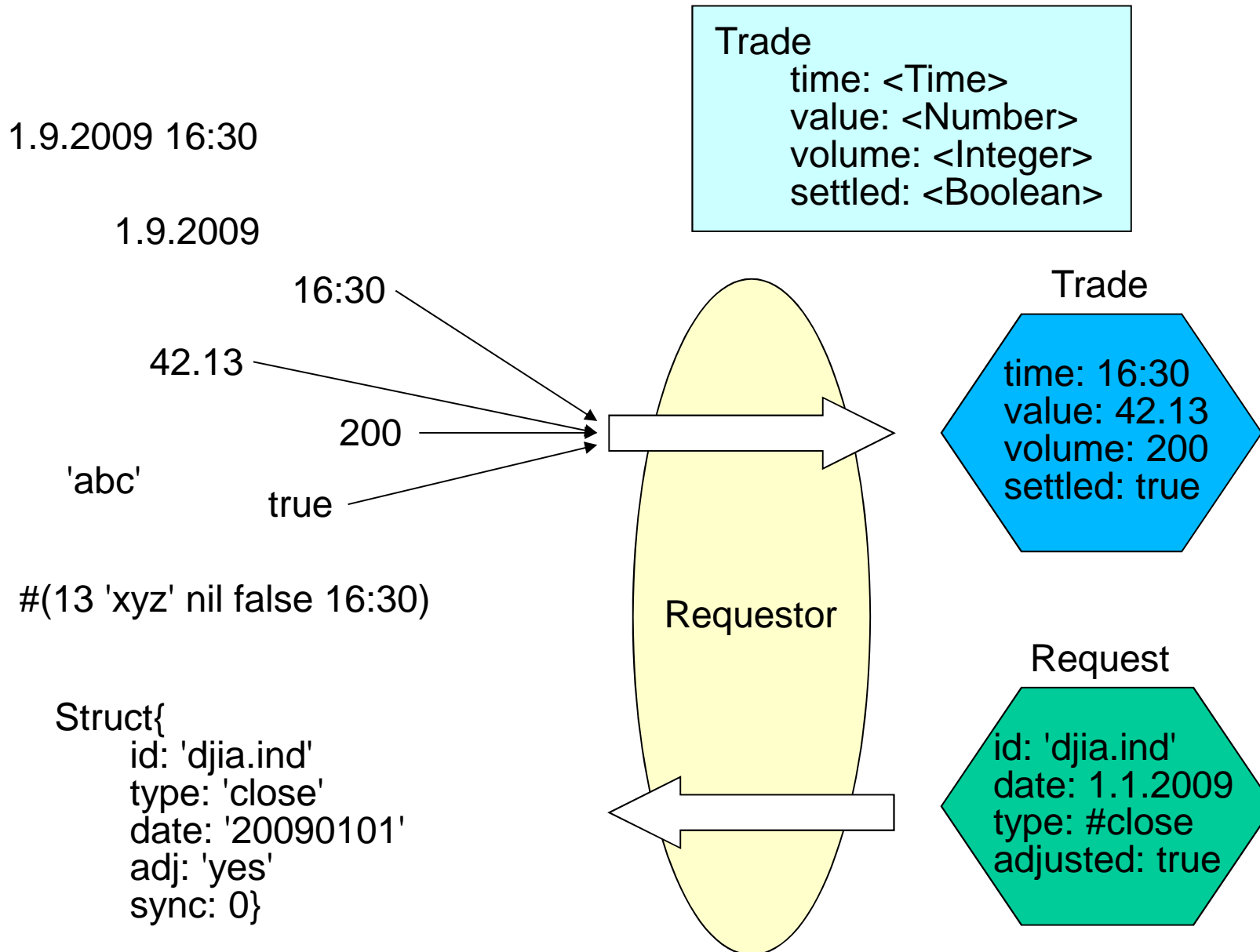
# Opentalk Service

passInstVars

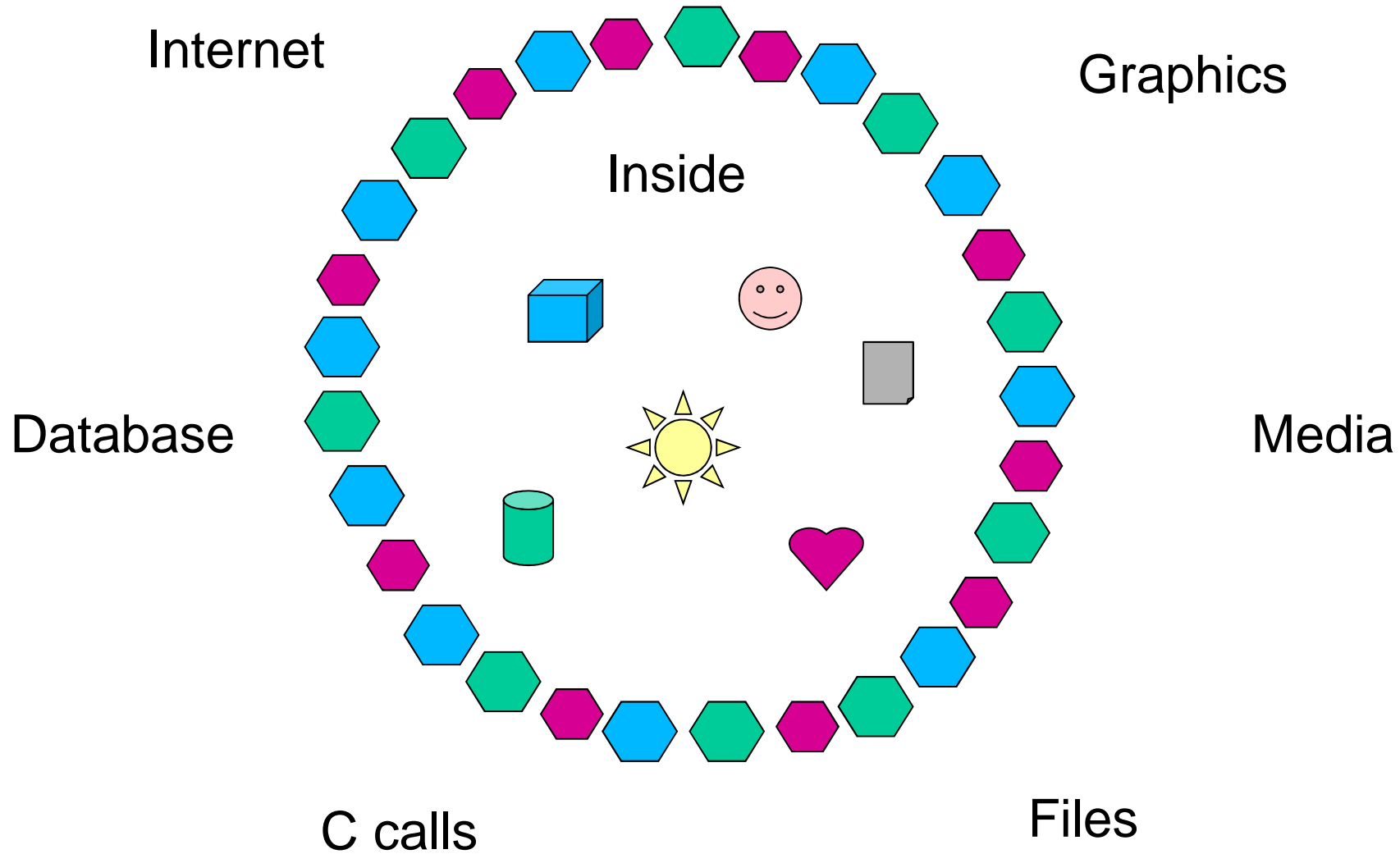
"for OpenTalk StSt"

^#(#default #default #default #value)

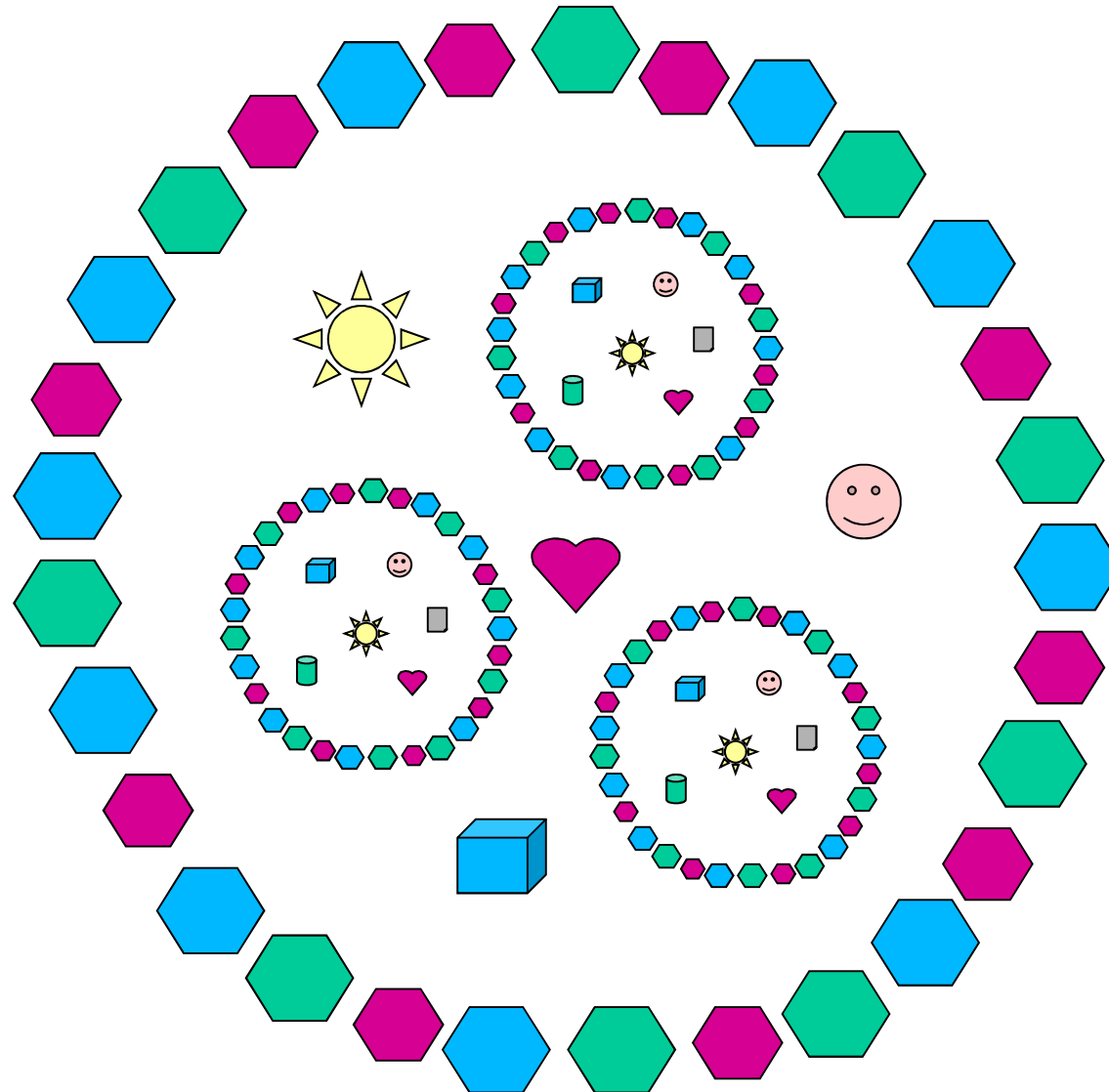
# Interface



# System Interface



# Module Interfaces



# Configuration

```
#{#{UI.FullSpec}
  #window:
  #{#{UI.WindowSpec}
    #label: #{#{Kernel.UserMessage}
      #key: #UnlabeledCanvas ...}
    #bounds: #{#{Graphics.Rectangle} ...} )
  #component:
  #{#{UI.SpecCollection}
    #collection: #(
      #{#{UI.TextEditorSpec}
        #layout: #{#{Graphics.LayoutFrame} ...}
        #name: #textEditor
        #model: #textHolder
        #isReadOnly: true
        #tabRequiresControl: true ) ) ) )
```

## FullSpec

```
window: (WindowSpec
  label: (UserMessage
    key: #UnlabeledCanvas ...)
  bounds: (Rectangle ...))
component: (SpecCollection
  collection: (Array
    with: (TextEditorSpec
      layout: (LayoutFrame ...)
      name: #textEditor
      model: #textHolder
      isReadOnly: true
      tabRequiresControl: true)))
```

# as Value

## FullSpec

```
window: (WindowSpec
  label: (UserMessage
    key: #UnlabeledCanvas
    defaultString: 'Unlabeled Canvas'
    catalogID: #labels)
  bounds: (Rectangle origin: 512@384 corner: 858@635))
component: (SpecCollection collection: (Array with: (TextEditorSpec
  layout: (LayoutFrame
    leftFraction: 0 offset: 10 rightFraction: 1 offset: -10
    topFraction: 0 offset: 10 bottomFraction: 1 offset: -10)
  name: #textEditor
  model: #textHolder
  isReadOnly: true)))
```



# VW Setting

```
<?xml version="1.0"?>  
<settings domain="VisualWorksSettings">  
  <setting>  
    <id>  
      <key>tools</key>  
      <key>browser</key>  
      <key>defaultBrowserType</key>  
    </id>  
    <state>  
      <choice-key>Package</choice-key>  
    </state>  
  </setting>  
</settings>
```

# as Value

## Settings

domain: 'VisualWorksSettings'

setting: (**Id**

with: #tools

with: #browser

with: #defaultBrowserType)

state: (**ChoiceKey** value: 'Package')

# Opentalk

```
(BasicBrokerConfiguration new
  adaptor: (
    ConnectionAdaptorConfiguration new
      isBiDirectional: false;
      processingPolicy: WSProcessingPolicy new;
      transport: (
        {HTTPTransportConfiguration} value new
          marshaler: (
            SOAPMarshalerConfiguration new
              binding: aWsdIBinding;
              yourself)))
  ) newAt: anIPSocketAddress
```

# as Value

(**BasicBrokerConfiguration**

adaptor: (**ConnectionAdaptorConfiguration**

isBiDirectional: **false**

processingPolicy: **WSProcessingPolicy** new

transport: (**HTTPTransportConfiguration**

marshaller: (**SOAPMarshalerConfiguration**

binding: **aWsdIBinding**)))

) newAt: **anIPSocketAddress**