

Quality of Component-Based Software Development

Miguel Goulão

Faculdade de Ciências e Tecnologia
Universidade Nova de Lisboa

miguel.goulao@di.fct.unl.pt

<http://ctp.di.fct.unl.pt/QUASAR>



Summary

- Motivation
- Some notions on software components
- An Architecture for Component Based Software Engineering
- Conclusions

The shift to Software Components (a matter of self-respect?)

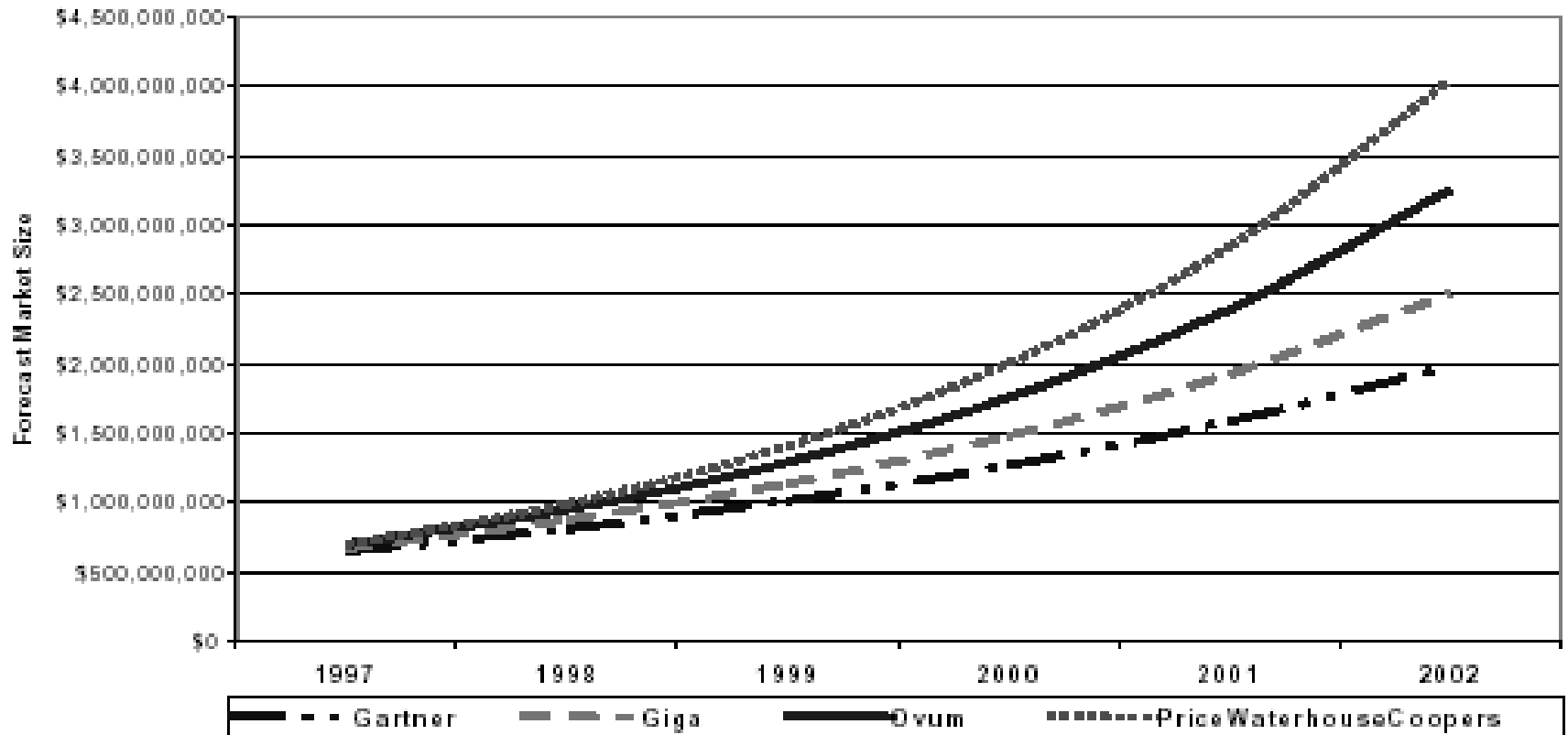
- *The onset of server-side components is inevitable; nearly every self-respecting application server vendor is planning to incorporate a component model in its product. The added abstraction and control that component models will bring will be a certain productivity benefit. [Gartner 98]*

The shift to Software Components (a trend?)

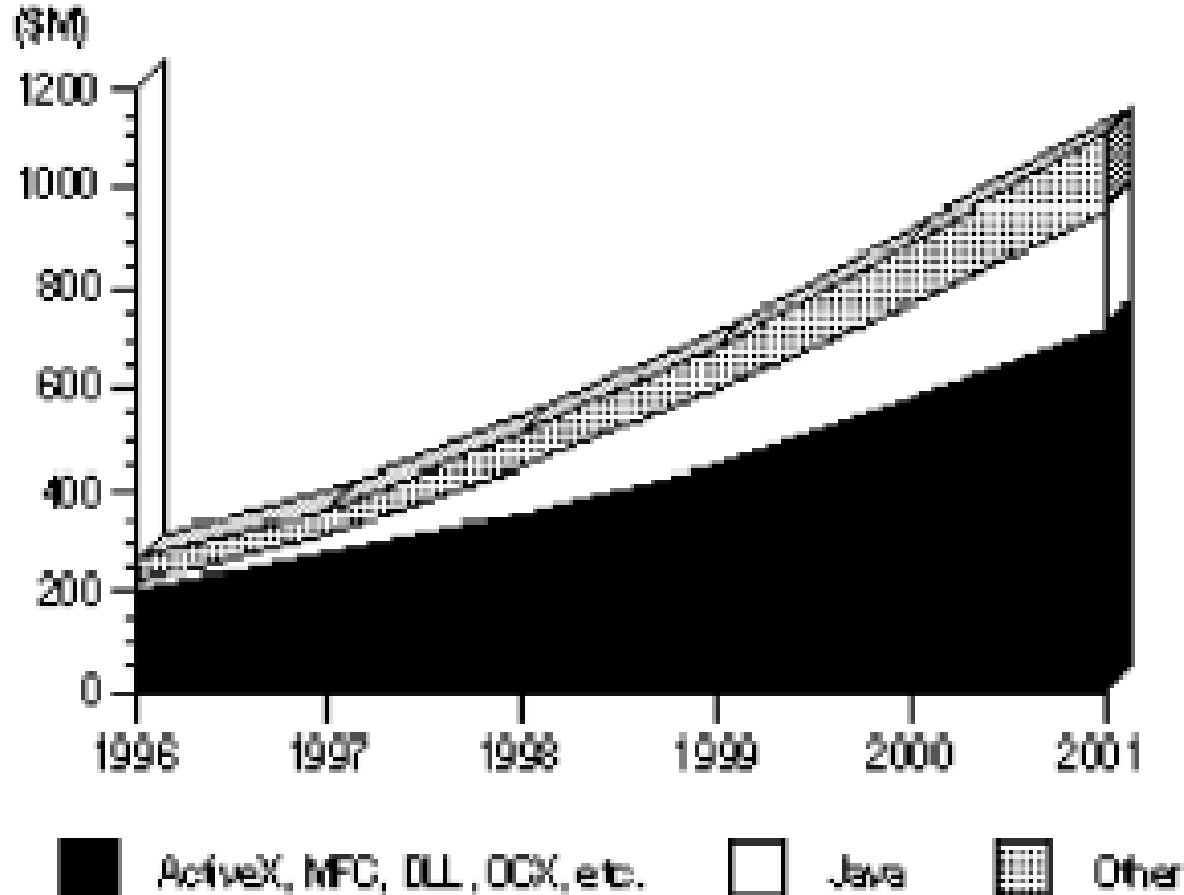
- *By the year 2002, 70 percent of all new applications will be deployed using component-based application building blocks (0.8 probability). [Gartner 00]*
- *By 2001, at least 60 percent of EC applications will be built using components (0.7 probability). [Gartner 98]*

The shift to Software Components (an inevitable trend?)

Research Firms' Forecast Size of Software Components Market

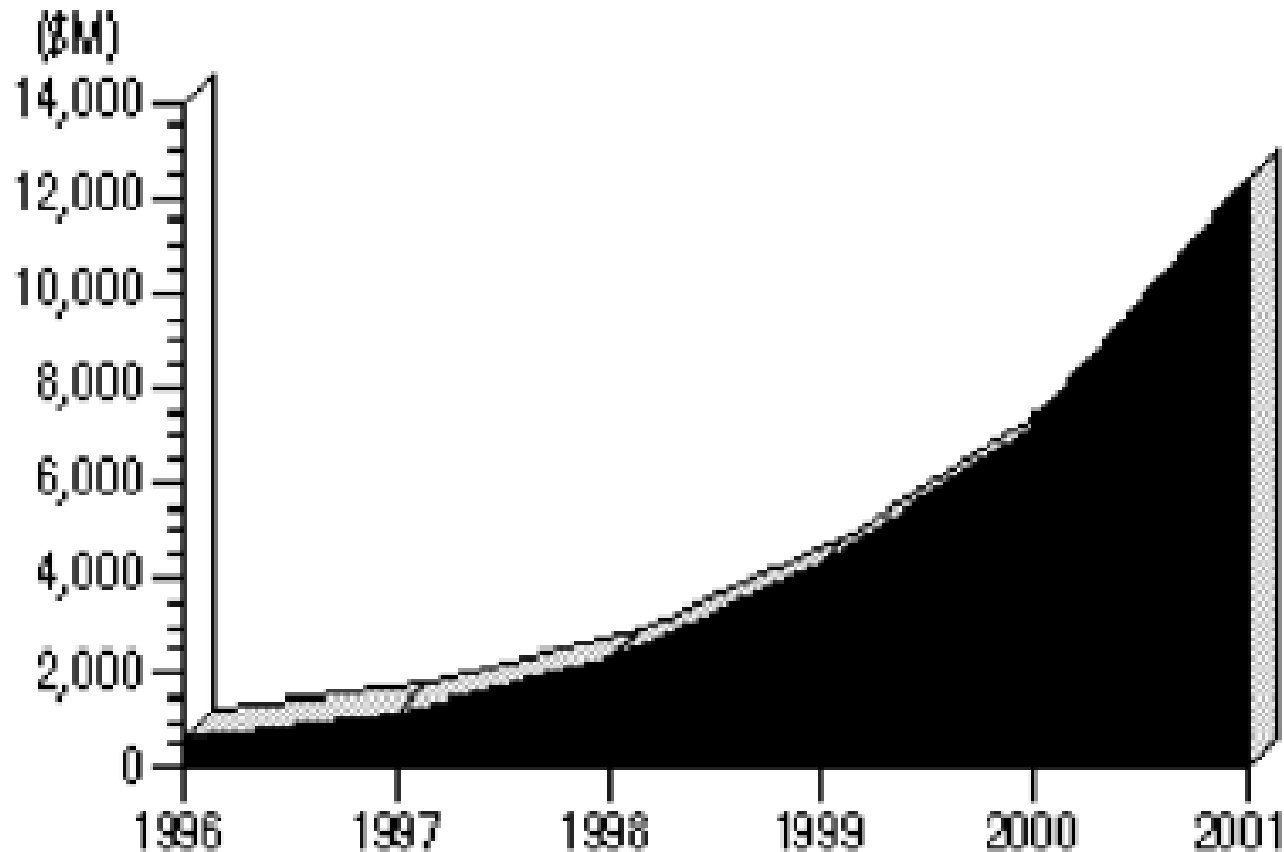


Component technology players



Source: International Data Corporation, © 1998

Component based development market





Component market niches

- Components as commodities or product lines
- Infrastructure
- Component Platforms and Application Servers
- Component tools
- Consulting/Integrator
- Brokerage
- Third party component certification



A competitiveness issue...

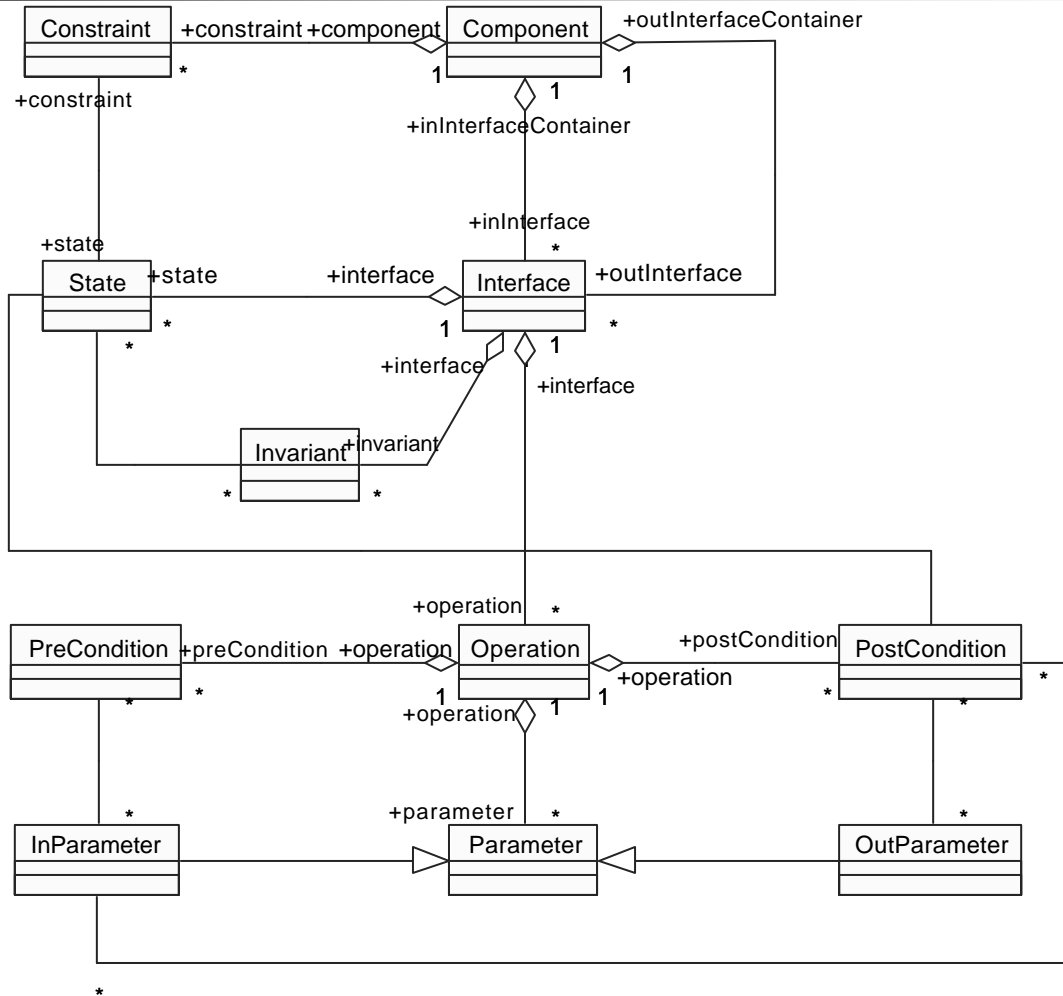
- Improve programmer productivity and reduce time to market for IT resources.
- Produce systems that are flexible enough to withstand technology and business changes.
- Produce systems that deliver excellent performance and are scalable, secure, and robust.



What is a software component?

- An opaque implementation of functionality
- Subject to third party composition
- Conformant to a component model

Component semantics



Example of component contract

```
(* adapted from [Szipersky 98] *)
DEFINITION Directory; (* introduce a new interface type *)
IMPORT Files; (* import an existing interface type *)
IMPORT Notifier; (* interface type for callbacks *)
TYPE Name=ARRAY OF CHAR;

PROCEDURE Init();
(* PRE True *)
(* POST Init=True *)

PROCEDURE ThisFile(n:Name):Files.File;
(* PRE Init=True AND n/="" *)
(* POST result=File named n OR result=NIL AND no such file *)

PROCEDURE AddEntry(n:Name; f:Files.File);
(* PRE Init=True AND n/="" and f=/NIL *)
(* POST ThisFile(n)=f AND FORALL Notifiers CB: CB.Notify(n) *)

PROCEDURE RegisterNotifier(CB:Notifier)
(* PRE CB/=NIL *)
...other details deleted
END Directory;
```

Component Based Development (CBD) claims:

- Reduces development costs
- Reduces development time
- **Increases overall software quality**

- How can these claims be checked?
 - CBD process evaluation
 - **CBD products evaluation**

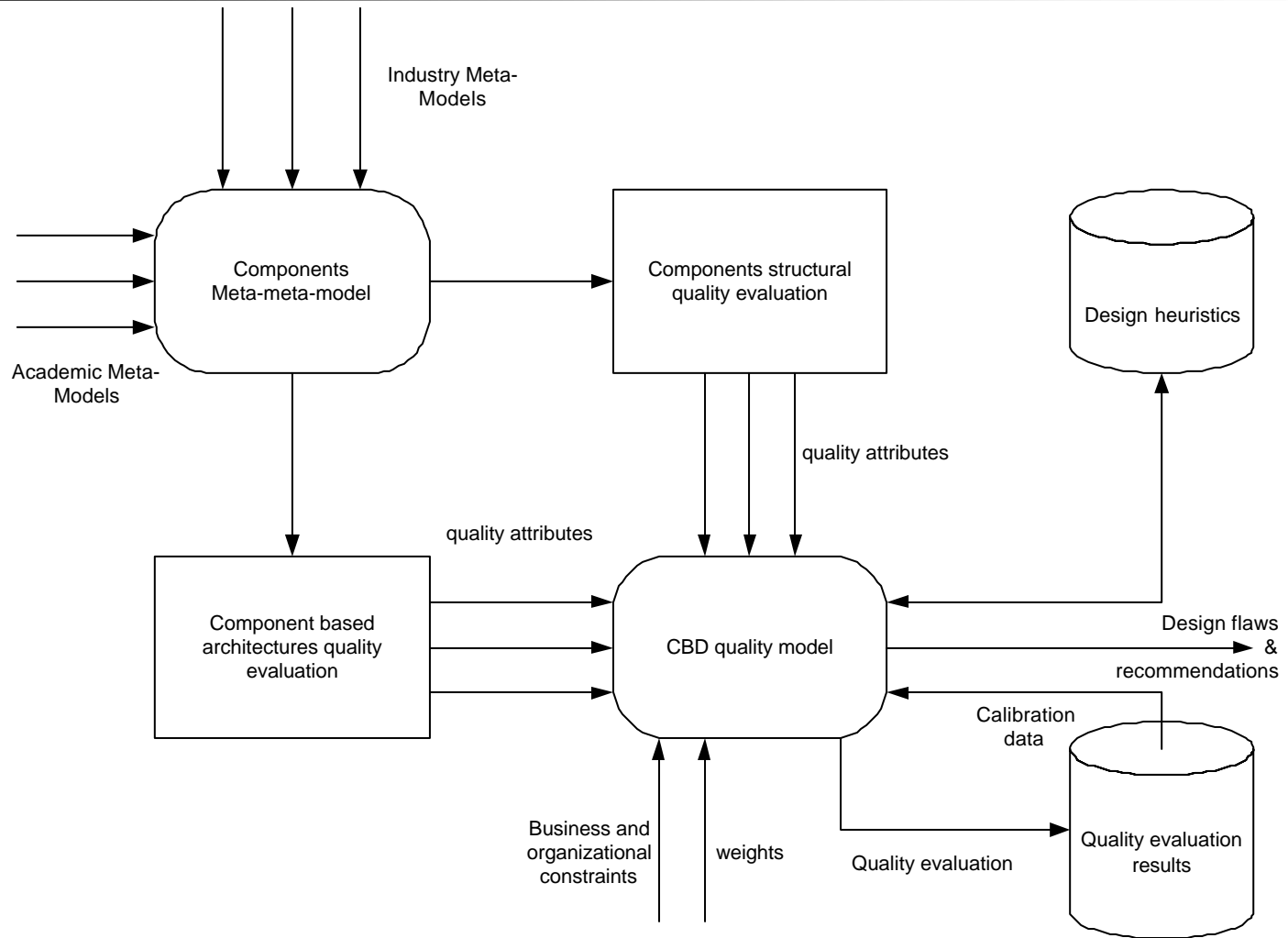
Component based software engineering (CBSE)

- Studies the problems involved with CBD:
 - Both components and component models must be evaluated
 - Final system properties to be previewed from the properties of the software components
- Why is it different when compared to “traditional software”?

Some component peculiarities (example: testing)

	Component provider	Component buyer	Certification body
White-box testing	Yes	No	Yes(?)
Black-box testing	Yes	Yes	Yes
System-level fault injection	No	Yes	No
Operational system testing	No	Yes	No

An architecture for CBSE



Software components meta- meta-models

	Generic concepts	Generic representation
Component meta-meta-model	Yes	Yes
Component meta-model	No	Yes
Component model	No	No
Component		
Component instance		

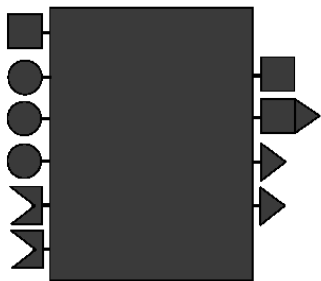


Visualization of components

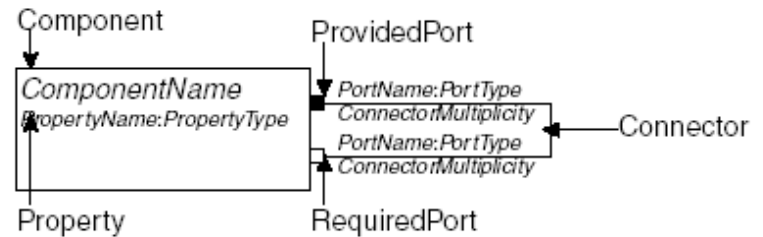
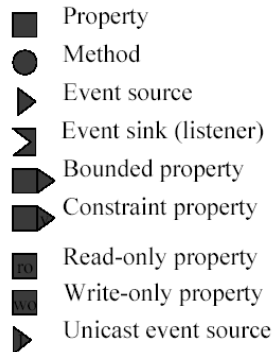
- Using a graphical notation is useful
 - Allows software engineers to communicate at a conceptual level, using the visual metaphors to convey technical concepts
 - Provides an intuitive view of the role of a component in a specific component model
 - Helps to understand the component model itself by providing concrete examples
- But...

The set of concepts varies with each component technology

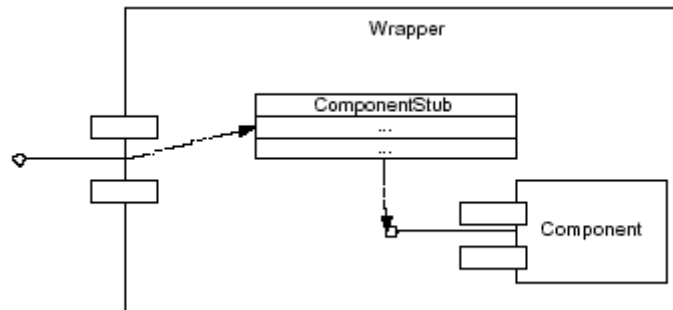
- Variety of notations at different levels of abstraction and with different levels of detail... No generally accepted standards.



Beanome



Rastofor meta-model



UML component diagrams

Quantitative Evaluation of Software Components

- Open research area
- Need for new metrics on component complexity
- Metrics need to be defined in a formal way, but this cannot hamper the ability of software engineers to understand their definition
(formality+understandability)

Software Components Quality Modelling

- Old quality models are not adequate
- Which attributes should be measured?
- How can they be measured?
- Who is responsible for measuring them?
- When should they be measured?
- What is the suitable weight for each attribute?
- How general should the model be?



Conclusions

- Components have a considerable market
- Most research focus on the research has been devoted to functionality and composability
- Lack of generally accepted best practices in component modeling
- Analyzability is an open issue
- Research on components quality evaluation is an emerging area

More on this subject...

- You are welcome to visit our website:
<http://ctp.di.fct.unl.pt/QUASAR>
- Or, you can reach us by email:
miguel.goulao@di.fct.unl.pt
- Questions? Comments?