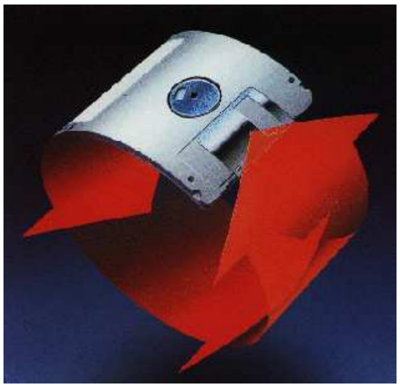
SOUR^{++} or Some Ideas For the Future



Sour'94 — Advanced Techniques for Software Reuse
15–17 December 1994
Braga

J. Nuno Oliveira
University of Minho / INESC (Group 2361)
Braga

email: jno@di.uminho.pt

Only a Few Topics for Discussion

. . . in Antecipated Form. . .

(Why didn't I write my slides yesterday night?)

Main "Lessons" of SOUR

On the Product Line

- The rôle of **Conceptualization** as a kind of specification
- **AOs**: Evolution from files to conceptual models of files, *etc.*, in a *crescendo* of IS building units:

  files →PAOs →LAOs →CLAOs →Containers →Clusters. . .

- **AO-repository**: Evolution from hierarchical file systems to hierarchical semantic networks.
- **Clusters**: sub-hierarchical semantic networks
- **Facets**: strict attributive model is not enough!
- **IQS**: uncertainty tolerant SQL
- **CMOD**: handling fuzzy metrics in AO comparison
- **RM**: a good picture is better than 100 list panes!

On the Process Line

- **Development**: *crescendo* of middle-ware "shields":

  machine $\longrightarrow$DBMS $\longrightarrow$ODBC $\longrightarrow$ERA $\longrightarrow$
  $\longrightarrow$AOENTRY $\longrightarrow$Windows pane $\longrightarrow$end user!

- **APIs**: good old academic abstract data types have survived the myth!
- **Rapid Prototyping**: or the ubiquity of functional specification (by the way: C is a "functional" language)
- **VBX**: or how to convince UNIX-addicted people to believe in Windows!
- **Batch**: or how to obtain AOs from anywhere (never say batch is old-fashioned!)
- **UI**:

$$\begin{cases} \text{input} & \text{— syntax directed edition of batch language} \\ \text{output} & \text{— structural beautifying of observable types} \end{cases}$$

How Can this Experience be Enriched?

Automatic Conceptualization

SOUR please: I need a batch conceptualization of my directory

`/usr/jno/sour/contractology/letters`

I'll be back tomorrow!

AAC — “attempt automatic conceptualization”

Related topics:

- Class oriented
- DOC: textual analysis background
- CODE: Reverse specification
- *etc.*

This deserves a better study

Conceptualization vs Object Modelling

Object Modelling $\left\{ \begin{array}{lll} \text{“exogenous”} & \text{—} & \text{attributes and facets} \\ \text{“endogenous”} & \text{—} & \text{specify abstract syntax/semantics} \end{array} \right.$

Cf good old object templates (archetypes). Example:

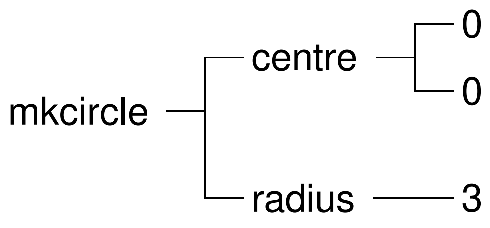
> *record that my circle’s radius is 3cm long and that my circle is centred upon the origin of the XY-plane*

In SOUR’s exogenous conceptualization:

$AO =$

ATTRIBUTES
CLASS: Circle RADIUS: 3 CENTER-X:0 CENTER-Y:0
LINKS
FACETS

In an abstract syntax conceptualization:



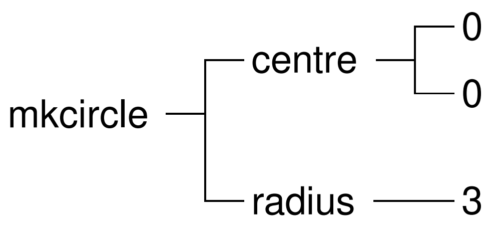
SGML: a (now) popular device for “marketing” syntax trees, something like...

```
<MKCIRCLE>
     <CENTRE>
        0,0
     </CENTRE>
     <RADIUS>
        3
     </RADIUS>
</MKCIRCLE>
```

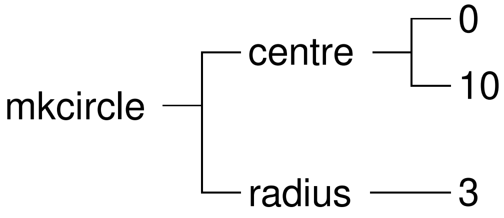
Comparing by Unification

How do you compare the above circle with another one with the same radius but above it 10cm?

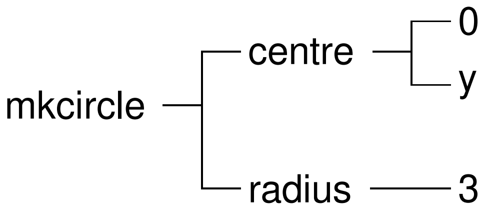
$$A \;=$$



$B =$



$A \cap B =$



or the rôle of formal symbolic expression unification

Related Work

If my "new" problem is to produce an information system for a given company, how can I reuse my repository of company models (*e.g.* ERAs)?

How do I compare ERA diagrams? They are pictures!

Strategy put forward in [*Oliveira & Cruz 93*]:

- Convert ERA $\longrightarrow$ SETS formal notation

  (we have a TOOLBOOK tool for convincing you. . .)

- Compare SETS expressions by unification

  (modulo SETS equivalence, SOUR'93 slides please . . . .)

- Convert result expression back to ERA pictures

In his final year project, Cruz has compared (via SETS) 21 ERA specifications of ISs and found a “bit” of redundant work…

Cluster Comparison

Very interesting, kind of “pushout” mechanism, but time consuming…

IQS assisted by CMOD

IQS can be fully assisted by CMOD in query evaluation/optimization. In fact, *every IQS query “is” an AO*, *cf.*

```
GET ALL DOC AUTHOR=BRUNIALTI, YEAR=1994
```

becomes

ATTRIBUTES
CLASS: DOC AUTHOR:Brunialti YEAR:1994
LINKS
FACETS

Why not saving the query-AO in the AO-lattice? Benefits:

- The next time the same query is parsed, its evaluation is replaced by a fast inspection of the AO-lattice
- The mechanism is self-optimizing: when evaluating

  $$A \wedge B$$

  in the case that $A \Rightarrow B$, its possible to deduce from the AO-lattice that $A \leq B$ and not evaluate B at all

- The mechanism is self-tuning: the user's "area of interest" becomes a more and more specialized zone of the repository, speeding up searching and adding to user-friendliness
- Complex queries can be represented by clusters and optimized by cluster comparison
- Space cost — ok, I know — but why not try it? Disk space is cheaper than decision makers time

Coping with Refinement

Or AO-representation, replaceability. In SOUR:

$$\left\{\begin{array}{lcl} \text{specification} & — & \text{conceptualization} \\ \text{classification} & — & \text{comparison} \\ \text{refinement} & — & \text{? (but can be modelled by links)} \end{array}\right.$$

Typical situation: Why do you always have to fill forms in MACINTOSH Word if you don't like it?

- Document specification: abstract syntax format in SGML
- Document comparison and classification: by SGML-implemented unification
- Document implementation: representation maps converting SGML formats to RTF, EXCEL, TEX, TIFF, EPS, *etc. etc.*
- Document interaction: back to archetype oriented user interfaces [*Martins & Oliveira 90*]
- Document abstraction: parsers mapping back to standard SGML
- Document interchange: user preferences tolerant network systems

References

[Martins & Oliveira 90] F.M. Martins, J.N. Oliveira *Archetype-Oriented User Interfaces*. Computer & Graphics, 1990, 14(1), 17–28.

[Oliveira & Cruz 93] Oliveira J.N., Cruz A.M. *Formal Calculi Applied to Software Component Knowledge Elicitation*. Technical Report C19-WP2d, Project *C.1.9. Sviluppo di Metodologie, Sistemi e Servizi Innovativi in Rete*, December 1993.