From Zero to SiD-Installation

How to Run SiD Sim and Reco on your desktop/laptop ?

First of all, you need a recent linux box with the following ingredients

- C/C++ Compiler gcc 4.4 or greater
- Java 7 (Java 6 may still work but)
- CVS
- SVN
- Latex2html

I've used

- OpenSuse 12.3 x86-64 with gcc 4.7.2 and Java OpenJDK 1.7.0_21
- SL 6.4 x86-64 with gcc 4.4.7 and Java OpenJDK 1.7.0_25
- SL 6.4 x86-64 with gcc 4.4.7 and Java OpenJDK 1.7.0_25
- Ubuntu 12.04 LTS with gcc 4.6.3 and Java OpenJDK 1.7.0_55

all modified scripts can be found in this tarball

Bugs reported by Alex Finch:

/sid_complete/v01-17-02-sid/KalDet/v01-12/kern/EXVKalDetector.h
virtual Double_t GetBfield (const TVector3 &xx = TVector3()) const

which won't compile with latest ROOT versions. The ROOT release note were updated at my request to indicate this.

Preparation

Prepare an area:

```
mkdir /scratch/sid_complete
cd /scratch/sid_complete
```

then get the most "recent" CERNLIB from here, you needs the libs and the includes, be careful to have the libs for your architecture

install the CERNLIB in therein the folder */scratch/sid_complete/cernlib2006* with two subfolders

- *lib*
- *include*

I have used symbolic links to set this up

```
ln -s 2006b/x86_64-slc5-gcc43-opt/lib lib
ln -s 2006b/include/ include
```

unpack the scripts into the myscripts area

```
tar -xzvf myscripts.tar.gz
```

ILCSOFT installation

Get ilcinstall from here: ilcinstall

I've used : v01-17-02

```
mkdir ilcinstall
cd ilcinstall

tar -xzvf ilctools-v01-17-02.tar.gz
```

there are two scripts .... release-version.py (that needs to be adapted) and release-sid.cfg, which does contain all the packages you'll need for running SiD software. the adapted scripts are in the myscripts Folder

```
cd /scratch/sid_complete
cp myscripts/release-sid.cfg  ilcinstall/v01-17-02/releases/v01-17/
cp myscripts/release-versions.py ilcinstall/v01-17-02/releases/v01-17/
```

you can test the installation with the -p switch and you can run it with the -i switch

```
cd v01-17-02
./ilcsoft-install  -p releases/v01-17/release-sid.cfg
./ilcsoft-install  -i releases/v01-17/release-sid.cfg
```

You can speed uop the compilation by changing this from -j2 to -j8 on a multi-core machine

```
# global options
ilcsoft.env["MAKEOPTS"]="-j2"
```

Now it's time to grab a coffee...

Known features/problems

- check for lib/lib64 fails, this is true for the following packages:Xerces, Fastjet,HepPDT
  Fix *ln -s lib64 lib*
- Problems in FastJet 2.4.2 compilation

```
fix in 2.4.2/FastJet/include/fastjet/internal/ClusterSequence_N2.icc

109c109
>       if (jetA < jetB) {std::swap(jetA,jetB);}
---
<       if (jetA < jetB) {swap(jetA,jetB);}

2.4.2/FastJet/include/fastjet/NNH.hh

Line 266   if (jetA < jetB) std::swap(jetA,jetB);
Line 270 change to  this->init_jet(jetB, jet, index);
```

LCSIM installation

First get the JAS 3.0.3 from here , it contains a handy LCIO browser and the Wired Event display

unpack it in sid_complete area

```
tar -xzvf jas-assembly-3.0.3-distribution.tar.gz
```

next thing is installing Maven 3.0.5 from the web , unpack and add the path

```
tar -xzvf  apache-maven-3.0.5-bin.tar.gz
export PATH=$PATH:/scratch/sid_complete/apache-maven-3.0.5/bin
```

Then get the lcsim packages via CVS

```
cd /scratch/sid_complete
mkdir lcsim
cd lcsim

cp /scratch/sid_complete/myscripts/build-lcsim.sh .
source ./build-lcsim.sh
```

This will check out everything from the lcsim repository, set the path to Maven correctly and build the lcsim distribution, this is a lot faster than the first step

SLICPandora Installation

this is done using another script

```
cp myscripts/build_slicpandora.sh .
source ./build_slicpandora.sh
```

Note:

in the current cvs version (23/07/13), there are two bugs

```
diff \-r1.13 CMakeLists.txt

27,28c27,28
< FIND_PACKAGE( PandoraSDK REQUIRED )
< FIND_PACKAGE( FineGranularityContent REQUIRED )
\--\-
> FIND_PACKAGE( PandoraPFANew REQUIRED )
> #FIND_PACKAGE( FineGranularityContent REQUIRED )
32c32
< FOREACH( pkg LCIO PandoraSDK  )
\--\-
> FOREACH( pkg LCIO PandoraPFANew )


and then there was one difficulty in
diff -r1.4 MCParticleProcessor.cpp


63c63
mcParticleParameters.m_mcParticleType = pandora::MC_STANDARD;
\--\-
> //mcParticleParameters.m_mcParticleType = pandora::MC_STANDARD;
```

this has been reported

that's it, you're done, now up to actually running a few events ?! Continue reading here

References

- LCSIM XML Confluence
- SID Reco Twiki
- SLICPandora
- Building the LCSim Software Toolchain from Scratch