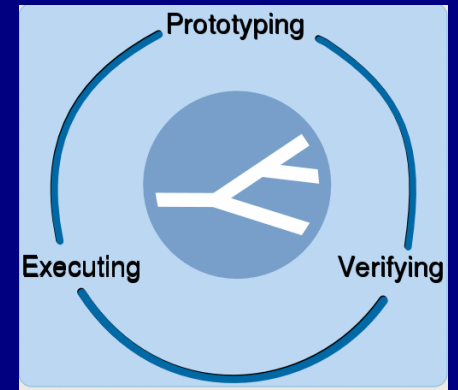


CMS and ILC Applications within the Visual Physics Analysis Project

Tatsiana Klimkovich for the **VISPA** group

M.Brodski, M.Erdmann, R.Fischer, A.Hinzmann, T.Klimkovich, D.Klingebiel,
M.Komm, G.Müller, T.Münzer, J.Steggemann, T.Winchen

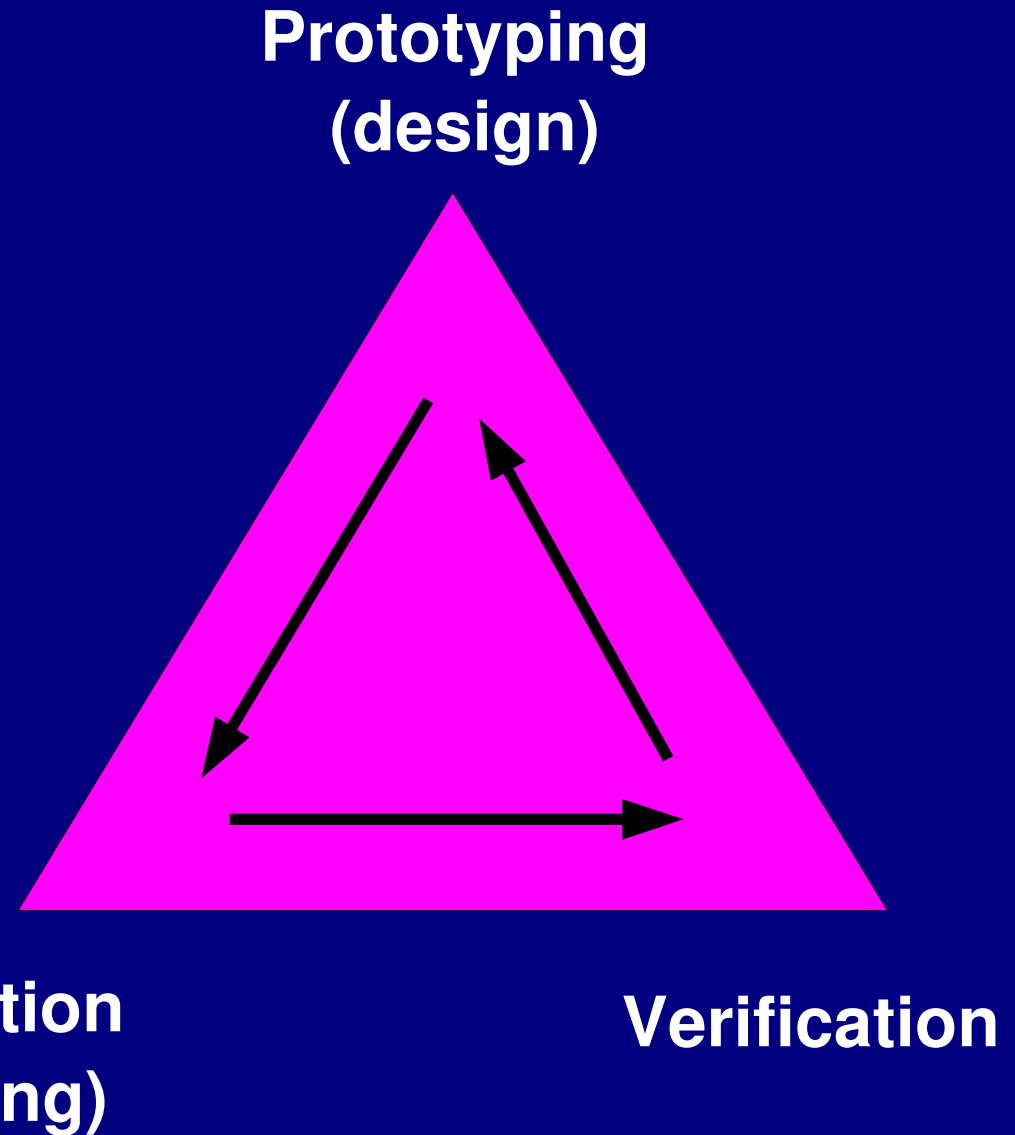
Workshop of the Helmholtz Alliance "Physics at the Terascale"
Hamburg, November 2009



Contents

- **Physics Analysis in High Energy Physics experiment**
- **VISPA Applications at CMS and ILC**
- **Look & feel with analysis example from CMS**

High Energy Physics Analysis



Wish List of the Analyser

- **To start fast**
- **Small summary data sets (ntuples)**
- **Modular structure of the analysis**
- **Many reusable components**
- **Reduced time for analysis cycle**
- **Perform analysis on the laptop (including MAC)**

VISPA: Visual Physics Analysis

Mixture of graphical and textual programming (like **LabView**)

The screenshot shows the VISPA graphical user interface. On the left is a 'Tree View' with 'Reconstructed' (Muon, Jet, MET) and 'Generated' (p+, u, t, W+, mu+, nu_mu, b, d) categories. The main area is a 'Line Decay View' showing a particle decay chain: p+ decays into u and u, which then decay into t and d. The t quark decays into W+ and b, and the W+ decays into mu+ and nu_mu. A 'Property' table on the right shows event details.

Property	Value
Object info	
Type	Event
Id	cb3d4c6841e7
UserRecord	
Event Number	22
Process	gleTop_Channel

The screenshot shows the graphical programming environment for VISPA. It features a central workspace with a flowchart of analysis modules. On the left is an 'Available Modules' list, and on the right is a 'Property' table for the selected module.

Available Modules:

- PySwitch
- PyModule
- PyGenerator
- PyDecide
- PyAnalyse
- File Output
- File Input
- AutoProcess

Python Scripts:

- analysis
- Andreas
- archive
- backup
- bin
- boss
- btag
- CMS_literature
- Desktop
- f2projekt
- fireworks
- for_Andreas
- for_naf
- forDennis
- formulars
- ilC_old
- ilC_samples
- Lehre
- literature
- local
- Mail
- mail

Property Table:

Property	Value
Main	
Name	selection module
Type	PyModule
Options	
filename	/selection_module.py
script	
parameter	
sinks	2
sources	3

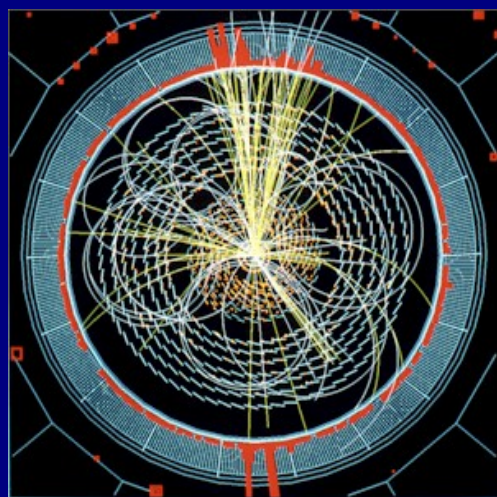
VISPA: Development environment for HEP analyses

Graphical part

- Multi-purpose window
- Visualisation of analysis data and analysis flow in one Graphical User Interface (GUI)
- Module steering

Underlying Software PXL

- PXL is a C++ toolkit for high-level physics analysis
- Version 2.5.2 (October 2009)
- Successor of **PAX** (Physics Analysis Expert) (2002-2007)

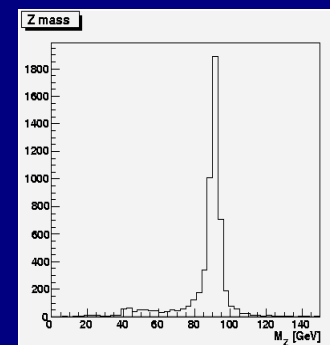


Tatsiana Klimkovich

Experiment
Software
Framework

PXL

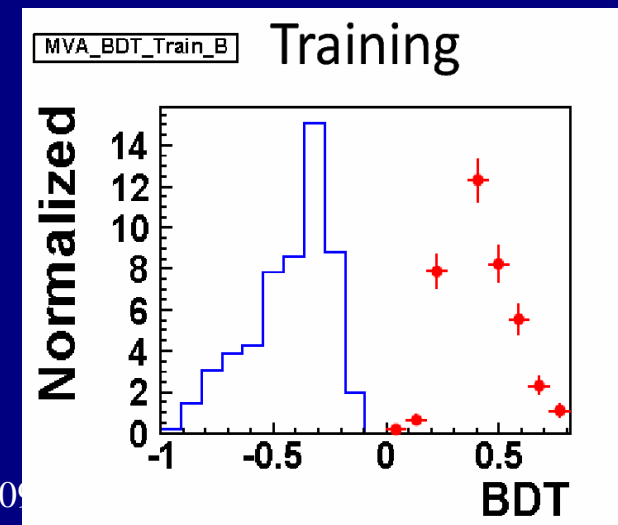
VISPA 



Python Interface to PXL

Python extension **PyPXL**: enables the usage of all PXL objects and their methods within Python programs:

- Python code is easy to read
- Less code compared to C++
- Dynamic typing
- Automatic memory management
- Use of **SWIG** for automatic **interface of C++ to Python**
- Histogramming: **PyROOT**



VISPA graphical part: Common Layout of VISPA Window

The screenshot displays the VISPA graphical user interface. The main window is titled "Line Decay View" and shows a particle decay diagram. The diagram illustrates a top quark (t) decaying into a bottom quark (b) and a W boson. The W boson then decays into two quarks (q). The top quark is represented by a green line, the bottom quark by a blue line, and the W bosons by red dashed lines. The quarks are represented by black lines. The diagram is labeled "Generator" at the top.

On the left side, there is a "Tree View" panel showing a hierarchical structure of the event. The "Event" is expanded to show "Reconstructed" and "Generator" objects.

On the right side, there is a "Property" table showing the properties of the selected object. The table has two columns: "Property" and "Value".

Property	Value
Object info	
Name	t
Type	Particle
Charge	0
ParticleId	0
Locked	<input type="checkbox"/>
Workflag	0
Id	46-97f7-4495195c09a7
Vector	
E	344.6272489
Px	203.2473326
Py	-124.5706522
Pz	-177.1613375
Mass	174.7983856
Pt	238.3848267
Eta	-0.6876772046
Phi	-0.5498521328
P	297.0075169
Et	276.6054808
Theta	2.20991428
SoftRelations	
UserRecord	
Name	default
Pdgid	6

**Navigator
window**

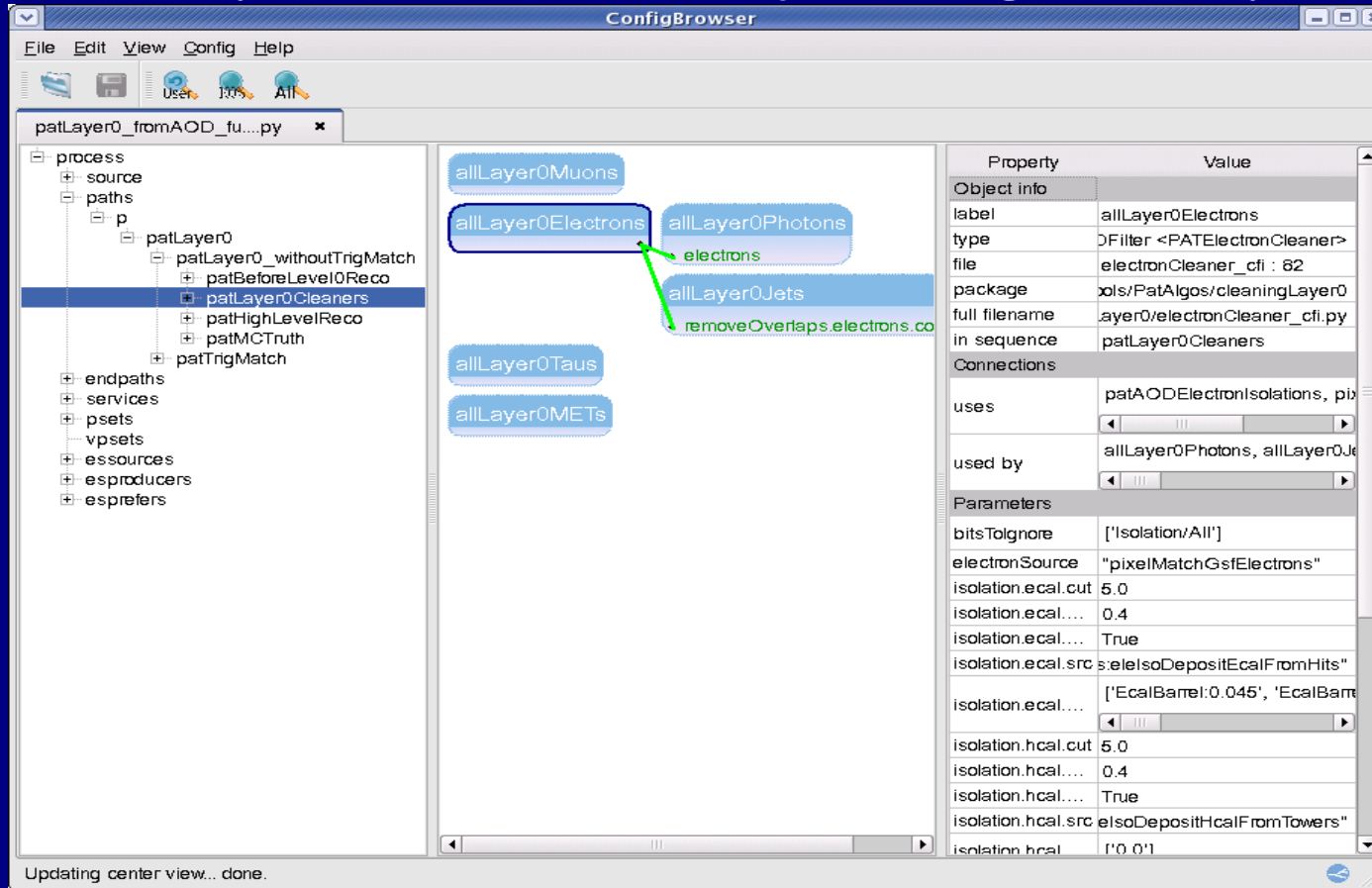
**Graphical
window**

**Property
window**

Updating property view... done.

VISPA graphical part: applications at CMS

- **CMS Configuration Browser** uses VISPA GUI as a platform
 - Allows visually browse and edit the job configuration system of CMS



- **EDM Browser**: an event data browser at CMS, developed as VISPA GUI plugin
 - Allows to inspect the content of CMS data files, event by event

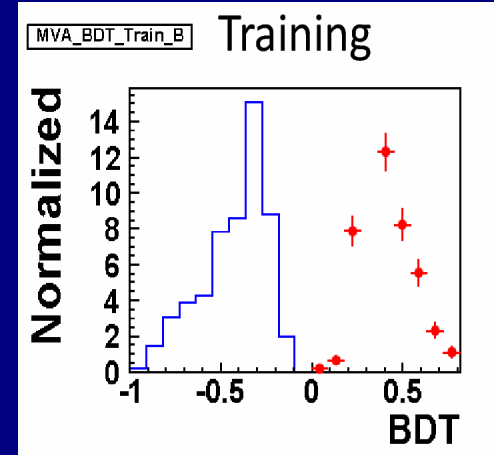
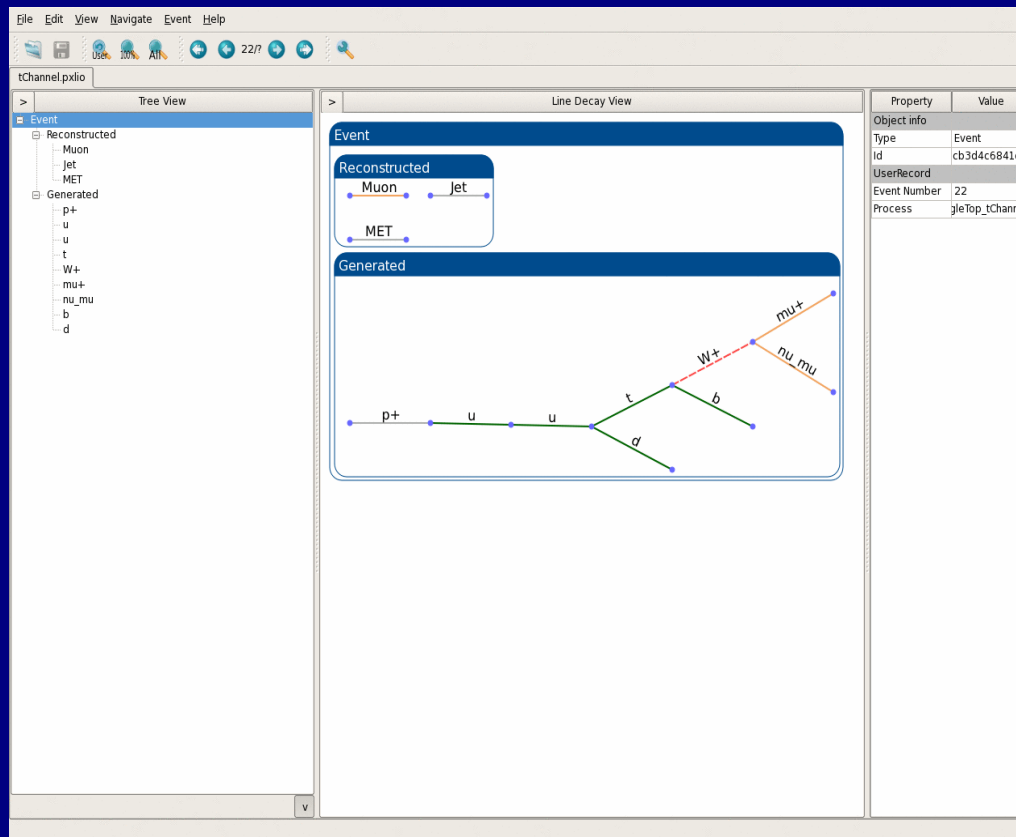
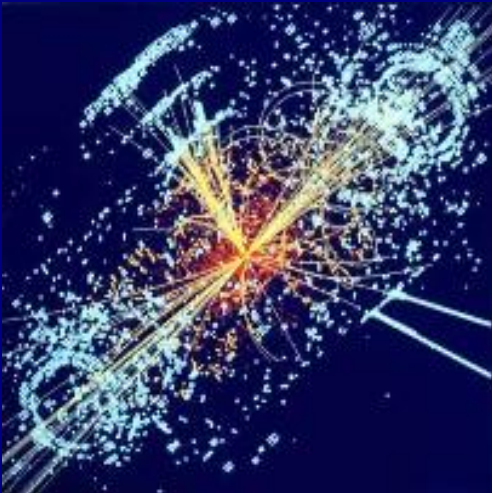
Structure of Physics Analysis

Data input

Data selection

Advanced analysis

Histograms



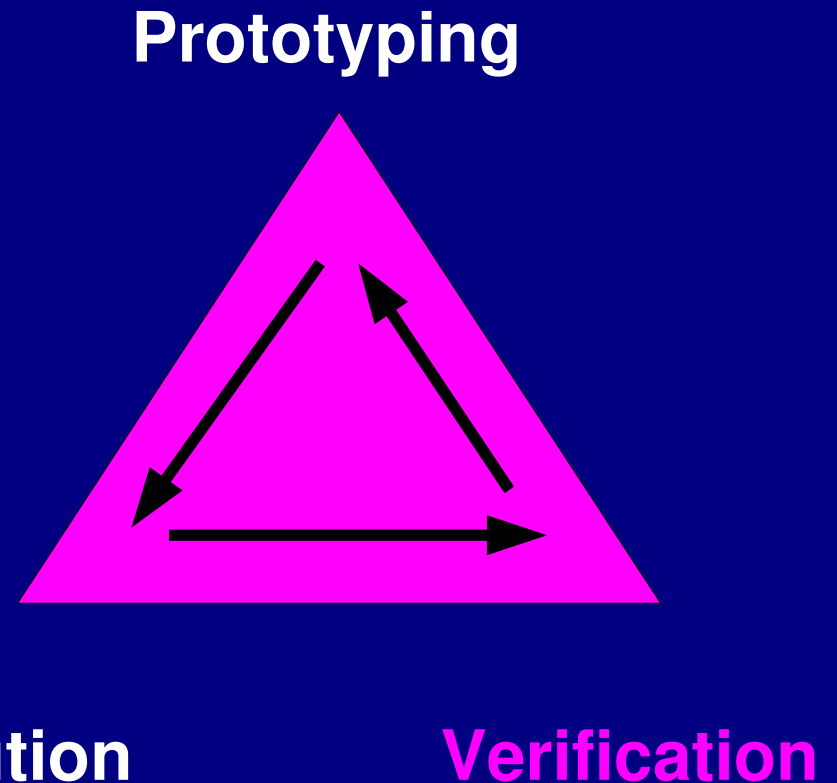
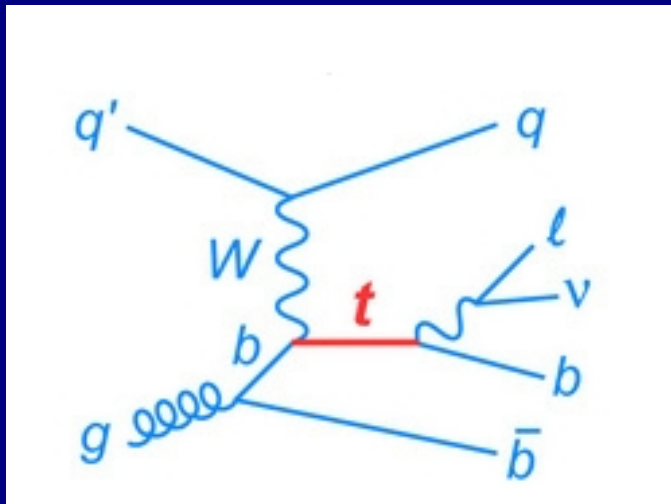
VISPA Application for the ILC Analysis

- VISPA can be used for **ILC analyses**
- **LCIO format** files can be converted into **pxlio format**
- **Converter** implemented as a **Marlin processor**

The screenshot shows the VISPA application interface. The main window displays a 'Line Decay View' with a grid of particles. The particles are arranged in a grid, with labels such as pi-, positron, electron, muon, muonbar, gamma, K0, and Jet. The 'Generated' section at the bottom shows an 'electron' particle. On the right side, there is a 'Property Value' table with the following data:

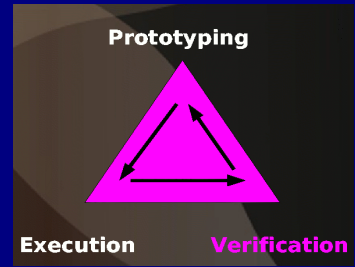
Property	Value
Object info	
Name	Jet
Type	Particle
Charge	0
ParticleId	245942
Locked	<input type="checkbox"/>
Workflag	0
Id	9-27b1fce1e114
Vector	
E	88.58421326
Px	37.80883789
Py	-20.53215027
Pz	-73.17150116
Mass	25.33805121
Pt	43.02414923
Eta	-1.301179181
Phi	-0.4974929069
P	84.88313142
Et	44.90009201
Theta	2.610050454
SoftRelations	
UserRecord	

Look closer: Single Top analysis at CMS



Single Top Analysis

First step: inspect an input file



The screenshot displays the tChannel software interface with the following components:

- Tree View:** A hierarchical tree structure on the left side, categorized into 'Reconstructed' (Muon, Jet, MET) and 'Generated' (p+, u, t, W+, mu+, nu_mu, b, d).
- Line Decay View:** A central diagram showing the decay chain of the event. It includes a 'Reconstructed' section with Muon, Jet, and MET, and a 'Generated' section showing the particle flow from p+ and u quarks through a top quark (t) to a W+ boson, which then decays into mu+ and nu_mu.
- Properties Panel:** A table on the right side listing event details.

Property	Value
Object info	
Type	Event
Id	cb3d4c6841e7
UserRecord	
Event Number	22
Process	gleTop_tChannel

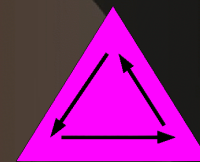
Reconstructed and generated levels

Properties

Single Top Modular Analysis

BDT Training: Single Top vs Background

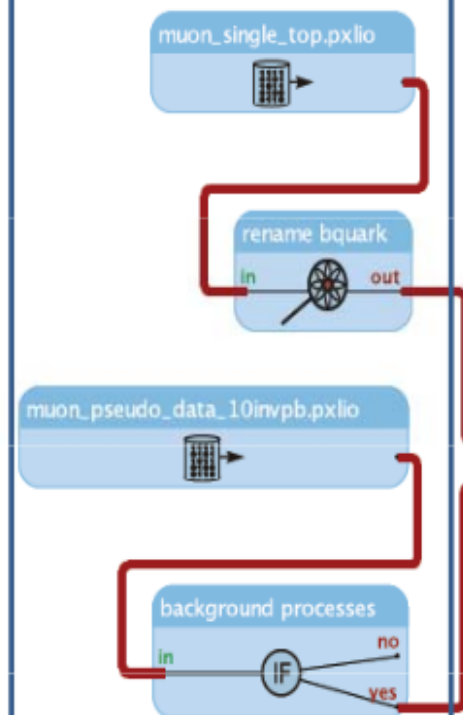
Prototyping



Execution

Verification

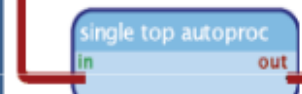
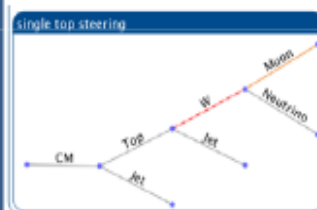
Input & Prepare



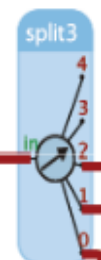
Select



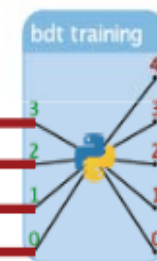
Reconstr. Single Top



Split

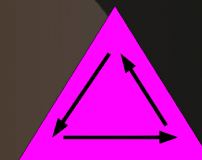


Evaluation BDT single top rec. & Training S/B



Use GUI to design analysis

Prototyping

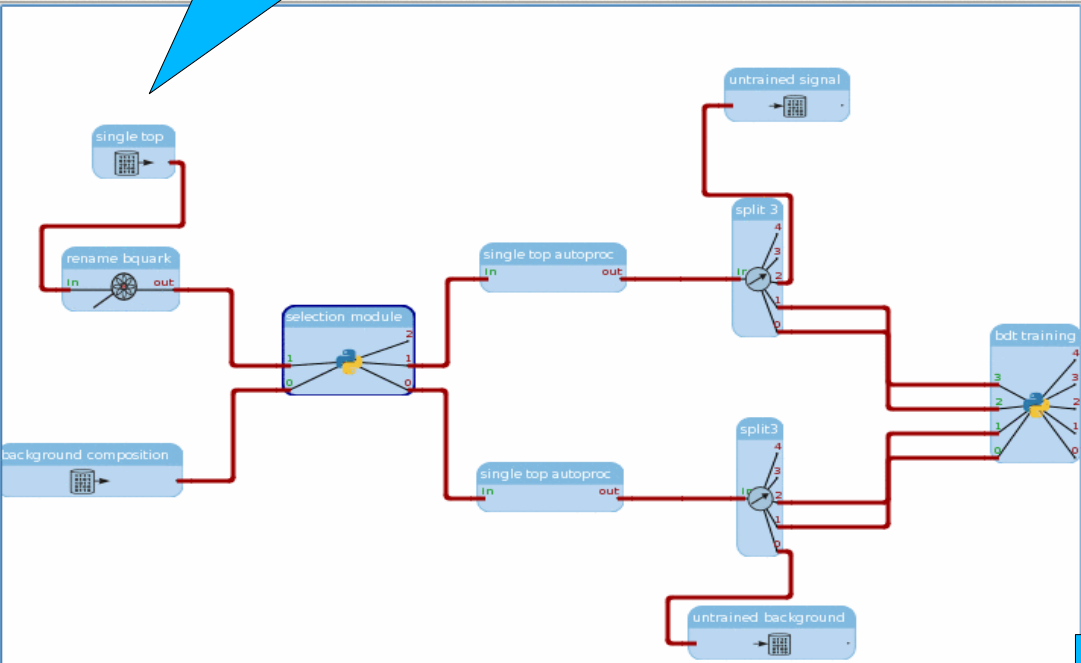


Execution

Verification

Input file

Choose module



Insert and connect modules

Configure module

Property	Value
Main	
Name	selection module
Type	PyModule
Options	
filename	/selection_module.py
script	
parameter	
sinks	2
sources	3

File Edit Analysis Designer View Help

*bdt_training_single_....xml

Available Modules

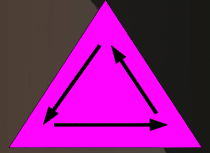
- PySwitch
- PyModule
- PyGenerator
- PyDecide
- PyAnalyse
- File Output
- File Input
- AutoProcess

Python Scripts

- analysis
- Andreas
- archive
- backup
- bin
- boss
- btag
- CMS_literature
- Desktop
- f2projekt
- fireworks
- for_Andreas
- for_naf
- forDennis
- formulars
- ILC_old
- ILC_samples
- Lehre
- literature
- local
- Mail
- mail

Create Python Analysis Module

Prototyping



Execution

Verification

The screenshot shows the Analysis Designer interface with a workflow diagram. The workflow starts with a 'background composition' block, which feeds into a 'selection module'. The 'selection module' has two outputs: 'single top' and 'single top autoproc'. The 'single top' output goes through a 'rename bquark' block. The 'single top autoproc' output goes through a 'split: 3' block, which then feeds into 'untrained signal' and 'untrained background' blocks. The 'single top autoproc' output also goes through another 'split: 3' block, which then feeds into 'bd: training'. The 'bd: training' block has multiple outputs, including 'single top' and 'single top autoproc'. A large blue arrow points to the 'selection module' block, with the text 'Edit user analysis' overlaid on it.

Available Modules:

- PySwitch
- PyModule
- PyGenerator
- PyDecide
- PyAnalyse
- File Output
- File Input
- AutoProcess

Python Scripts:

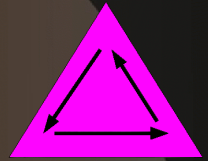
- analysis
- Andreas
- archive
- backup
- bin
- boss
- btag
- CMS_literature
- Desktop
- f2projekt
- fireworks
- for_Andreas
- for_naf
- forDennis
- formulars
- ILC_old
- ILC_samples
- Lehre
- literature
- local
- Mail
- mail

Property	Value
Main	
Name	selection module
Type	PyModule
Options	
filename	/selection_module.py
script	
parameter	
sinks	2
sources	3

Edit user analysis

Create Python Analysis Module

Prototyping



Execution Verification

File Edit Analysis Designer View Help

*bdt_training_single_....xml

Available Modules

- PySwitch
- PyModule
- PyGenerator
- PyDecide
- PyAnalyse
- File Output
- File Input
- AutoProcess

Python Scripts

- analysis
- Andreas
- archive
- backup
- bin
- boss
- btag
- CMS_literature
- Desktop
- f2projekt
- fireworks
- for_Andreas
- for_naf
- forDennis
- formulars
- ILC_old
- ILC_samples
- Lehre
- literature
- local
- Mail
- mail

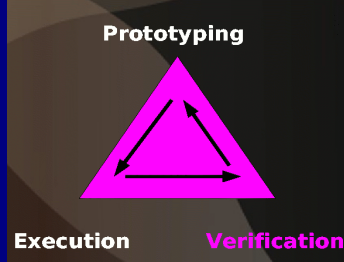
Property Value

Property	Value
Main	
Name	selection module
Type	PyModule
Options	
filename	/selection_module.py
script	
parameter	
sinks	2
sources	3

```
for particle in eventview.getParticles():  
    if (particle.getName() == "Jet" and particle.getPt() > 30)  
        selected.setObject(particle)
```

Rapid prototyping of the analysis

Verify Analysis Output



vispa - /home/home1/institut_3a/klingebiel/SingleTop/erdmann/singletop/mc_untrained_signal_single_top_bkg.pxlio

File Edit View Navigate Event Help

bdt_training_single_...xml | bdt_evaluate_single_...xml | mc_untrained_signal_...pxlio

Tree View

- Event
 - Reconstructed
 - Muon
 - Jet
 - Jet
 - Jet low-pt
 - MET
 - Generated
 - p+
 - g
 - dbar
 - t
 - W+
 - mu+
 - nu_mu
 - b_top
 - ubar
 - AutoProcess
 - Muon
 - MET
 - W
 - Jet
 - Top
 - Jet
 - CM
 - AutoProcess
 - Muon
 - MET
 - W
 - Jet
 - Top
 - Jet
 - CM
 - AutoProcess
 - Muon
 - MET
 - W
 - Jet
 - Top
 - Jet
 - CM
 - AutoProcess
 - Muon
 - MET
 - W
 - Jet
 - Top
 - Jet
 - CM

Line Decay View

Event

Reconstructed

Generated

AutoProcess

AutoProcess

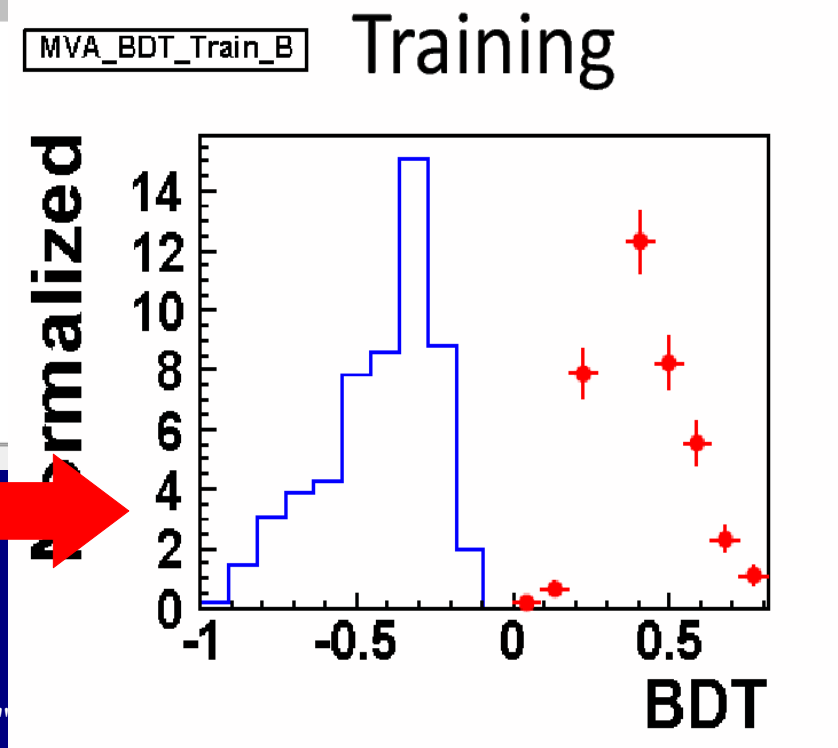
AutoProcess

AutoProcess

Property Value

Property	Value
Object info	
Name	Top
Type	Particle
Charge	0
ParticleId	0
Locked	<input type="checkbox"/>
Workflag	0
Id	f8c47bb7-0a15-5b44-9842-a4919c753c31
Vector	
E	1099.843938
Px	36.31888843
Py	-62.78426354
Pz	1084.199837
Mass	170.0190489
Pt	72.53223701
Eta	3.398830592
Phi	-1.046357259
P	1086.623307
Et	73.41471573
Theta	0.06679978884
SoftRelations	
UserRecord	

Updating property view... done.



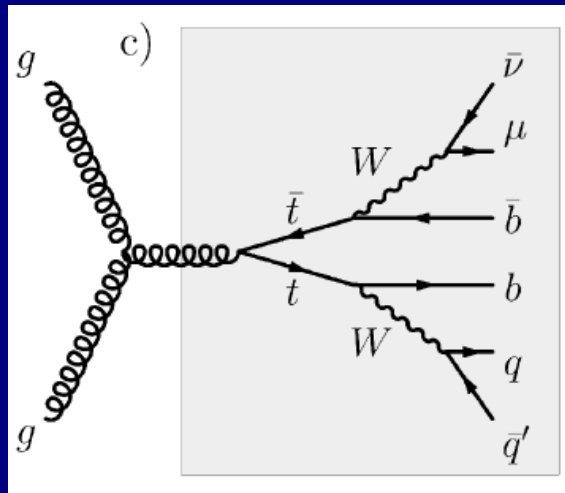
Create histogram using PyROOT
 If needed repeat prototyping,
 execution, verification

VISPA Module: Autoprocess

Steering
event container

Data event
container

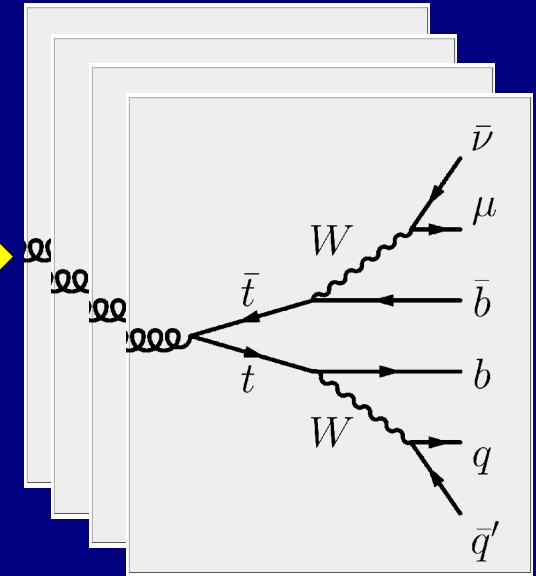
Output event
containers



+

Jets
Muons
MET

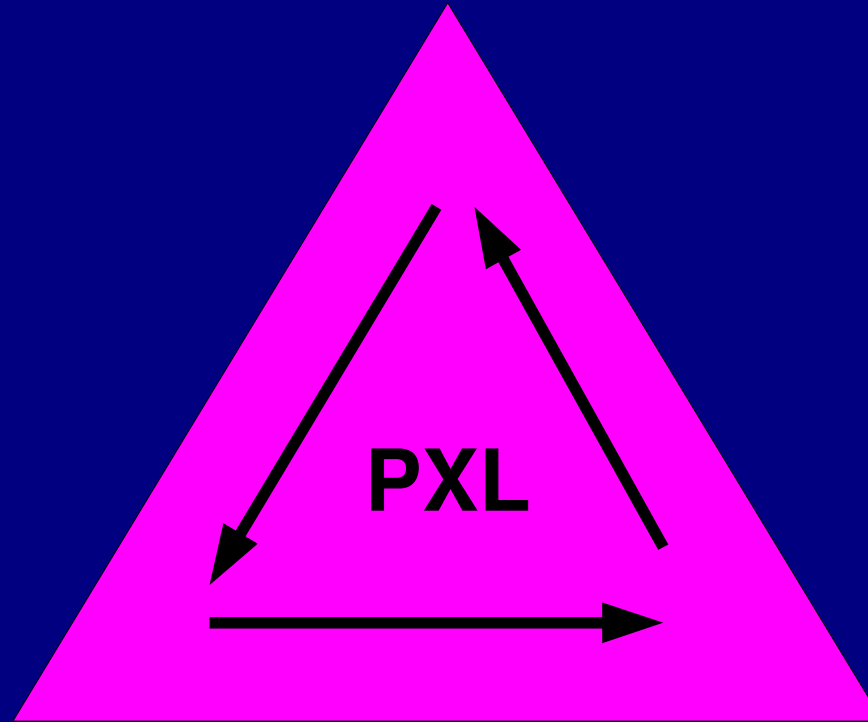
Autoprocess
algorithm



- In various physics analyses (Top, Higgs, SUSY) a reconstruction of the whole decay chain is needed
- Several possible configurations need to be built
- **Autoprocess** is a module for automated reconstruction of particle cascades

Summary: Analysis flow with VISPA

Prototyping: Development Environment
Analysis modules: Python or C++



Execution

XML or Python steering
interactive or batch

Verification

Event Browser
ROOT histograms

Summary

- **VISPA** is a **development environment** for high energy physics analyses
 - Combines **visual and textual** programming
 - Allows fast **prototyping, execution, and verification** of an analysis
 - For application in **any HEP and astroparticle experiment**
 - First applications for **CMS analyses**
 - First steps for applications in **ILC analyses**

Summary II

- All software is continuously maintained and **fully documented**:

<http://vispa.sourceforge.net>

<http://pxl.sourceforge.net>

- Installers for **MS Windows, MAC OS X, Debian and Ubuntu Linux** are provided

- **Publications:**

<http://arxiv.org/abs/0810.3609>

<http://arxiv.org/abs/0801.1302v2>

Backup slides

PXL key components: Event Container

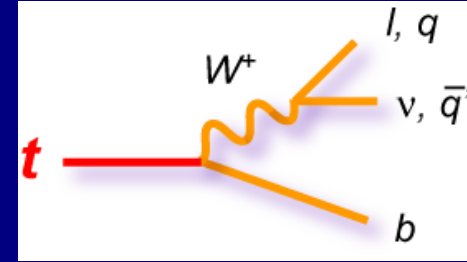
Event container *pxl::Event*

- Particles (*pxl::Particle*)
- Vertices (*pxl::Vertex*)
- Collisions (*pxl::Collision*)
- User data (*pxl::UserRecord*)
- Their relations and roles

Physics
objects

Event Interpretation

pxl::EventView



pxl::Event represents an entire physics event

pxl::Event can hold several **EventViews**

pxl::EventView is a special view of this event

Copies of these classes preserve all contained information (e.g. particle relations)

PXL key components: *pxl::UserRecord*

- All major PXL objects provide **UserRecord** for storage of arbitrary user data
- Deploys **Copy-On-Write** mechanism

PXL key components: I/O System *pxl::Serializable*

- **Fast, flexible**
- **Small file size (use **ZLIB** library for data compression)**

Module Steering System

- **Data flow**

- each module has a number of **sources and sinks**
- interface between modules: **PXL event container**

- **Modules**

- plug-in mechanism
- interactive creation of PYTHON modules

- **XML configuration**

- exchange format
- save and restore any state of the analysis

- **PYTHON configuration**

- high flexibility
- easy-to-read

Interface: Event Container

