

Ponto Target Specification

Version 1.0

June 16th, 2004

created by Christiane Borchel und Martin Reinertz
for a research project part of "Didaktik der Informatik"
in the summer semester of 2004

University of Dortmund

Sub-Department 12 - Computer Science, Curriculum and Instruction ("Didaktik der Informatik")

Contents

1	Modifications	3
2	Goals	3
2.1	Mandatory features	3
2.2	Desirable features	3
2.3	Limits (What <i>Ponto</i> is not... / does not)	4
3	Product use	4
3.1	Fields of use:	4
3.2	Operating conditions	4
4	Product environment	4
4.1	Software	4
4.2	Hardware	5
4.3	Orgware	5
4.4	Interfaces to other products	5
5	Product functionality	5
6	Product data	6
6.1	<i>Ponto</i> software data	6
6.2	Product benefits	6
7	Graphical user interface (GUI)	6
8	Quality objectives	6
9	Global testing	6
10	Development environment	6
10.1	Software	6
10.2	Hardware	7
10.3	Orgware	7
11	Additions / Other aspects	7

1 Modifications

In the process of developing *Ponto* certain modifications to this document may be necessary. Any such changes will be summarized in this section.

Version 0.1 - Initial version

Version 0.2 - Minor corrections

Version 0.3 - Project name change from *Bridgeware* to *Ponto*

Version 1.0 - Definite specification (incl. exclusion of optional components)

2 Goals

Ponto is a small program that makes it possible to conveniently control the OpenOffice Writer software (word processor) - which is part of the OpenOffice project from the Python programming language shell. In more detail, *Ponto* is an extensively simplified (and thus not complete!) interface / API between Python and the Python UNO bridge software. Since *Ponto* is mainly conceived as a tool for use in the secondary education computer science classroom (*Sekundarstufe I* in Germany), all of *Ponto's* qualities and features will be chosen and implemented according to methodological principles and classroom needs (these criteria will be elaborated on at a later point in the project).

2.1 Mandatory features

- *Ponto* must provide classes for Character, ..., Paragraph, Template and Document
- For each of these classes, *Ponto* must also provide set- & get-methods to provide access to relevant attributes (such as font size, font and text color for Character objects)
- Besides simple text, *Ponto* should also provide methods to insert (and subsequently modify) tables and images into documents
- *Ponto* modules can be imported into python programs

2.2 Desirable features

- Defineability of templates through inheritance (e.g. for paragraphs and headings)
- Automated generation of TOCs from headings in a document
- Extensive information material such as a tutorial and exercises
- Control over vector images in OpenOffice Draw via *Ponto*

2.3 Limits (What *Ponto* is not... / does not)

- a full re-implementation of the Python UNO bridge software
- provide access to other parts of the OpenOffice software package (e.g. OpenOffice Calc, Draw and Impress)

3 Product use

Using *Ponto* (in the classroom), students / people can gain a deep, object-oriented understanding of the kinds of structures implemented in word processing software.

3.1 Fields of use:

- Computer science classroom topic: word processors
- Computer science classroom topic: introduction to object-oriented thinking / modeling
- Self-study
- Users aimed at
- Students and teachers
- Autodidacts and other learner groups

3.2 Operating conditions

- Computer science classrooms in schools
- School computer networks
- Single Desktop PCs, e.g. at school / home

4 Product environment

Ponto has been written in the Python programming language and can therefore be used under various operating systems, i.e. it is platform independent.

4.1 Software

- Operating systems: Linux, Windows 9x/NT/2000/XP, etc.
- OpenOffice (from version 1.1)
- Python
- PyUno (since OpenOffice Version 1.1RC4 Python and PyUno are part of the standard installation)

4.2 Hardware

- PC
- Screen
- Keyboard
- Mouse/Trackball/Trackpad

4.3 Orgware

none

4.4 Interfaces to other products

- Python
- PyUno

5 Product functionality

Aim: Python programs process /manipulate (i.e. create, change, etc.) text documents within OpenOffice Writer.

Precondition: A Python program has been created (with a text editor of the users choice), OpenOffice is running. Optionally, a text file has been created that contains the text to be processed.

Application flow:

1. User starts Python program (here he may also pass the previously created text file as a parameter).
2. Based on the instructions within the Python program the text that is: a) passed as a parameter, b) contained in the Python program or c) already open inside OpenOffice Writer, is being processed and changed accordingly.

Alternatives:

- 1a.** The text to be processed is part of the Python program.
- 1b.** The text is being imported from a text file that is passed on as a parameter.
- 1c.** The text already exists in the form of an OpenOffice Writer document and is being passed on as a parameter.
- 1d.** The text exists in the form of an OpenOffice Writer document that has already been opened.

6 Product data

6.1 *Ponto* software data

Python programs are saved as text files with the ending `.py` . This also applies to Python programs using *Ponto*.

6.2 Product benefits

Using *Ponto*, a user can control the OpenOffice Writer software.

7 Graphical user interface (GUI)

Ponto does not provide a user interface of its own. Most (if not all) programming tools for the Python programming language (e.g. specifically designed editors, but also simple ones) can be used along with *Ponto* it has been designed for use from the Python shell.

8 Quality objectives

Product quality	very good	good	normal	not relevant
Functionality	x			
Reliability	x			
Useability		x		
Efficiency			x	
Transferability		x		

9 Global testing

- Method tests, as part of Software Development Cycle
- Optional class tests
- Visual evaluation of program runs on self-created documents / texts
- Beta users (feedback)

10 Development environment

10.1 Software

- same as product environment +
- Lyx
- Emacs

- other editors and tools, if necessary

10.2 Hardware

same as product environment

10.3 Orgware

- Internet connection
- BSCW system
- Version management system CVS

11 Additions / Other aspects

none