

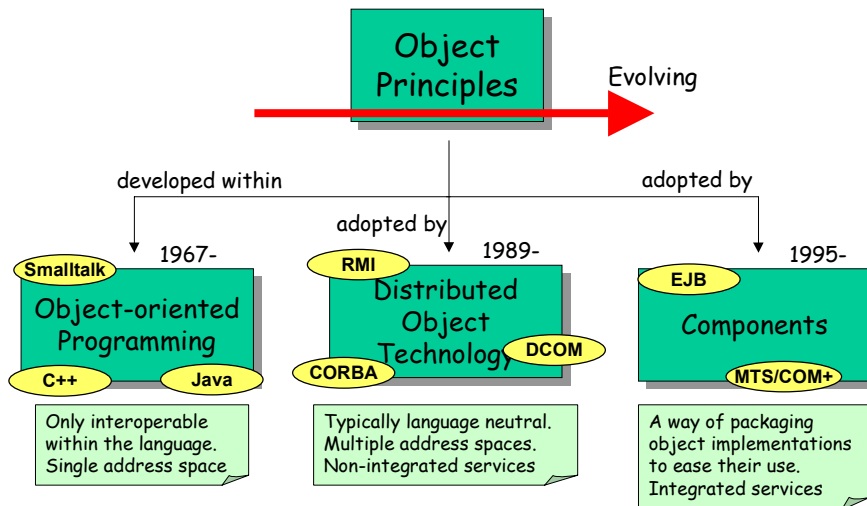
Objects and Components: What's the difference?

John Daniels
Syntropy Limited
john@syntropy.co.uk

Agenda

- Components in context
- Object principles
- Component principles
- Components vs. objects
- Examples
- Modelling components with UML

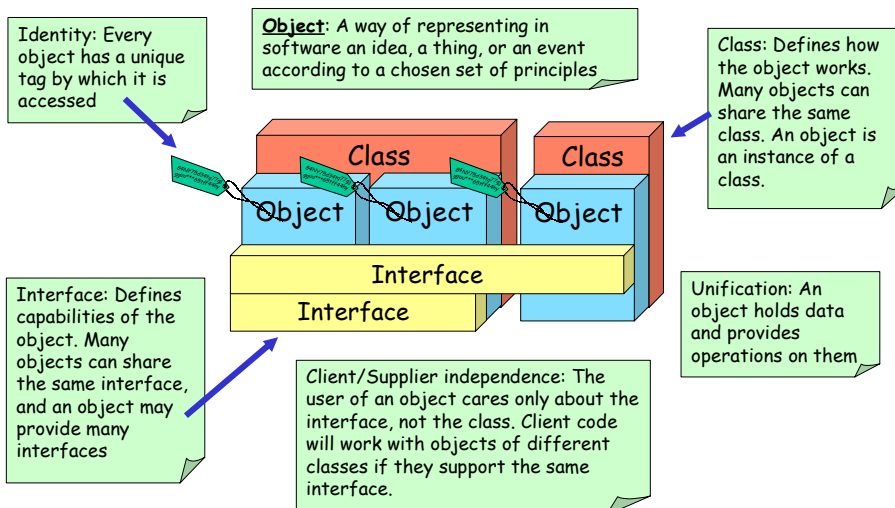
Components in context



email:john@syntropy.co.uk

Syntropy Limited

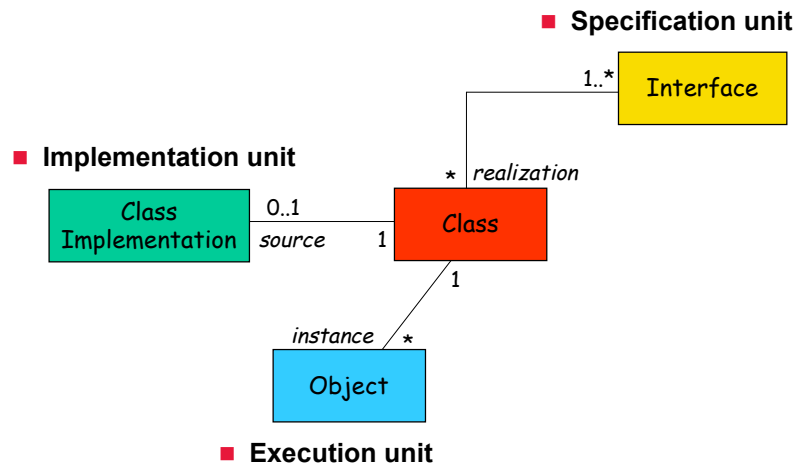
Object principles



email:john@syntropy.co.uk

Syntropy Limited

What is an Object ?



email:john@syntropy.co.uk

Syntropy Limited

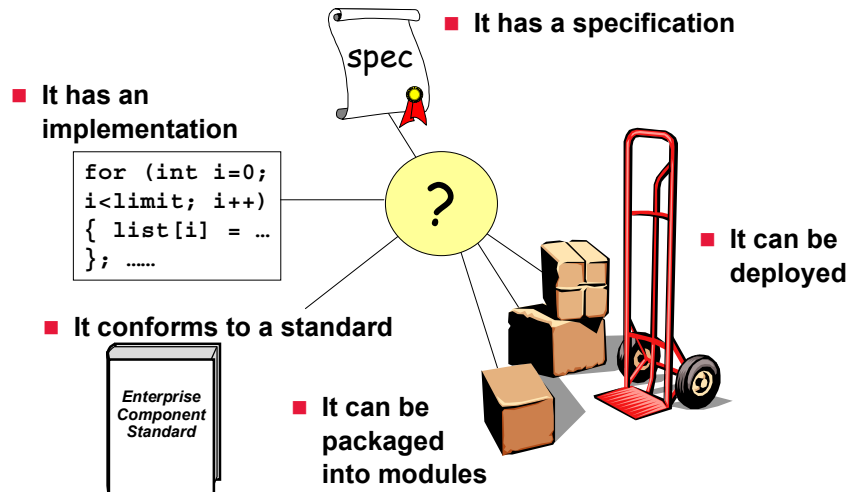
Component principles

- **Component** is not a well-defined term
- Many people believe they are “doing components” but they are doing many different things
- Ranges from loose notion of “useful package” (c.f. UML) to fully standardised plug-in part
- I’m using **component** to mean something quite specific: a piece of software conforming to a defined component standard

email:john@syntropy.co.uk

Syntropy Limited

Aspects of a component



email:john@syntropy.co.uk

Syntropy Limited

Industry component standards

Microsoft®
COM+ *Set to become the dominant standard in the NT world*



Enterprise JavaBeans

The emerging "corporate" standard

 **CORBA Components**

EJB without the Java?

 **CORBA & DCOM for the plumbing**

email:john@syntropy.co.uk

Syntropy Limited

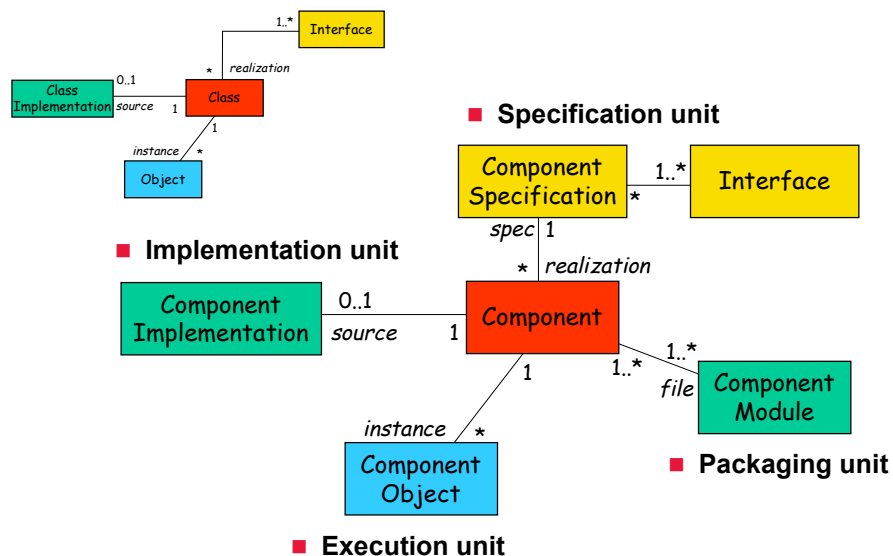
Component standard features

- **Component Model:**
 - defined set of services that support the software
 - set of rules that must be obeyed in order to take advantage of the services
- Simple programming model, no need to design/know about the infrastructure
- Services include:
 - remote access, transactions, persistent storage, security
 - typically use services by configuring not programming

email:john@syntropy.co.uk

Syntropy Limited

What is a Component ?

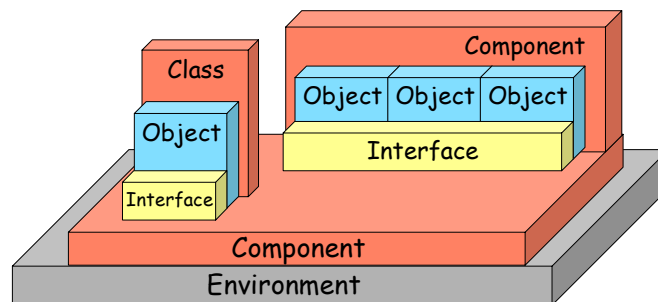


email:john@syntropy.co.uk

Syntropy Limited

Components and Classes

- A component is a special kind of class
 - Has hooks to connect into an environment
 - Can provide other classes (dependent on standard)
- The component (with any provided classes) is the unit of deployment and replacement

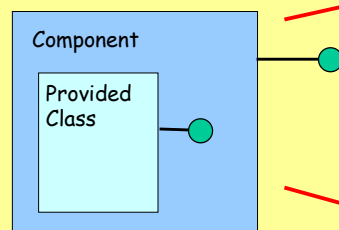


email:john@syntropy.co.uk

Syntropy Limited

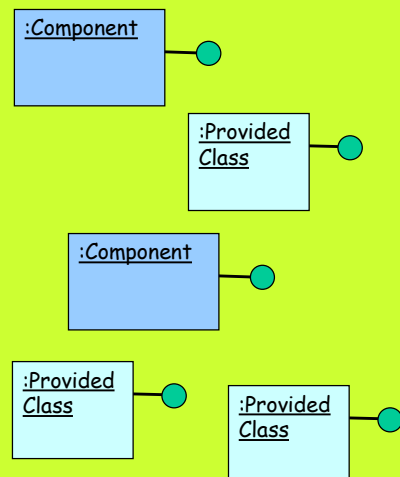
Compile-time and run-time

Compile-time



At run-time we can create many independent instances of the component and its provided classes

Run-time



email:john@syntropy.co.uk

Syntropy Limited

Interfaces vs Component Specs

Interface

- Represents the **usage** contract
- Provides a list of operations
- Defines an underlying logical information model specific to the interface
- Specifies how operations affect or rely on the information model
- Describes local effects only

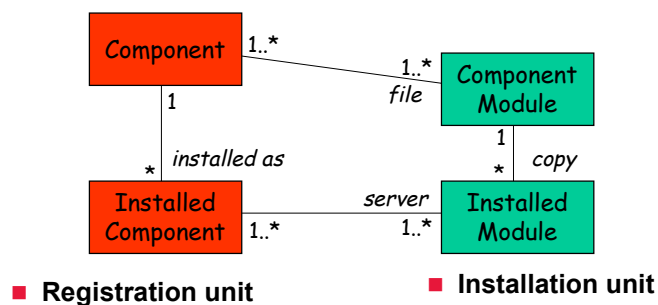
Component Specification

- Represents the **realization** contract
- Provides a list of supported interfaces
- Defines the run-time unit
- Defines the relationships between the information models of different interfaces
- Specifies how operations should be implemented in terms of usage of other interfaces

email:john@syntropy.co.uk

Syntropy Limited

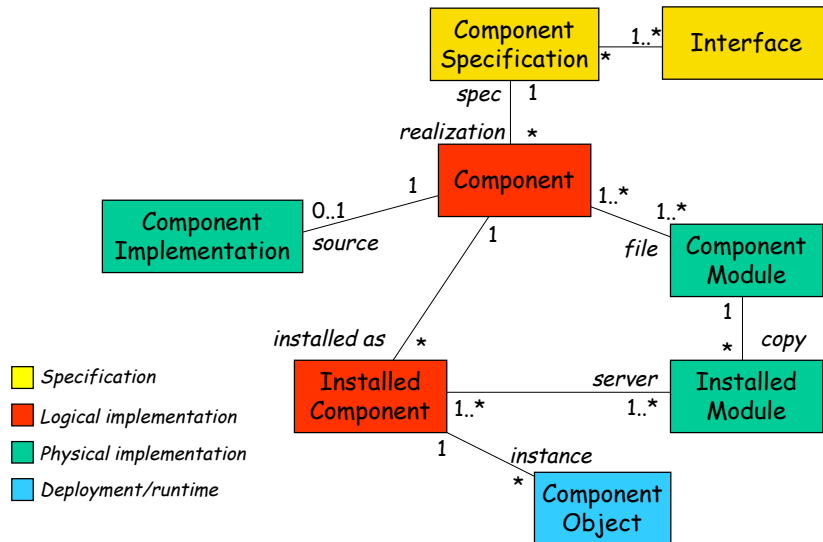
Component deployment



email:john@syntropy.co.uk

Syntropy Limited

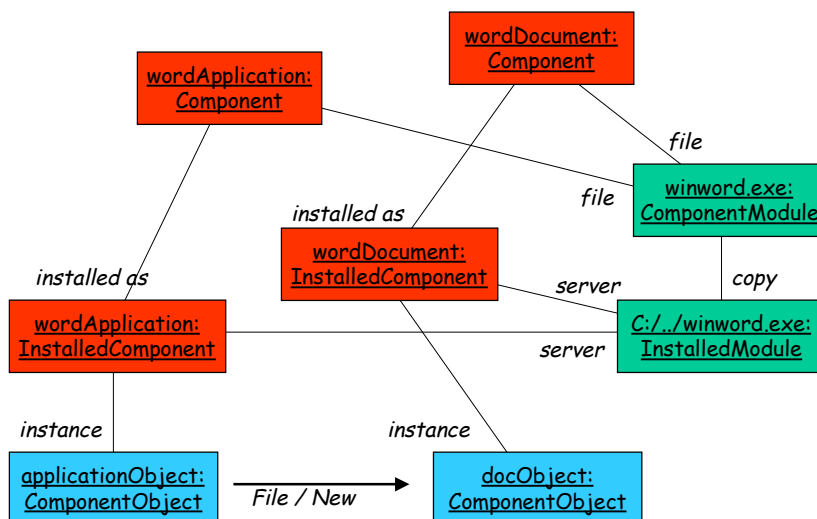
Concept summary



email:john@syntropy.co.uk

Syntropy Limited

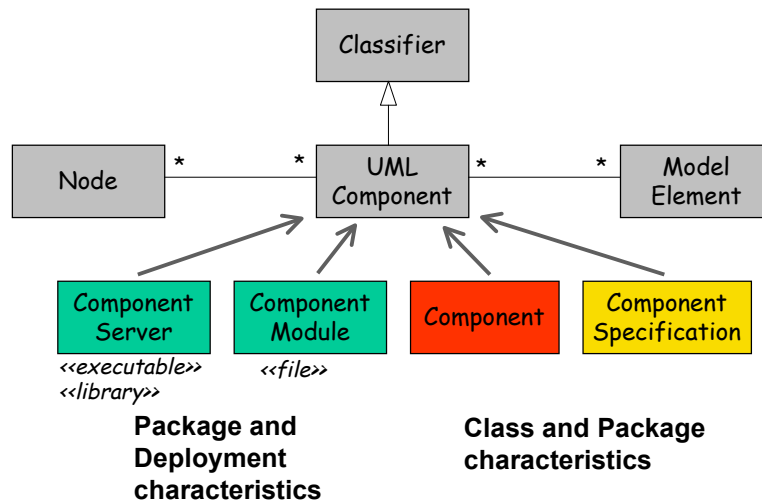
Example - Microsoft Word



email:john@syntropy.co.uk

Syntropy Limited

UML Component (v1.3)



email:john@syntropy.co.uk

Syntropy Limited

UML

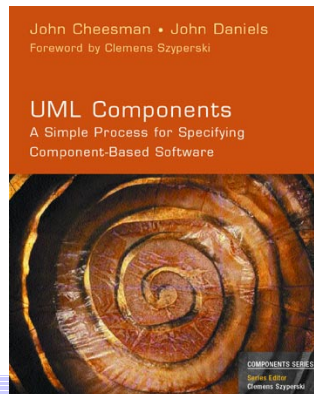
- UML has very loose notion of component and interface
- It isn't possible to express all the subtleties we want without extending UML
- We need an OMG initiative to define a "UML Profile" for components

email:john@syntropy.co.uk

Syntropy Limited

Want to know more?

- UML Components by John Cheesman and John Daniels, Addison-Wesley
- <http://www.umlcomponents.com>



email: john@syntropy.co.uk

Syntropy Limited