Literate programming IDE for LARCC

November 30, 2013

1. Introduction
    - Literate Programming
    - Compute-with-Cochains over LAR
2. Software tools
    - LaTeX
    - Leo
    - Nuweb
    - Makefile
    - GitHub
3. Up and Running
4. Structure of LARCC

Introduction

Why this document was written

Some ideas, links, and docs on how to write technical papers, including source programming codes, that anyone can understand and execute

This document was written to start documenting the implementation of the new software system entitled either 3C/LAR (to read as 'Compute with CoChains over LAR') or LAR4CCC (LAR for Compute with CoChains)

BUT

it may work well for any kind of scientific programming including the computer code and both the how and the why it works ...
In any language or combination of languages

Literate Programming

Donald Knuth. "Literate Programming (1984)" in Literate Programming. CSLI, 1992, pg. 99.

I believe that the time is ripe for significantly better documentation of programs, and that we can best achieve this by considering programs to be works of literature. Hence, my title: "Literate Programming."

Let us change our traditional attitude to the construction of programs: Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on explaining to human beings what we want a computer to do.

Literate Programming

Donald Knuth. "Literate Programming (1984)" in Literate Programming. CSLI, 1992, pg. 99.

I believe that the time is ripe for significantly better documentation of programs, and that we can best achieve this by considering programs to be works of literature. Hence, my title: "Literate Programming."

Let us change our traditional attitude to the construction of programs: Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on explaining to human beings what we want a computer to do.

The practitioner of literate programming can be regarded as an essayist, whose main concern is with exposition and excellence of style. Such an author, with thesaurus in hand, chooses the names of variables carefully and explains what each variable means. He or she strives for a program that is comprehensible because its concepts have been introduced in an order that is best for human understanding, using a mixture of formal and informal methods that reinforce each other.

Donald Knuth

The CWEB System of Structure Documentation. Addison-Wesley. 1994. pg. 1.

An experienced programmer, to provide the best possible documentation of software products, needs two things simultaneously: a language like TeX for formatting, and a language like C for programming.

Neither type of language can provide the best documentation by itself; but when both are appropriately combined, we obtain a system that is much more useful than either language separately.

Donald Knuth

The CWEB System of Structure Documentation. Addison-Wesley. 1994. pg. 1.

An experienced programmer, to provide the best possible documentation of software products, needs two things simultaneously: a language like TeX for formatting, and a language like C for programming.

Neither type of language can provide the best documentation by itself; but when both are appropriately combined, we obtain a system that is much more useful than either language separately.

- The Art of Computer Programming

  At the end of 1999, these books were named among the best twelve physical-science monographs of the century by *American Scientist*, along with: Dirac on *quantum mechanics*, Einstein on *relativity*, Mandelbrot on *fractals*, Pauling on the *chemical bond*, Russell and Whitehead on *foundations of mathematics*, von Neumann and Morgenstern on *game theory*, Wiener on *cybernetics*, Woodward and Hoffmann on *orbital symmetry*, Feynman on *quantum electrodynamics*, Smith on the *search for structure*, and Einstein's *collected papers*

- Computers & Typesetting

  complete documentation of the TeX and METAFONT systems for digital typography. "Never before has a computer program of this size been spelled out so clearly and completely."

Literate Programming resources

http://www.literateprogramming.com/

Quotes

- Literate Programming
- Software Documentation
- Design Documentation
- Agile Documentation
- Source Code Comments
- Software Aging

Literate Programming resources

http://www.literateprogramming.com/

Quotes

- Literate Programming
- Software Documentation
- Design Documentation
- Agile Documentation
- Source Code Comments
- Software Aging

Category

- CWEB
- Articles
- Tools
- Links

Community

- Feedback

Starting today !!

Compute-with-CoChains over LAR software in development

1. CAD-PLM Lab, Dip. Matematica e Fisica, Univ Roma Tre
2. Spatial Automation Lab, Univ of Wisconsin at Madison
3. CEDMAV, SCI Institute, Univ of Utah

Starting today !!

Compute-with-CoChains over LAR software in development

1. CAD-PLM Lab, Dip. Matematica e Fisica, Univ Roma Tre
2. Spatial Automation Lab, Univ of Wisconsin at Madison
3. CEDMAV, SCI Institute, Univ of Utah

Integrated tools being used (for now)

- GitHub (social development)
- Literate programming (Latex + Leo + Nuweb)
- Haskell (as specification language)
- Python & PyOpenCL (for rapid Prototyping)
- Javascript & WebCL (for client-based web applications)
- C++ & OpenCL (for optimized deployement)

LaTeX — A document preparation system

Obtaining LaTeX

- LaTeX is the de facto standard for the communication and publication of scientific documents
- It is a specialisation of TeX, the highest-quality typesetting system by D. Knuth, and includes features designed for fast production of technical and scientific documentation
- LaTeX is available as free software

(from www.latex-project.org)

LaTeX — A document preparation system

Obtaining LaTeX

- LATEX is the de facto standard for the communication and publication of scientific documents
- It is a specialisation of TEX, the highest-quality typesetting system by D. Knuth, and includes features designed for fast production of technical and scientific documentation
- LATEX is available as free software

(from www.latex-project.org)

The larcc IDE requires the users to embed the compute code within LATEX files written for documenting their work. Therefore the first requirement is a working LATEX environment

Remark (For Windows users)

"As TEX Live is the basis of MacTEX, and is the TEX system for Unix, if you work cross-platform and want an identical system on all of your machines, then TEX Live is the way to go".

Leo editor

After the first experiments with literate programming I realised that any standard (linear) editor is not the best one for it. **Then I found a wonderful non-linear editor**

Remark (What it is)

Leo is a PIM, IDE and outliner that accelerates the work-flow of programmers, authors and web designers

"Leo is the best IDE that I have had the pleasure to use. It has totally changed not only the way that I program, but also the way that I store and organize all of the information that I need for the job that I do." Ian Mulvany

Leo editor

After the first experiments with literate programming I realised that any standard (linear) editor is not the best one for it. **Then I found a wonderful non-linear editor**

Remark (What it is)

Leo is a PIM, IDE and outliner that accelerates the work-flow of programmers, authors and web designers

Remark (Leo features)

- *Leo outlines are views on an underlying graph (DAG)*
- *Outline nodes can reside in many places within a single outline.*
- *Outline-oriented markup generates external files from outlines.*

"Leo is the best IDE that I have had the pleasure to use. It has totally changed not only the way that I program, but also the way that I store and organize all of the information that I need for the job that I do." Ian Mulvany

Learn about Leo in two hours

download: http://leoeditor.com/

Multi-language literate programming

The simplest incarnation of the Knut's original work

Nuweb works with any programming language and LaTeX, and is probably the simplest incarnation of the Knut's original work.
The web site of the tool is
sourceforge.net/projects/nuweb/
A revised version of source files:
code.google.com/p/nuweb

Multi-language literate programming

The simplest incarnation of the Knut's original work

Nuweb works with any programming language and LaTeX, and is probably the simplest incarnation of the Knut's original work.
The web site of the tool is sourceforge.net/projects/nuweb/
A revised version of source files: code.google.com/p/nuweb

This package can build using the standard tools:

```
$ cd <path-to>/nuweb/
$ ./configure
$ make
$ sudo make install
```

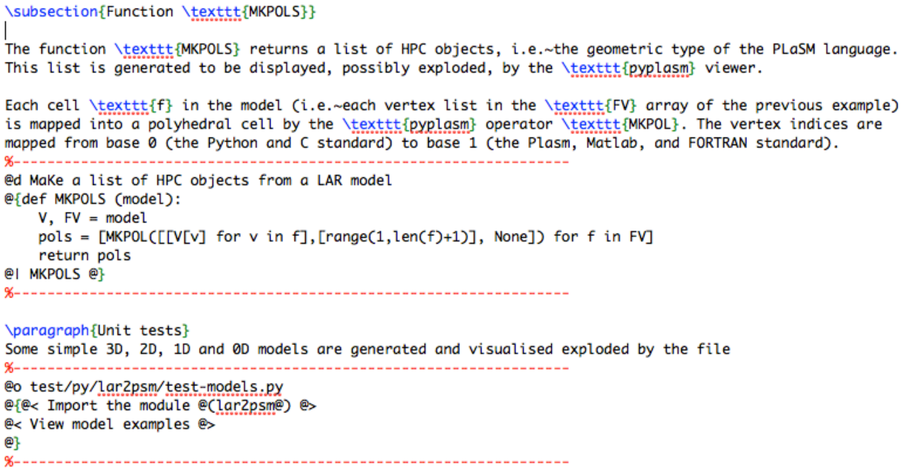
For some documentation read the wiki page. Test your installation by just compiling to pdf the nuweb.w document itself, whose chapter one contains the user documentation:

```
$ nuweb nuweb.w
```

User manual (first chapter of compiled `nuweb.w`): nuwebdoc.pdf

Nuweb example 1/2

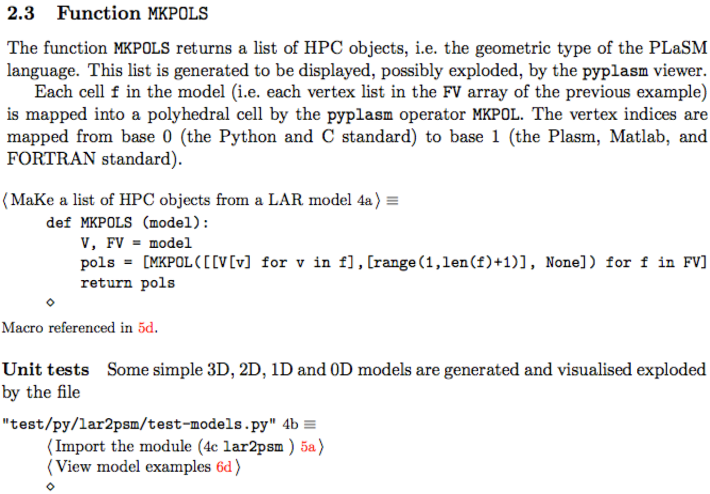
Sorgente LaTeX con marcatura Nuweb



```
\subsection{Function \texttt{MKPOLS}}

The function \texttt{MKPOLS} returns a list of HPC objects, i.e.~the geometric type of the PLaSM language.
This list is generated to be displayed, possibly exploded, by the \texttt{pyplasm} viewer.

Each cell \texttt{f} in the model (i.e.~each vertex list in the \texttt{FV} array of the previous example)
is mapped into a polyhedral cell by the \texttt{pyplasm} operator \texttt{MKPOL}. The vertex indices are
mapped from base 0 (the Python and C standard) to base 1 (the Plasm, Matlab, and FORTRAN standard).
%-------------------------------------------------------------------------------
@d MaKe a list of HPC objects from a LAR model
@{def MKPOLS (model):
    V, FV = model
    pols = [MKPOL([[V[v] for v in f],[range(1,len(f)+1)], None]) for f in FV]
    return pols
@| MKPOLS @}
%-------------------------------------------------------------------------------

\paragraph{Unit tests}
Some simple 3D, 2D, 1D and 0D models are generated and visualised exploded by the file
%-------------------------------------------------------------------------------
@o test/py/lar2psm/test-models.py
@{@< Import the module @(lar2psm@) @>
@< View model examples @>
@}
%-------------------------------------------------------------------------------
```

Nuweb example 2/2

compilato pdf



2.3 Function MKPOLS

The function MKPOLS returns a list of HPC objects, i.e. the geometric type of the PLaSM language. This list is generated to be displayed, possibly exploded, by the pyplasm viewer.

Each cell f in the model (i.e. each vertex list in the FV array of the previous example) is mapped into a polyhedral cell by the pyplasm operator MKPOL. The vertex indices are mapped from base 0 (the Python and C standard) to base 1 (the Plasm, Matlab, and FORTRAN standard).

⟨MaKe a list of HPC objects from a LAR model 4a⟩ ≡

```
    def MKPOLS (model):
        V, FV = model
        pols = [MKPOL([[V[v] for v in f],[range(1,len(f)+1)], None]) for f in FV]
        return pols
    ◇
```

Macro referenced in 5d.

Unit tests Some simple 3D, 2D, 1D and 0D models are generated and visualised exploded by the file

"test/py/lar2psm/test-models.py" 4b ≡

```
    ⟨Import the module (4c lar2psm ) 5a⟩
    ⟨View model examples 6d⟩
    ◇
```

Make utility

Just few words ...

In software development, Make is a utility that automatically builds executable programs and libraries from source code by reading files called makefiles which specify how to derive the target program.

Make utility

Just few words ...

In software development, Make is a utility that automatically builds executable programs and libraries from source code by reading files called makefiles which specify how to derive the target program.

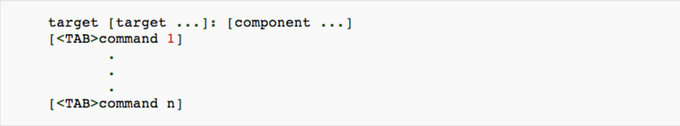
Make is invoked with a list of target file names to build as command-line arguments:

```
$ make [TARGET ...]
```

Without arguments, Make builds the first target that appears in its makefile, which is traditionally a target named all

Makefile short tutorial (wikipedia)

- A makefile consists of rules. Each rule begins with a textual dependency line which defines a target followed by a colon (:) and optionally an enumeration of components (files or other targets) on which the target depends
- It is common to refer to components as prerequisites of the target.
- Usually each rule has a single unique target, rather than multiple targets.



```
target [target ...]: [component ...]
[<TAB>command 1]
       .
       .
       .
[<TAB>command n]
```

- After each dependency line, a series of command lines may follow which define how to transform the components (usually source files) into the target (usually the "output"). If any of the components have been modified, the command lines are run.
- Each command line must begin with a tab character to be recognized as a command.

The (current) Makefile for larcc — I

```
#
# Makefile for cclar
#

NAME = lar2psm
LANGUAGE = py
BIBFILE = $(NAME).bib

IDIR = src/tex/
ODIR = lib/
DOCTEX = doc/tex/
DOCPDF = doc/pdf/

all:
    echo building $(NAME)
    make pdf
    make clear
    open $(DOCPDF)$(NAME).pdf

exec:
    cp $(IDIR)macros.tex macros.tex
    cp $(IDIR)bib.bib $(BIBFILE)
    cp -R $(IDIR)images .
    cp $(IDIR)$(NAME).tex $(NAME).w

    nuweb $(NAME).w
```

The (current) Makefile for larcc — II

```
pdf: $(IDIR)$(NAME).tex
    make exec

    pdflatex $(NAME).tex
    nuweb $(NAME)
    bibtex $(NAME)

    pdflatex $(NAME).tex
    pdflatex $(NAME).tex

html:
    make pdf

    rm -dfR $(NAME)/*
    rm -dfR $(NAME)
    mkdir $(NAME)
    cp src/html/css.cfg $(NAME).cfg
    makeindex $(NAME).tex
    htlatex $(NAME).tex "$(NAME).cfg,TocLink,html,index=2,3"
    mv -fv images $(NAME).html $(NAME).css $(NAME)
    rm -fv $(NAME).* macros.tex
    mv -fv $(NAME)*.* $(NAME)
    if [ -d doc/html/$(NAME) ] ; then rm -R doc/html/$(NAME) ; fi
    mv $(NAME) doc/html/
    open doc/html/$(NAME)/$(NAME).html

tests:
    echo ‘python test/py/test01.py‘
    echo ‘python test/py/test02.py‘
```

The (current) Makefile for larcc — III

```
    echo ‘python test/py/test06.py‘


clear:
    mv -fv $(NAME).tex $(NAME).bbl macros.tex $(DOCTEX)
    mv -fv $(NAME).pdf $(DOCPDF)
    mv -fv $(NAME).w $(ODIR)w
    if [ -d $(DOCTEX)images ] ; then rm -R $(DOCTEX)images ; fi
    mv -fv images $(DOCTEX)
    rm $(NAME).*
```

Git & GitHub

Git the tool, **GitHub** the service for projects that uses Git

Every Git working directory is a full-fledged repository with complete history and full version tracking capabilities, not dependent on network access or a central server

- Put your IDE under the protection of a version control system.
- `larcc` comes from Github equipped with an integrated Git

Git & GitHub

Git the tool, **GitHub** the service for projects that uses Git

Every Git working directory is a full-fledged repository with complete history and full version tracking capabilities, not dependent on network access or a central server

- Put your IDE under the protection of a version control system.
- `larcc` comes from Github equipped with an integrated Git

- Mac OS X: `Git` available by default.
- Else: download and install http://git-scm.com/downloads

Remark (To download `larcc`)

```
$ git clone https://github.com/cvdlab/larcc
```

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb

aaaaaaaaaa

bbbbbb