# "Inter-component communication as a vehicle towards end-user modeling"

**Computer Technology Institute**
**Patras, Greece**

Manolis Koutlis (koutlis@cti.gr)

Petros Kourouniotis (P.Kourouniotis@asyk.ase.gr)

Kriton Kyrimis (kyrimis@cti.gr)

☞Nikolina Renieri (renieri@cti.gr)

# The starting point

**educational software**

*… teacher-driven requirements*

$\Rightarrow$ they should be able to

- construct <u>their own</u> courses (teaching style, intended audience,…)
- think and build using modeling concepts <u>close to</u> their domain of concern
- <u>exchange</u> educational material with colleagues

$\Rightarrow$ they should not have to

- become programmers!

**…software components instead of applications**

# Components as building blocks of (educational) applications

**software components**
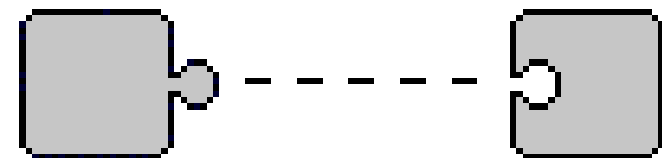
*… having a self-contained functionality*

*… able to inter-operate*

*… available as building blocks for construction of specific scenaria*
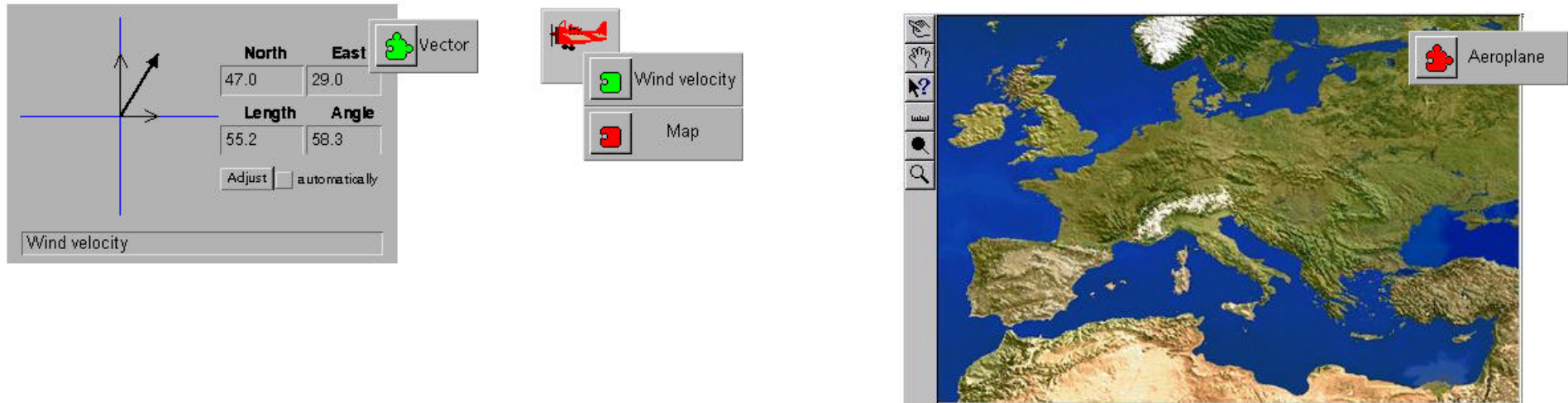
**…inter-component communication mechanism**

**the "plug" metaphor**

- components have plugs (one or more)
- components with <u>matching</u> plugs can be connected

# The communication mechanism (user view)



| **Components have** | **Plugs have** | **Matching plugs have** | **Connectable components have** |
|---|---|---|---|
| • one or more plugs | • shape<br>• colour<br>• name | • shapes that <u>fit</u> <u>together</u><br>• the <u>same</u> colour | • one or more <u>matching</u> plugs |

# The communication mechanism (implementation description)

**Plugs are implemented as**

$\Rightarrow$ Shared objects
  … communication via *data sharing*

$\Rightarrow$ Protocols
  *… communication via a well-defined set of methods*

# Implementation description - Shared objects

## A) Shared objects

- behave as if they were part of all components participating in the connection

- inform all connected components whenever their value changes

*example:*
- components: wind velocity vector, aeroplane
- shared object: wind velocity

*implementation details (Java "Event" mechanism):*
- the "owner" (creator) of the shared object
- the "listeners" of the value changes

# Implementation description - Protocols

## B) Protocols

- Each plug P corresponds to the specification of a set of methods
- Each component must implement the methods specified by the component to which it connects

*example:*

- components:
  joystick, spaceship
- protocol:
  [fire(), turn(direction)],
  [fire_done()]
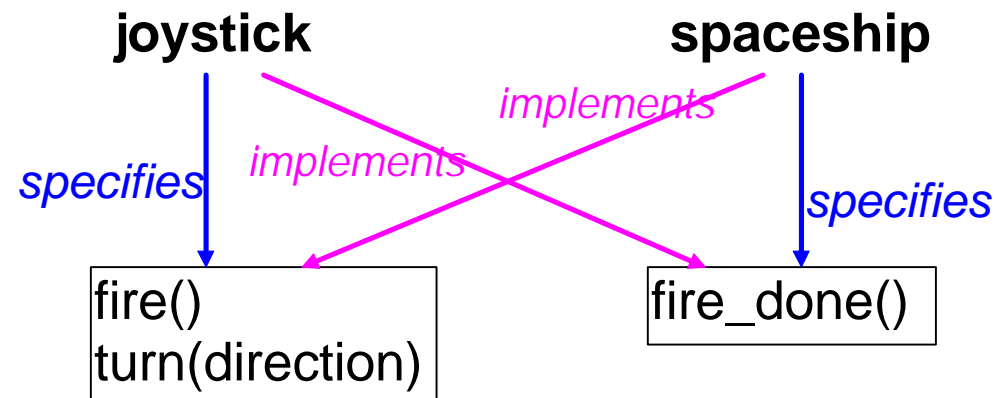
*implementation details:*

- Java "Interface" concept

**joystick**          **spaceship**

*implements*

*specifies*    *implements*    *specifies*

fire()
turn(direction)          fire_done()

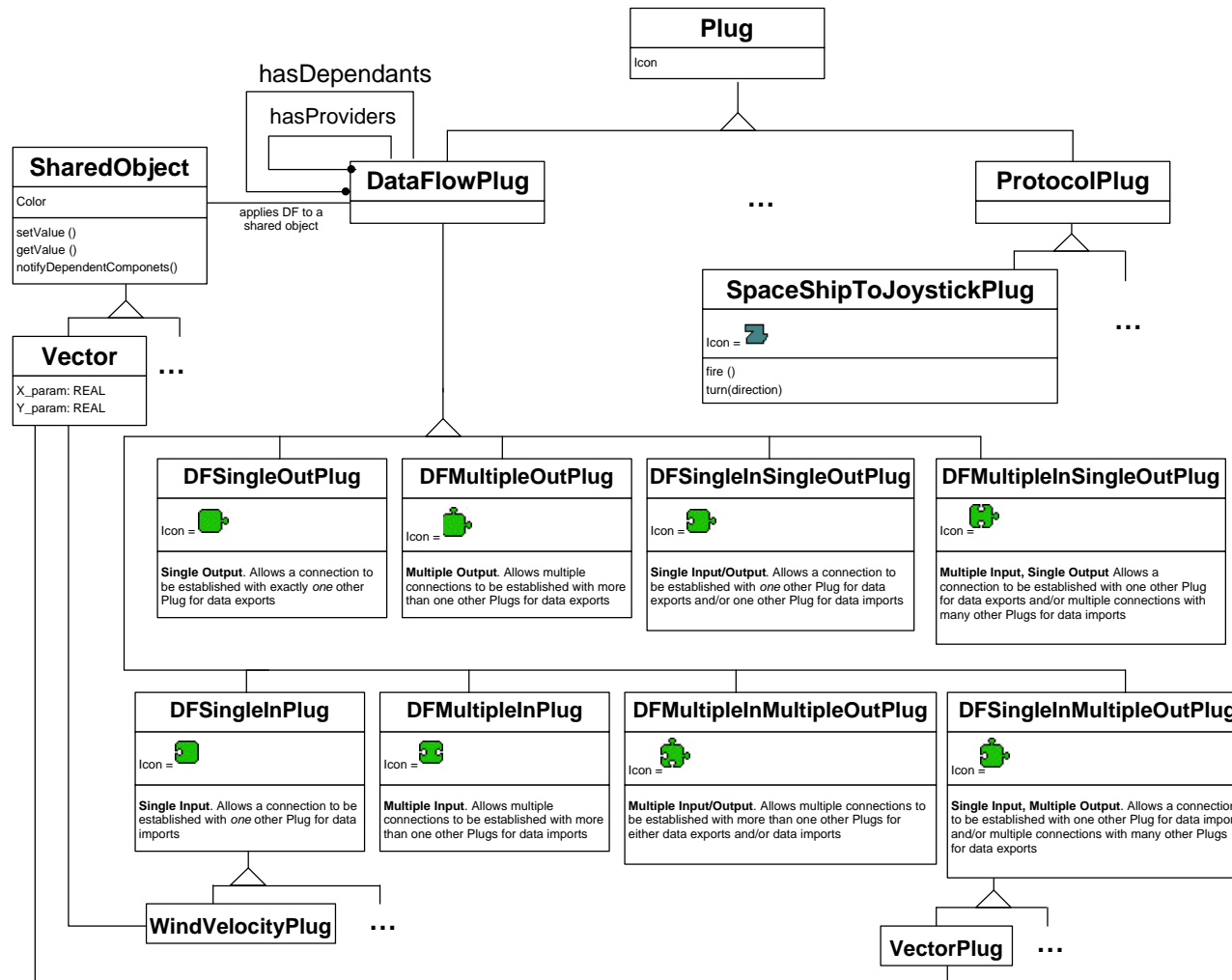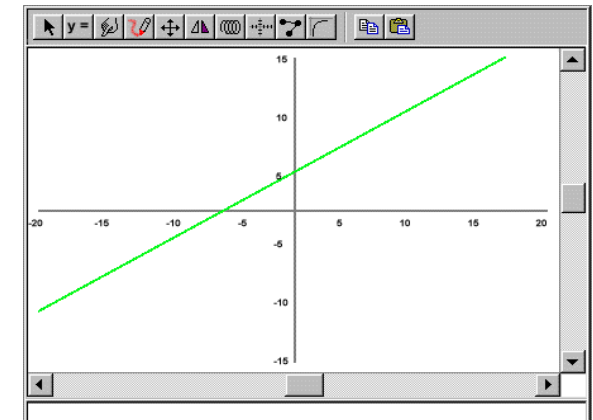# OMT description of the plug mechanism

**Plug**

Icon

hasDependants

hasProviders

**SharedObject**

Color

setValue ()
getValue ()
notifyDependentComponets()

applies DF to a
shared object

**DataFlowPlug**

...

**ProtocolPlug**

**SpaceShipToJoystickPlug**

Icon =

fire ()
turn(direction)

**Vector**

X_param: REAL
Y_param: REAL

...

...

**DFSingleOutPlug**

Icon =

**Single Output**. Allows a connection to
be established with exactly *one* other
Plug for data exports

**DFMultipleOutPlug**

Icon =

**Multiple Output**. Allows multiple
connections to be established with more
than one other Plugs for data exports

**DFSingleInSingleOutPlug**

Icon =

**Single Input/Output**. Allows a connection to
be established with *one* other Plug for data
exports and/or one other Plug for data imports

**DFMultipleInSingleOutPlug**

Icon =

**Multiple Input, Single Output** Allows a
connection to be established with one other Plug
for data exports and/or multiple connections with
many other Plugs for data imports

**DFSingleInPlug**

Icon =

**Single Input**. Allows a connection to be
established with *one* other Plug for data
imports

**DFMultipleInPlug**

Icon =

**Multiple Input**. Allows multiple
connections to be established with more
than one other Plugs for data imports

**DFMultipleInMultipleOutPlug**

Icon =

**Multiple Input/Output**. Allows multiple connections to
be established with more than one other Plugs for
either data exports and/or data imports

**DFSingleInMultipleOutPlug**

Icon =

**Single Input, Multiple Output**. Allows a connection
to be established with one other Plug for data imports
and/or multiple connections with many other Plugs
for data exports

**WindVelocityPlug** ...

**VectorPlug** ...

# Educational component examples



| Country | Icon | Population | Capital | Language | Time difference | EEC member | Country History |
|---------|------|-----------|---------|----------|-----------------|------------|-----------------|
| Ireland | | 3.600.000 | Dublin | English | 00:00 | True | http://.../Ireland.html |
| Greece | | 10.500.000 | Athens | Greek | 02:00 | True | http://.../Greece.html |
| Italy | | 57.700.000 | Rome | Italian | 01:00 | True | http://.../Italy.html |
| Czech | | 9.000.000 | Prague | Czechish | 02:00 | False | http://.../Czech.html |
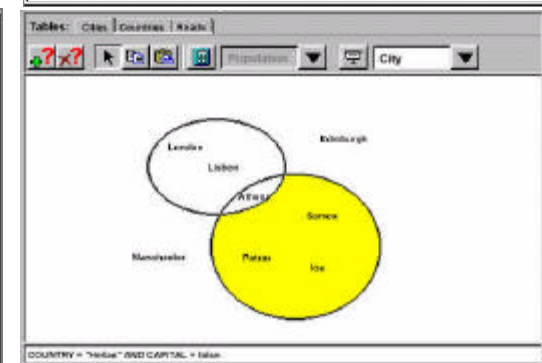
# Related projects

- Project "YDEES", 1995–98   *http://www.cti.gr/RD3/EduTech/ydees.html*
  "The computer as a tool for exploration, expression of ideas and communication for everyone in the school"
Funded by the European Community Support Framework II (Greek Ministry of Industry Energy and Technology)

- Project IMEL, 1996–98   *http://www.cti.gr/RD3/EduTech/IMEL.htm*
  "Intercultural Microworld courseware for Exploratory Learning"
  Funded by the European Union's SOCRATES programme.

- Project "ODYSSEAS", 1996–99,
  **http://odysseia.cti.gr/odysseas/english/ukabout.html**
  "Integrated Network of School and Educational Regeneration in Achaia, Thrace and the Aegean"
  Funded by the European Community Support Framework II (Greek Ministry of National Education and Religious Affairs)