Design of an Informatics System to Bridge the Gap between Using and Understanding in Informatics (Ponto)

Christiane Borchel ¹  Ludger Humbert ²  Martin Reinertz ¹

¹Didaktik der Informatik - Universitaet Dortmund
²Fachgruppe Informatik - Willy-Brandt-Gesamtschule Bergkamen

March 31st, 2005
Introduction and Basics
- Motivation
- Basic informatics instruction in Bavaria
- Ponto - What’s that?
- Ponto at work - ”Hello World”
- Standardization debate

Further details and didactic decisions
- The CURSOR class - a (not too) deliberate decision
- Choice of programming language & word processor

Outlook
- Potential future developments
- More information on Ponto / Conclusion
Introduction and Basics
- Motivation
- Basic informatics instruction in Bavaria
- Ponto - What’s that?
- Ponto at work - ”Hello World”
- Standardization debate

Further details and didactic decisions
- The CURSOR class - a (not too) deliberate decision
- Choice of programming language & word processor

Outlook
- Potential future developments
- More information on Ponto / Conclusion
Motivation

- Schools are supposed to fulfill their "Mission of Education" by taking up informatics into the range of compulsory subjects.
- Informatics instruction must not consist in the mere teaching of how to use standard software.
- On the other hand, a sole focus on theory seems equally futile.
- What might a compromise look like?
Motivation

- Schools are supposed to fulfill their "Mission of Education" by taking up informatics into the range of compulsory subjects.
- Informatics instruction must not consist in the mere teaching of how to use standard software.
- On the other hand, a sole focus on theory seems equally futile.
- What might a compromise look like?
Motivation

- Schools are supposed to fulfill their "Mission of Education" by taking up informatics into the range of compulsory subjects.
- Informatics instruction must not consist in the mere teaching of how to use standard software.
- On the other hand, a sole focus on theory seems equally futile.
- What might a compromise look like?
Motivation

- Schools are supposed to fulfill their "Mission of Education" by taking up informatics into the range of compulsory subjects.
- Informatics instruction must not consist in the mere teaching of how to use standard software.
- On the other hand, a sole focus on theory seems equally futile.
- What might a compromise look like?
Recently, Bavaria has introduced informatics as a mandatory subject into its secondary schools. All students (usually in the 6th grade) receive a minimum of one year of informatics instruction. During this time they learn the so-called *Fundamentum*.

The curriculum is structured around a central topic: Various kinds of documents. (e.g. text, graphic and hypertext documents)

Students approach & treat these documents as objects with attributes and methods - which is plainly the object-oriented way of thinking.
Recently, Bavaria has introduced informatics as a mandatory subject into its secondary schools. All students (usually in the 6th grade) receive a minimum of one year of informatics instruction. During this time they learn the so-called *Fundamentum*.

The curriculum is structured around a central topic: Various kinds of **documents**. (e.g. text, graphic and hypertext documents)

Students approach & treat these documents as **objects** with **attributes** and **methods** - which is plainly the **object-oriented way of thinking**.
Recently, Bavaria has introduced informatics as a mandatory subject into its secondary schools. All students (usually in the 6th grade) receive a minimum of one year of informatics instruction. During this time they learn the so-called Fundamentum.

The curriculum is structured around a central topic: Various kinds of documents. (e.g. text, graphic and hypertext documents)

Students approach & treat these documents as objects with attributes and methods - which is plainly the object-oriented way of thinking.
The idea of approaching & describing word processing (and esp. its output: text documents) in an object-oriented way is original to P. Hubwieser, professor at the TU Muenchen.
Ponto - What’s that?

- *Ponto* is a software -more exactly a *Python* module- which makes it possible to instantiate **objects** (i.e. paragraphs, characters, etc.) in *OpenOffice Writer* - the well-known open-source word processor.

- Furthermore, these **objects** are equipped with **values**... and **methods** to alter the latter (e.g. CHARACTER:FontSize)

- Thus, the **objects**, their **attributes**, and **methods**, which are provided via *Ponto’s* classes, constitute a simple yet strictly object-oriented terminology...

- ...which, to a large extent, complies with **Hubwieser’s** object-oriented text document model & terminology!
Ponto - What’s that?

- *Ponto* is a software -more exactly a *Python* module- which makes it possible to instantiate **objects** (i.e. paragraphs, characters, etc.) in *OpenOffice Writer* - the well-known open-source word processor.

- Furthermore, these **objects** are equipped with **values**... and **methods** to alter the latter (e.g. CHARACTER:FontSize)

- Thus, the **objects**, their **attributes**, and **methods**, which are provided via *Ponto’s* classes, constitute a simple yet strictly object-oriented terminology...

- ...which, to a large extent, complies with Hubwieser’s object-oriented text document model & terminology!
Ponto - What’s that?

- *Ponto* is a software -more exactly a *Python* module- which makes it possible to instantiate **objects** (i.e. paragraphs, characters, etc.) in *OpenOffice Writer* - the well-known open-source word processor.

- Furthermore, these **objects** are equipped with **values**... and **methods** to alter the latter (e.g. CHARACTER:FontSize)

- Thus, the **objects**, their **attributes**, and **methods**, which are provided via *Ponto’s* classes, constitute a simple yet strictly object-oriented terminology...

- ...which, to a large extent, complies with *Hubwieser’s* object-oriented text document model & terminology!
**Ponto - What’s that?**

- *Ponto* is a software -more exactly a *Python* module- which makes it possible to instantiate **objects** (i.e. paragraphs, characters, etc.) in *OpenOffice Writer* - the well-known open-source word processor.

- Furthermore, these **objects** are equipped with **values**... and **methods** to alter the latter (e.g. CHARACTER:FontSize)

- Thus, the **objects**, their **attributes**, and **methods**, which are provided via *Ponto’s* classes, constitute a simple yet strictly object-oriented terminology...

- ...which, to a large extent, complies with *Hubwieser’s* object-oriented text document model & terminology!
The following code snippet opens and instantiates a new OpenOffice Writer document:

```python
from ponto import DOCUMENT

document1 = DOCUMENT()
```
Ponto at work - "Hello World"

Then let's add a paragraph...

paragraph1 = document1.createParagraph('Hello world, I am a happy little paragraph!')
...set its left-sided indentation to 2 cm...

paragraph1.setIndentationLeft(2000)
...and change the first character’s size to 120pt.

```java
character1 = paragraph1.getCharacter(0)
character1.setFontSize(120)
```
Standardization debate

Framework for informatics instruction (results from research group *Educational Standards for Informatics*, GI at Dagstuhl 2004):

**Content Lines**
- information and data
- algorithms
- informatics systems
- technology
- theory
- society

**Perspectives**
- problem solving and modeling
- interpreting and reasoning
- communication
- connections
- representation
Outline

1 Introduction and Basics
   • Motivation
   • Basic informatics instruction in Bavaria
   • Ponto - What’s that?
   • Ponto at work - ”Hello World”
   • Standardization debate

2 Further details and didactic decisions
   • The CURSOR class - a (not too) deliberate decision
   • Choice of programming language & word processor

3 Outlook
   • Potential future developments
   • More information on Ponto / Conclusion
Original assumption:
A simple implementation of the "Bavarian" class diagrams would be most rewarding. This way, *Ponto* would automatically be in line with the teaching materials already available.

Second glance:
"Bavarian" class diagrams mainly describe static structures (ready-made text documents).

*Ponto*, however, would have to allow step-by-step document creation and thus needs to handle documents dynamically.

Conclusion: The "Bavarian" class diagrams would have to be slightly adapted.
The CURSOR class - a (not too) deliberate decision

- **Original assumption:**
  A simple implementation of the "Bavarian" class diagrams would be most rewarding. This way, *Ponto* would automatically be in line with the teaching materials already available.

- **Second glance:**
  "Bavarian" class diagramms mainly describe **static** structures (ready-made text documents).

- *Ponto*, however, would have to allow step-by-step document creation and thus needs to handle documents dynamically.

- **Conclusion:** The "Bavarian" class diagrams would have to be slightly adapted.
The CURSOR class - a (not too) deliberate decision I

- **Original assumption:**
  A simple implementation of the "Bavarian" class diagrams would be most rewarding. This way, *Ponto* would automatically be in line with the teaching materials already available.

- **Second glance:**
  "Bavarian" class diagrams mainly describe **static** structures (ready-made text documents).

- *Ponto*, however, would have to allow step-by-step document creation and thus needs to handle documents dynamically.

- **Conclusion:** The "Bavarian" class diagrams would have to be slightly adapted.
The CURSOR class - a (not too) deliberate decision

- **Original assumption:**
  A simple implementation of the "Bavarian" class diagrams would be most rewarding. This way, *Ponto* would automatically be in line with the teaching materials already available.

- **Second glance:**
  "Bavarian" class diagrams mainly describe static structures (ready-made text documents).

- **Ponto**, however, would have to allow step-by-step document creation and thus needs to handle documents dynamically.

- **Conclusion:** The "Bavarian" class diagrams would have to be slightly adapted.
The development of *Ponto* started out from UML diagrams provided by S. Voss (TU Muenchen), which make up a detailed object-oriented model of document handling in *OpenOffice Writer*.

Basically, Voss's *OpenOffice Writer* class diagram contains these classes (translated into English):

- **DOCUMENT**
- **SECTION**
- **PARAGRAPH**
- **CHARACTER**
- **PAGESTYLE**
- **PAGE**
The CURSOR class - a (not too) deliberate decision II

- The development of *Ponto* started out from UML diagrams provided by S. Voss (TU Muenchen), which make up a detailed object-oriented model of document handling in *OpenOffice Writer*.

- Basically, Voss's *OpenOffice Writer* class diagram contains these classes (translated into English):

  DOCUMENT   SECTION   PARAGRAPH
  CHARACTER   PAGESTYLE   PAGE
The development of Ponto started out from UML diagrams provided by S. Voss (TU Muenchen), which make up a detailed object-oriented model of document handling in OpenOffice Writer.

Basically, Voss's OpenOffice Writer class diagram contains these classes (translated into English):

- DOCUMENT
- SECTION
- PARAGRAPH
- CHARACTER
- PAGESTYLE
- PAGE
Besides making slight simplifications to the class structure as suggested by Voss, we have also added a CURSOR class - for several reasons:

1. *Ponto* handles documents dynamically, i.e. they can be created step-by-step.

2. *Ponto* was meant to support a maximum number of standards for informatics instruction. The CURSOR class makes the execution of small algorithms on text documents possible:
   - CURSOR provides methods for navigating through the text: `forward()`, `back()`
   - CURSOR.getCharacter() can be used to retrieve and manipulate the character at the current cursor position.
Besides making slight simplifications to the class structure as suggested by Voss, we have also added a CURSOR class - for several reasons:

1. *Ponto* handles documents dynamically, i.e. they can be created step-by-step.

2. *Ponto* was meant to support a maximum number of standards for informatics instruction. The CURSOR class makes the execution of small algorithms on text documents possible:
   - CURSOR provides methods for navigating through the text: `forward()`, `back()`
   - `CURSOR.getCharacter()` can be used to retrieve and manipulate the character at the current cursor position.
Besides making slight simplifications to the class structure as suggested by Voss, we have also added a CURSOR class - for several reasons:

1. *Ponto* handles documents dynamically, i.e. they can be created step-by-step.

2. *Ponto* was meant to support a maximum number of standards for informatics instruction. The CURSOR class makes the execution of small algorithms on text documents possible:
   - CURSOR provides methods for navigating through the text: forward(), back()
   - CURSOR.getCharacter() can be used to retrieve and manipulate the character at the current cursor position
Besides making slight simplifications to the class structure as suggested by Voss, we have also added a CURSOR class - for several reasons:

1. *Ponto* handles documents dynamically, i.e. they can be created step-by-step.

2. *Ponto* was meant to support a maximum number of standards for informatics instruction. The CURSOR class makes the execution of small algorithms on text documents possible:
   - CURSOR provides methods for navigating through the text: `forward()`, `back()`
   - CURSOR.getCharacter() can be used to retrieve and manipulate the character at the current cursor position
Besides making slight simplifications to the class structure as suggested by Voss, we have also added a CURSOR class - for several reasons:

1. Ponto handles documents dynamically, i.e. they can be created step-by-step.
2. Ponto was meant to support a maximum number of standards for informatics instruction. The CURSOR class makes the execution of small algorithms on text documents possible:
   - CURSOR provides methods for navigating through the text: forward(), back()
   - CURSOR.getCharacter() can be used to retrieve and manipulate the character at the current cursor position
Why *Python*? Because...

- *Python* supports object orientation.
- *Python* syntax is simple, yet powerful.
- Programs/algorithms can be entered interactively through the *Python* shell.
- *OpenOffice.org* by default already comes with a low-level Python interface (pyUNO - the Python-UnO bridge).
- As opposed to *OpenOffice.org’s* built-in *StarBasic*, *Python* is a standard programming language with a wide range of applications - *StarBasic* is merely another BASIC variant limited to a single domain.
- Hence, *Python* is also ideally suited as the standard programming language in informatics classes.
Why *Python*? Because...

- *Python* supports object orientation.
- *Python* syntax is simple, yet powerful.
- Programs/algorithms can be entered interactively through the *Python* shell.
- *OpenOffice.org* by default already comes with a low-level Python interface (pyUNO - the Python-UnO bridge).
- As opposed to *OpenOffice.org*’s built-in *StarBasic*, *Python* is a standard programming language with a wide range of applications - *StarBasic* is merely another BASIC variant limited to a single domain.
- Hence, *Python* is also ideally suited as the standard programming language in informatics classes.
Why *Python*? Because...

- *Python* supports object orientation.
- *Python* syntax is simple, yet powerful.
- Programs/algorithms can be entered interactively through the *Python* shell.
- *OpenOffice.org* by default already comes with a low-level *Python* interface (pyUNO - the Python-Uno bridge).
- As opposed to *OpenOffice.org*’s built-in *StarBasic*, *Python* is a standard programming language with a wide range of applications - *StarBasic* is merely another BASIC variant limited to a single domain.
- Hence, *Python* is also ideally suited as the standard programming language in informatics classes.
Why *Python*? Because...

- *Python* supports object orientation.
- *Python* syntax is simple, yet powerful.
- Programs/algorithms can be entered interactively through the *Python* shell.
- *OpenOffice.org* by default already comes with a low-level Python interface (pyUNO - the Python-Uno bridge).
- As opposed to *OpenOffice.org’s* built-in *StarBasic*, *Python* is a standard programming language with a wide range of applications - *StarBasic* is merely another BASIC variant limited to a single domain.
- Hence, *Python* is also ideally suited as the standard programming language in informatics classes.
Why *Python*? Because...

- *Python* supports object orientation.
- *Python* syntax is simple, yet powerful.
- Programs/algorithms can be entered interactively through the *Python* shell.
- *OpenOffice.org* by default already comes with a low-level Python interface (pyUNO - the Python-Uno bridge).
- As opposed to *OpenOffice.org*'s built-in *StarBasic*, *Python* is a standard programming language with a wide range of applications - *StarBasic* is merely another BASIC variant limited to a single domain.
- Hence, *Python* is also ideally suited as the standard programming language in informatics classes.
**Why Python? Because...**

- *Python* supports object orientation.
- *Python* syntax is simple, yet powerful.
- Programs/algorithms can be entered interactively through the *Python* shell.
- *OpenOffice.org* by default already comes with a low-level Python interface (pyUNO - the Python-Uno bridge).
- As opposed to *OpenOffice.org*’s built-in *StarBasic*, *Python* is a standard programming language with a wide range of applications - *StarBasic* is merely another BASIC variant limited to a single domain.
- Hence, *Python* is also ideally suited as the standard programming language in informatics classes.
Why *OpenOffice.org*? Because...

- ...it is platform independent.
- ...it comes with the pyUNO bridge software.
- ...it is a well-established, state-of-the-art office package.
- ...it is free of charge (open source, *GPL*).
Why *OpenOffice.org*? Because...

- ...it is platform independent.
- ...it comes with the pyUNO bridge software.
- ...it is a well-established, state-of-the-art office package.
- ...it is free of charge (open source, *GPL*).
Why *OpenOffice.org*? Because...

- ...it is platform independent.
- ...it comes with the pyUNO bridge software.
- ...it is a well-established, state-of-the-art office package.
- ...it is free of charge (open source, *GPL*).
Why *OpenOffice.org*? Because...

- ...it is platform independent.
- ...it comes with the pyUNO bridge software.
- ...it is a well-established, state-of-the-art office package.
- ...it is free of charge (open source, *GPL*).
Outline

1. Introduction and Basics
   - Motivation
   - Basic informatics instruction in Bavaria
   - Ponto - What’s that?
   - Ponto at work - ”Hello World”
   - Standardization debate

2. Further details and didactic decisions
   - The CURSOR class - a (not too) deliberate decision
   - Choice of programming language & word processor

3. Outlook
   - Potential future developments
   - More information on Ponto / Conclusion
Further inferaces for school use in Python? [Hum04]

- Gimp – image editing
  - http://www.gimp.org/
- Dia – vector drawings
  - http://www.gnome.org/projects/dia/
- Skencil [formerly "Sketch"] – vector drawings
  - http://www.nongnu.org/skencil/
- Scribus – desktop publishing
  - http://www.scribus.org.uk/
Further infereraces for school use in Python? [Hum04]

- **Gimp** – image editing
- **Dia** – vector drawings
- **Skencil** [formerly "Sketch"] – vector drawings
  - [http://www.nongnu.org/skencil/](http://www.nongnu.org/skencil/)
- **Scribus** – desktop publishing
  - [http://www.scribus.org.uk/](http://www.scribus.org.uk/)
Further inferaces for school use in Python?

[Hum04]
Further inferaces for school use in Python? [Hum04]

- Gimp – image editing
- Dia – vector drawings
- Skencil [formerly ”Sketch”] – vector drawings
  - [http://www.nongnu.org/skencil/](http://www.nongnu.org/skencil/)
- Scribus – desktop publishing
  - [http://www.scribus.org.uk/](http://www.scribus.org.uk/)
Further inferaces for school use in Python?

Design of an Informatics System to Bridge the Gap between Using and Understanding in Informatics (Ponto)
Further interfaces for school use in Python? [Hum04]

- Gimp – image editing
  - http://www.gimp.org/

- Dia – vector drawings
  - http://www.gnome.org/projects/dia/

- Skencil [formerly ”Sketch”] – vector drawings
  - http://www.nongnu.org/skencil/

- Scribus – desktop publishing
  - http://www.scribus.org.uk/
Further interfaces for school use in Python? [Hum04]

- Gimp – image editing
  - http://www.gimp.org/
- Dia – vector drawings
  - http://www.gnome.org/projects/dia/
- Skencil [formerly "Sketch"] – vector drawings
  - http://www.nongnu.org/skencil/
- Scribus – desktop publishing
  - http://www.scribus.org.uk/
Further inferaces for school use in Python?

Christiane Borchel, Ludger Humbert, Martin Reinertz

Design of an Informatics System to Bridge the Gap between Using and Understanding in Informatics (Ponto)
Further interfaces for school use in Python? [Hum04]

- Gimp – image editing
  - http://www.gimp.org/
- Dia – vector drawings
  - http://www.gnome.org/projects/dia/
- Skencil [formerly ”Sketch”] – vector drawings
  - http://www.nongnu.org/skencil/
- Scribus – desktop publishing
  - http://www.scribus.org.uk/
Conclusion

- In the meantime, Ponto was tested with 6th and 7th graders at the Willy-Brandt-Gesamtschule, Bergkamen, Germany.
- In the course of this experimental teaching session, students made frequent use of dot-notation and also exhibited object-oriented ways of thinking in other respects. [Hum04]
- These first results MAY indicate that Ponto indeed helps ”bridge the gap” between theory and practice.
In the meantime, Ponto was tested with 6th and 7th graders at the Willy-Brandt-Gesamtschule, Bergkamen, Germany. In the course of this experimental teaching session, students made frequent use of dot-notation and also exhibited object-oriented ways of thinking in other respects. [Hum04]

These first results MAY indicate that Ponto indeed helps "bridge the gap" between theory and practice.
In the meantime, Ponto was tested with 6th and 7th graders at the Willy-Brandt-Gesamtschule, Bergkamen, Germany.

In the course of this experimental teaching session, students made frequent use of dot-notation and also exhibited object-oriented ways of thinking in other respects. [Hum04]

These first results MAY indicate that Ponto indeed helps "bridge the gap" between theory and practice.
Introduction and Basics
Further details and didactic decisions
Outlook

More information on Ponto

http://ddi.cs.uni-dortmund.de/projekte/Ponto
Jürgen Baumert, Cordula Artelt, Eckhard Klieme, Michael Neubrand, Manfred Prenzel, Ulrich Schiefele, Wolfgang Schneider, Klaus-Jürgen Tillmann, and Manfred Weiß, editors.


Torsten Brinda, Volker Claus, Ira Diethelm, Ludger Humbert, Dietmar Johlen, Johannes Magenheim, Peter Micheuz, Eckart Modrow, Hermann Puhlmann, Olaf Scheel, Markus Schneider, Sigrid Schubert, Carsten Schulte, Andreas Schwill, and Albert Zündorf.


Peter Brichzin, Ulrich Freiberger, Klaus Reinold, and Albert Wiedemann.


Elke Frey, Peter Hubwieser, and Ferdinand Winhard.

*Informatik 1. Objekte, Strukturen, Algorithmen (Schülerbuch).*


Christian F. Görlich and Ludger Humbert.
In Lutterbeck and Gehring [LG05].


Ludger Humbert and Hermann Puhlmann. 
Essential Ingredients of Literacy in Informatics. 
In Magenheim and Schubert [MS04], pages 65–76.


Peter Hubwieser. 
Informatik als Pflichtfach an bayerischen Gymnasien. 
In Schwill [Sch99], pages 165–174.


Peter Hubwieser, editor. 


Ludger Humbert. 
Which programming language supports my concepts for education in informatics at school?, July 2002.
Abstract:
presentation, manuscript:
http://www.ham.nw.schule.de/pub/bscw.cgi/0/20866.

Ludger Humbert.
Let’s teach informatics – empowering pupils, students and teachers.
In van Weert and Munro [vWM03], pages 141–147.
– July 22-26, 2002, University of Dortmund, Germany.

Ludger Humbert.
Zur wissenschaftlichen Fundierung der Schulinformatik.
pad-Verlag, Witten, März 2003.
zugl. Dissertation an der Universität Siegen
http://www.ham.nw.schule.de/pub/bscw.cgi/d38820/ – last visited:

Ludger Humbert.
Presentation, manuscript:
http://www.ham.nw.schule.de/pub/bscw.cgi/0/89349.

Bernd Lutterbeck and Robert A. Gehring, editors.

Johannes Magenheim and Sigrid Schubert, editors.

Hermann Puhlmann.
Informatische Literalität nach dem PISA-Muster.
In Hubwieser [Hub03], pages 145–154.

Andreas Schwill, editor.
Informatik und Schule – Fachspezifische und fachübergreifende didaktische Konzepte.

Siglinde Voss.
Objektorientierte Modellierung von Software zur Textgestaltung.
In Hubwieser [Hub03], pages 211–223.

Christiane Borchel, Ludger Humbert, Martin Reinertz
Design of an Informatics System to Bridge the Gap between Understanding and Using in Informatics (Ponto)
Tom J. van Weert and Robert K. Munro, editors.
– July 22-26, 2002, University of Dortmund, Germany.

Joachim Wirth and Eckard Klieme.
*Computernutzung*.
In Baumert et al. [BAK+03], pages 195–211.