GammaLib - Feature #490

Avoid casts for derived classes

09/19/2012 08:50 PM - Knödlseder Jürgen

Status:	Closed	**Start date:**	09/20/2012
Priority:	Normal	**Due date:**	
Assigned To:	Knödlseder Jürgen	**% Done:**	43%
Category:		**Estimated time:**	0.00 hour
Target version:	00-07-00		

Description

Subtasks:

Action # 501: Rework GOptimizer interface	**Closed**
Action # 500: Rework GXmlNode interface	**Closed**
Action # 498: Rework GRoi interface	**Closed**
Action # 497: Rework GPointing interface	**Closed**
Action # 495: Rework GResponse interface	**Closed**
Action # 494: Rework GObservation interface	**Closed**
Action # 493: Rework GInstDir interface	**Closed**
Action # 492: Rework GEvents interface	**Closed**
Action # 491: Rework GEvent interface	**Closed**
Action # 503: Rework GModelSpectral interface	**Closed**
Action # 499: Rework GModel interface	**Closed**
Action # 504: Rework GModelTemporal interface	**Closed**
Action # 502: Rework GModelSpatial interface	**Closed**
Action # 496: Rework GFitsImage interface	**Closed**

History

#1 - 09/19/2012 08:53 PM - Knödlseder Jürgen

- Priority changed from Normal to Urgent

- Target version set to 00-07-00

- Estimated time set to 40.00

The Python module includes a couple of cast functions that allow converting from base class pointers to derived class pointers. The necessity for having these casts points to a poorly defined base class interface which should be avoided in the future.

The relevant base class interfaces should be modified so that casts are no longer necessary.

#2 - 09/20/2012 01:00 AM - Knödlseder Jürgen

Here a list of files where Python casts are actually used in GammaLib:

```
inst/cta/test/example_make_model.py:    events = cast_GCTAEventCube(obs.events())
inst/cta/test/example_sim_photons.py:    model  = cast_GModelSky(models[0])
inst/cta/test/example_sim_photons.py:       crab   = cast_GModelSky(models[0])
inst/cta/test/test_irf_offset.py:       bin = cast_GCTAEventBin(bin)
inst/cta/test/test_model.py:       bin = cast_GCTAEventBin(bin)
inst/cta/test/test_model.py:       bin = cast_GCTAEventBin(bin)
inst/cta/test/test_radial_acceptance.py:    radial = cast_GCTAModelRadialAcceptance(models["Background"]).radial()
inst/lat/test/test_python.py:       cube = cast_GLATEventCube(run.events())
inst/mwl/test/test_python.py:           crab = cast_GModelSky(model)
test/test_GFits.py:       img_double = cast_GFitsImageDouble(fits.image("Double"))
test/test_GFits.py:       #img_byte = cast_GFitsImageByte(fits.image("Double"))
```

And here a list of files where Python casts are actually used in ctools:

```
examples/make_binned_analysis.py:           cube = cast_GCTAEventCube(obs.events())
examples/make_binned_analysis.py:           cube = cast_GCTAEventCube(obs.events())
examples/make_unbinned_analysis.py:            list = cast_GCTAEventList(obs.events())
examples/make_unbinned_analysis.py:            list = cast_GCTAEventList(obs.events())
scripts/cssens.py:          pflux     = cast_GModelSky(result_all["Test"]).spectral().flux(emin, emax)
scripts/cssens.py:          eflux     = cast_GModelSky(result_all["Test"]).spectral().eflux(emin, emax)
scripts/obsutils.py:         event = cast_GCTAEventAtom(event)
test/example_survey.py:          list = cast_GCTAEventList(run.events())
test/example_survey.py:          list = cast_GCTAEventList(run.events())
test/example_survey.py:          event = cast_GCTAEventAtom(event)
```

And here a list of other scripts that I currently use on my Mac (in the cta folder):

```
dc1/results-v0/example_unbinned_ml_fit.py:       obs = cast_GCTAObservation(obs)
dc1/results-v0/example_unbinned_ml_fit.py:       rois.append(cast_GCTAEventList(obs.events()).roi())
dc1/results-v0/example_unbinned_ml_fit.py:       obs = cast_GCTAObservation(obs)
dc1/results-v0/example_unbinned_ml_fit.py:       cube = cast_GCTAEventCube(obs.events())
dc1/results-v0/example_unbinned_ml_fit.py:       rois.append(cast_GCTAEventList(obs.events()).roi())
dc1/results-v0/example_unbinned_ml_fit.py:       obs = cast_GCTAObservation(obs)
dc1/results-v0/example_unbinned_ml_fit.py:       events = cast_GCTAEventCube(obs.events())
dc1-analysis/fermi/show_spectrum.py:       source   = cast_GXmlElement(library.element("source", i))
dc1-analysis/fermi/show_spectrum.py:       spectrum = cast_GXmlElement(source.element("spectrum", 0))
dc1-analysis/fermi/show_spectrum.py:          node         = cast_GXmlElement(spectrum.element("node", k))
dc1-analysis/fermi/show_spectrum.py:          par_energy    = cast_GXmlElement(node.element("parameter", 0))
dc1-analysis/fermi/show_spectrum.py:          par_intensity = cast_GXmlElement(node.element("parameter", 1))
dc1-analysis/fermi-hess/show_spectrum.py:       source   = cast_GXmlElement(library.element("source", i))
dc1-analysis/fermi-hess/show_spectrum.py:       spectrum = cast_GXmlElement(source.element("spectrum", 0))
dc1-analysis/fermi-hess/show_spectrum.py:          node         = cast_GXmlElement(spectrum.element("node", k))
dc1-analysis/fermi-hess/show_spectrum.py:          par_energy    = cast_GXmlElement(node.element("parameter", 0))
dc1-analysis/fermi-hess/show_spectrum.py:          par_intensity = cast_GXmlElement(node.element("parameter", 1))
dc1-analysis/fermi-magic/show_spectrum.py:       source   = cast_GXmlElement(library.element("source", i))
dc1-analysis/fermi-magic/show_spectrum.py:       spectrum = cast_GXmlElement(source.element("spectrum", 0))
dc1-analysis/fermi-magic/show_spectrum.py:          node         = cast_GXmlElement(spectrum.element("node", k))
dc1-analysis/fermi-magic/show_spectrum.py:          par_energy    = cast_GXmlElement(node.element("parameter", 0))
dc1-analysis/fermi-magic/show_spectrum.py:          par_intensity = cast_GXmlElement(node.element("parameter", 1))
dc1-analysis/fermi-magic-hess/show_spectrum.py:       source   = cast_GXmlElement(library.element("source", i))
dc1-analysis/fermi-magic-hess/show_spectrum.py:       spectrum = cast_GXmlElement(source.element("spectrum", 0))
dc1-analysis/fermi-magic-hess/show_spectrum.py:          node         = cast_GXmlElement(spectrum.element("node", k))
dc1-analysis/fermi-magic-hess/show_spectrum.py:          par_energy    = cast_GXmlElement(node.element("parameter", 0))
dc1-analysis/fermi-magic-hess/show_spectrum.py:          par_intensity = cast_GXmlElement(node.element("parameter", 1))
dc1-analysis/hess-magic/analyse_crab_spectrum.py:       rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-analysis/hess-magic/analyse_crab_spectrum.py:       cube = cast_GCTAEventCube(obs.events())
dc1-analysis/hess-magic/analyse_crab_spectrum.py:       rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-analysis/hess-magic/analyse_crab_spectrum.py:       events = cast_GCTAEventCube(obs.events())
dc1-analysis/hess-magic/analyse_crab_unbinned.py:       rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-analysis/hess-magic/analyse_crab_unbinned.py:       cube = cast_GCTAEventCube(obs.events())
dc1-analysis/hess-magic/analyse_crab_unbinned.py:       rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-analysis/hess-magic/analyse_crab_unbinned.py:       events = cast_GCTAEventCube(obs.events())
dc1-analysis/hess-magic/show_spectrum.py:       source   = cast_GXmlElement(library.element("source", i))
dc1-analysis/hess-magic/show_spectrum.py:       spectrum = cast_GXmlElement(source.element("spectrum", 0))
dc1-analysis/hess-magic/show_spectrum.py:          node         = cast_GXmlElement(spectrum.element("node", k))
dc1-analysis/hess-magic/show_spectrum.py:          par_energy    = cast_GXmlElement(node.element("parameter", 0))
dc1-analysis/hess-magic/show_spectrum.py:          par_intensity = cast_GXmlElement(node.element("parameter", 1))
dc1-analysis/hess.old/analyse.py:       obs = cast_GCTAObservation(obs)
dc1-analysis/hess.old/analyse.py:       rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-analysis/hess.old/analyse.py:       cube = cast_GCTAEventCube(obs.events())
dc1-analysis/hess.old/analyse.py:       rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-analysis/hess.old/analyse.py:       obs = cast_GCTAObservation(obs)
dc1-analysis/hess.old/analyse.py:       events = cast_GCTAEventCube(obs.events())
dc1-analysis/magic/show_spectrum.py:       source   = cast_GXmlElement(library.element("source", i))
dc1-analysis/magic/show_spectrum.py:       spectrum = cast_GXmlElement(source.element("spectrum", 0))
dc1-analysis/magic/show_spectrum.py:          node         = cast_GXmlElement(spectrum.element("node", k))
dc1-analysis/magic/show_spectrum.py:          par_energy    = cast_GXmlElement(node.element("parameter", 0))
dc1-analysis/magic/show_spectrum.py:          par_intensity = cast_GXmlElement(node.element("parameter", 1))
dc1-analysis/magic.old/analyse.py:       obs = cast_GCTAObservation(obs)
dc1-analysis/magic.old/analyse.py:       rois.append(cast_GCTAEventList(obs.events()).roi())
```

```
dc1-analysis/magic.old/analyse.py:      cube = cast_GCTAEventCube(obs.events())
dc1-analysis/magic.old/analyse.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-analysis/magic.old/analyse.py:      obs = cast_GCTAObservation(obs)
dc1-analysis/magic.old/analyse.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials/hess/analyse.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials/hess/analyse.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials/hess/analyse.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials/hess/analyse.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials/hess/analyse.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials/hess/analyse.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials/hess-magic/analyse_crab_spectrum.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials/hess-magic/analyse_crab_spectrum.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials/hess-magic/analyse_crab_spectrum.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials/hess-magic/analyse_crab_spectrum.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials/hess-magic/analyse_crab_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials/hess-magic/analyse_crab_unbinned.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials/hess-magic/analyse_crab_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials/hess-magic/analyse_crab_unbinned.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials/hess-magic/show_spectrum.py:      source   = cast_GXmlElement(library.element("source", i))
dc1-tutorials/hess-magic/show_spectrum.py:      spectrum = cast_GXmlElement(source.element("spectrum", 0))
dc1-tutorials/hess-magic/show_spectrum.py:         node         = cast_GXmlElement(spectrum.element("node", k))
dc1-tutorials/hess-magic/show_spectrum.py:         par_energy   = cast_GXmlElement(node.element("parameter", 0))
dc1-tutorials/hess-magic/show_spectrum.py:         par_intensity = cast_GXmlElement(node.element("parameter", 1))
dc1-tutorials/magic/analyse.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials/magic/analyse.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials/magic/analyse.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials/magic/analyse.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials/magic/analyse.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials/magic/analyse.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/gaussian/analyse_unbinned.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/gaussian/analyse_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/gaussian/analyse_unbinned.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/gaussian/analyse_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/gaussian/analyse_unbinned.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/gaussian/analyse_unbinned.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/gaussian-test/analyse_unbinned.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/gaussian-test/analyse_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/gaussian-test/analyse_unbinned.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/gaussian-test/analyse_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/gaussian-test/analyse_unbinned.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/gaussian-test/analyse_unbinned.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess/analyse.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess/analyse.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess/analyse.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess/analyse.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess/analyse.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess/analyse.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess/analyse_pks_nodes.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess/analyse_pks_nodes.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess/analyse_pks_nodes.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess/analyse_pks_nodes.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess/analyse_pks_nodes.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess/analyse_pks_nodes.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess/analyse_pks_unbinned.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess/analyse_pks_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess/analyse_pks_unbinned.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess/analyse_pks_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess/analyse_pks_unbinned.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess/analyse_pks_unbinned.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess/show_spectrum.py:      source   = cast_GXmlElement(library.element("source", i))
dc1-tutorials-v0/hess/show_spectrum.py:      spectrum = cast_GXmlElement(source.element("spectrum", 0))
dc1-tutorials-v0/hess/show_spectrum.py:         node         = cast_GXmlElement(spectrum.element("node", k))
dc1-tutorials-v0/hess/show_spectrum.py:         par_energy   = cast_GXmlElement(node.element("parameter", 0))
dc1-tutorials-v0/hess/show_spectrum.py:         par_intensity = cast_GXmlElement(node.element("parameter", 1))
dc1-tutorials-v0/hess-magic/analyse_crab_binned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-magic/analyse_crab_binned.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-magic/analyse_crab_binned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-magic/analyse_crab_binned.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-magic/analyse_crab_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-magic/analyse_crab_unbinned.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-magic/analyse_crab_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-magic/analyse_crab_unbinned.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-magic/show_spectrum.py:      source   = cast_GXmlElement(library.element("source", i))
dc1-tutorials-v0/hess-magic/show_spectrum.py:      spectrum = cast_GXmlElement(source.element("spectrum", 0))
dc1-tutorials-v0/hess-magic/show_spectrum.py:         node         = cast_GXmlElement(spectrum.element("node", k))
```

```
dc1-tutorials-v0/hess-magic/show_spectrum.py:          par_energy    = cast_GXmlElement(node.element("parameter", 0))
dc1-tutorials-v0/hess-magic/show_spectrum.py:          par_intensity = cast_GXmlElement(node.element("parameter", 1))
dc1-tutorials-v0/hess-old/analyse_crab_gauss.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess-old/analyse_crab_gauss.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-old/analyse_crab_gauss.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-old/analyse_crab_gauss.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-old/analyse_crab_gauss.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess-old/analyse_crab_gauss.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-old/analyse_crab_nodes.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess-old/analyse_crab_nodes.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-old/analyse_crab_nodes.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-old/analyse_crab_nodes.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-old/analyse_crab_nodes.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess-old/analyse_crab_nodes.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-old/analyse_crab_spectrum.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess-old/analyse_crab_spectrum.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-old/analyse_crab_spectrum.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-old/analyse_crab_spectrum.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-old/analyse_crab_spectrum.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess-old/analyse_crab_spectrum.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-old/analyse_crab_spectrum_nodes.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess-old/analyse_crab_spectrum_nodes.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-old/analyse_crab_spectrum_nodes.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-old/analyse_crab_spectrum_nodes.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-old/analyse_crab_spectrum_nodes.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess-old/analyse_crab_spectrum_nodes.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-old/analyse_crab_unbinned.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess-old/analyse_crab_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-old/analyse_crab_unbinned.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-old/analyse_crab_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-old/analyse_crab_unbinned.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess-old/analyse_crab_unbinned.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-old/analyse_pks_nodes.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess-old/analyse_pks_nodes.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-old/analyse_pks_nodes.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-old/analyse_pks_nodes.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-old/analyse_pks_nodes.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess-old/analyse_pks_nodes.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-old/analyse_pks_unbinned.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess-old/analyse_pks_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-old/analyse_pks_unbinned.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-old/analyse_pks_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/hess-old/analyse_pks_unbinned.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/hess-old/analyse_pks_unbinned.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/hess-old/show_spectrum.py:      source   = cast_GXmlElement(library.element("source", i))
dc1-tutorials-v0/hess-old/show_spectrum.py:      spectrum = cast_GXmlElement(source.element("spectrum", 0))
dc1-tutorials-v0/hess-old/show_spectrum.py:          node         = cast_GXmlElement(spectrum.element("node", k))
dc1-tutorials-v0/hess-old/show_spectrum.py:          par_energy    = cast_GXmlElement(node.element("parameter", 0))
dc1-tutorials-v0/hess-old/show_spectrum.py:          par_intensity = cast_GXmlElement(node.element("parameter", 1))
dc1-tutorials-v0/magic/analyse.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/magic/analyse.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/magic/analyse.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/magic/analyse.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/magic/analyse.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/magic/analyse.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/unbinned-test/analyse_crab_nodes.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/unbinned-test/analyse_crab_nodes.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/unbinned-test/analyse_crab_nodes.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/unbinned-test/analyse_crab_nodes.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/unbinned-test/analyse_crab_nodes.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/unbinned-test/analyse_crab_nodes.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/unbinned-test/analyse_crab_unbinned.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/unbinned-test/analyse_crab_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/unbinned-test/analyse_crab_unbinned.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/unbinned-test/analyse_crab_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/unbinned-test/analyse_crab_unbinned.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/unbinned-test/analyse_crab_unbinned.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/unbinned-test/analyse_unbinned.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/unbinned-test/analyse_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/unbinned-test/analyse_unbinned.py:      cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/unbinned-test/analyse_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/unbinned-test/analyse_unbinned.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/unbinned-test/analyse_unbinned.py:      events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/unbinned-v0/analyse_unbinned.py:      obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/unbinned-v0/analyse_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
```

```
dc1-tutorials-v0/unbinned-v0/analyse_unbinned.py:       cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/unbinned-v0/analyse_unbinned.py:       rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/unbinned-v0/analyse_unbinned.py:       obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/unbinned-v0/analyse_unbinned.py:       events = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/unbinned-v1/analyse_unbinned.py:       obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/unbinned-v1/analyse_unbinned.py:       rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/unbinned-v1/analyse_unbinned.py:       cube = cast_GCTAEventCube(obs.events())
dc1-tutorials-v0/unbinned-v1/analyse_unbinned.py:       rois.append(cast_GCTAEventList(obs.events()).roi())
dc1-tutorials-v0/unbinned-v1/analyse_unbinned.py:       obs = cast_GCTAObservation(obs)
dc1-tutorials-v0/unbinned-v1/analyse_unbinned.py:       events = cast_GCTAEventCube(obs.events())
erlangen-tutorials/unbinned/analyse_unbinned.py:      obs = cast_GCTAObservation(obs)
erlangen-tutorials/unbinned/analyse_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
erlangen-tutorials/unbinned/analyse_unbinned.py:      cube = cast_GCTAEventCube(obs.events())
erlangen-tutorials/unbinned/analyse_unbinned.py:      rois.append(cast_GCTAEventList(obs.events()).roi())
erlangen-tutorials/unbinned/analyse_unbinned.py:      obs = cast_GCTAObservation(obs)
erlangen-tutorials/unbinned/analyse_unbinned.py:      events = cast_GCTAEventCube(obs.events())
examples/example_survey.py:       list = cast_GCTAEventList(run.events())
examples/example_survey.py:       list = cast_GCTAEventList(run.events())
examples/example_survey.py:           event = cast_GCTAEventAtom(event)
examples/milagro/example_survey.py:           event = cast_GCTAEventAtom(event)
examples/old/example_survey.py:       list = cast_GCTAEventList(run.events())
examples/old/example_survey.py:       list = cast_GCTAEventList(run.events())
examples/old/example_survey.py:           event = cast_GCTAEventAtom(event)
examples/old/problem_integration_did_not_converge.py:       list = cast_GCTAEventList(run.events())
examples/old/problem_integration_did_not_converge.py:       list = cast_GCTAEventList(run.events())
examples/old/problem_integration_did_not_converge.py:           event = cast_GCTAEventAtom(event)
examples/old/survey_study.py:       list = cast_GCTAEventList(run.events())
examples/old/survey_study.py:       list = cast_GCTAEventList(run.events())
examples/old/survey_study.py:           event = cast_GCTAEventAtom(event)
examples/old/survey_study_jk.py:           event = cast_GCTAEventAtom(event)
examples/old/survey_study_orig.py:       list = cast_GCTAEventList(run.events())
examples/old/survey_study_orig.py:       list = cast_GCTAEventList(run.events())
examples/old/survey_study_orig.py:           event = cast_GCTAEventAtom(event)
examples/old/survey_study_spread.py:           event = cast_GCTAEventAtom(event)
examples/old/survey_study_test.py:       list = cast_GCTAEventList(run.events())
examples/old/survey_study_test.py:       list = cast_GCTAEventList(run.events())
examples/old/survey_study_test.py:           event = cast_GCTAEventAtom(event)
examples/survey/example_survey.py:           event = cast_GCTAEventAtom(event)
examples/survey/example_survey.py~:           event = cast_GCTAEventAtom(event)
examples/survey/integration_convergence.py:           event = cast_GCTAEventAtom(event)
examples/survey/premature_fit_stop.py:           event = cast_GCTAEventAtom(event)
examples/survey/survey_sensitivity_forjurgen.py:           event = cast_GCTAEventAtom(event)
examples/survey/survey_sensitivity_forjurgen.py:                   pflux_save[test_num1]=cast_GModelSky(result_all[1]).spectral().flux(e_min, e_max)
examples/survey/survey_sensitivity_forjurgen.py:                   eflux_save[test_num1]=cast_GModelSky(result_all[1]).spectral().eflux(e_min, e_max)
sensitivity/old/survey_sensitivity_rev0.py:       list = cast_GCTAEventList(run.events())
sensitivity/old/survey_sensitivity_rev0.py:       list = cast_GCTAEventList(run.events())
sensitivity/old/survey_sensitivity_rev0.py:           event = cast_GCTAEventAtom(event)
sensitivity/old/survey_sensitivity_rev1.py:           event = cast_GCTAEventAtom(event)
sensitivity/old/survey_sensitivity_rev2.py:           event = cast_GCTAEventAtom(event)
sensitivity/old/survey_sensitivity_rev3.py:           event = cast_GCTAEventAtom(event)
sensitivity/old/survey_study.py:       list = cast_GCTAEventList(run.events())
sensitivity/old/survey_study.py:       list = cast_GCTAEventList(run.events())
sensitivity/old/survey_study.py:           event = cast_GCTAEventAtom(event)
sensitivity/old/survey_study_energy.py:       pflux = cast_GModelSky(full_model[1]).spectral().flux(e_min, e_max)
sensitivity/old/survey_study_energy.py:       eflux = cast_GModelSky(full_model[1]).spectral().eflux(e_min, e_max)
sensitivity/old/survey_study_orig.py:       list = cast_GCTAEventList(run.events())
sensitivity/old/survey_study_orig.py:       list = cast_GCTAEventList(run.events())
sensitivity/old/survey_study_orig.py:           event = cast_GCTAEventAtom(event)
sensitivity/old/survey_study_test.py:       list = cast_GCTAEventList(run.events())
sensitivity/old/survey_study_test.py:       list = cast_GCTAEventList(run.events())
sensitivity/old/survey_study_test.py:           event = cast_GCTAEventAtom(event)
testing/ctobssim/myutils.py:           event = cast_GCTAEventAtom(event)
testing/ctobssim/openmp.py:           cast_GCTAObservation(obs).response(self.m_irf, self.m_caldb)
testing/openmp/myutils.py:           event = cast_GCTAEventAtom(event)
testing/pierrick/survey_offsetanglestudy.py:           event = cast_GCTAEventAtom(event)
testing/pierrick/survey_offsetanglestudy.py:   rsp=cast_GCTAResponse(obs.response())
testing/pierrick/survey_offsetanglestudy.py:               pflux_save[test_num1]=cast_GModelSky(result_all[1]).spectral().flux(e_min, e_max)
testing/pierrick/survey_offsetanglestudy.py:               eflux_save[test_num1]=cast_GModelSky(result_all[1]).spectral().eflux(e_min, e_max)
```

At the end of the rework, all casts should have disappeared. Being aware of the existing casts may help in the redesign of the interfaces.

#3 - 09/20/2012 01:28 AM - Knödlseder Jürgen

Got it!

I managed to make a dynamic type conversion in SWIG using an output map. Here the code that I added to GModels.i:

```
%typemap(out) GModel& {
    if (dynamic_cast<GModelSky*>($1) != NULL) {
        $result = SWIG_NewPointerObj(SWIG_as_voidptr($1), SWIGTYPE_p_GModelSky, 0 |  0 );
    }
    else {
        $result = SWIG_NewPointerObj(SWIG_as_voidptr($1), SWIGTYPE_p_GModelData, 0 |  0 );
    }
}
```

This is correctly interpreted in the subsequent methods

```
    GModel& __getitem__(const int& index) {
        if (index >= 0 && index < self->size()) {
            return (*self)[index];
        }
        else
            throw GException::out_of_range("__getitem__(int)", index, self->size());
    }
    GModel& __getitem__(const std::string& name) {
        return (*self)[name];
    }
```

#4 - 09/20/2012 05:21 PM - Knödlseder Jürgen

- Due date set to 09/20/2012

- Start date set to 09/20/2012

due to changes in a related task

#5 - 09/20/2012 06:48 PM - Knödlseder Jürgen

- Assigned To set to Knödlseder Jürgen

#6 - 12/04/2012 11:39 PM - Knödlseder Jürgen

- Status changed from New to In Progress

#7 - 12/18/2012 04:47 PM - Knödlseder Jürgen

Here a list of GammaLib scripts from which the cast method could not be removed:

inst/cta/test/test_radial_acceptance.py: radial = cast_GCTAModelRadialAcceptance(models["Background"]).radial()

#8 - 12/18/2012 09:46 PM - Knödlseder Jürgen

- Status changed from In Progress to Closed