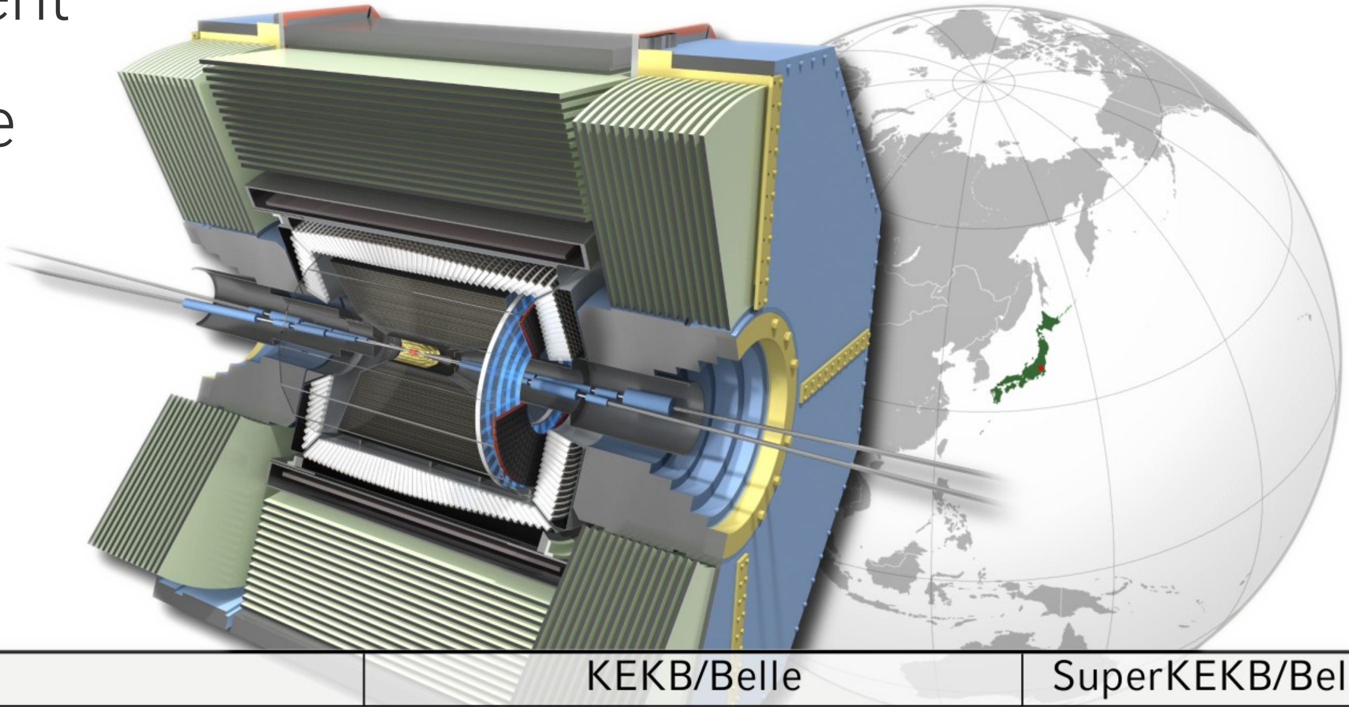
Belle 2 framework efforts

Hadrien Grasland, Martin Ritter 2020-04-13

Belle 2 experiment

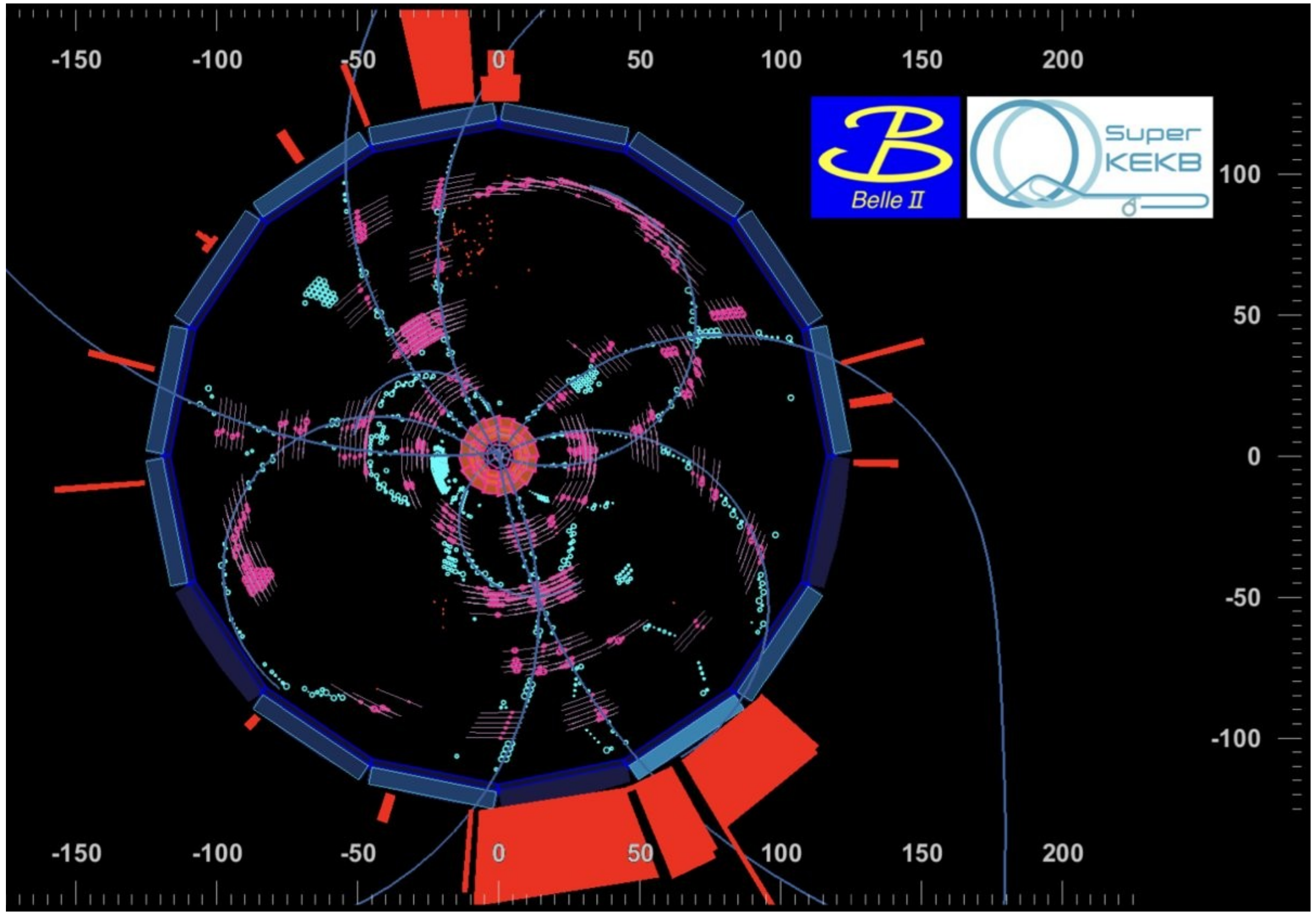
- Asymmetric e^+e^- experiment
- Mainly at Υ(4S) resonance (10.58 GeV)
- Focus on B, charm and τ physics



	KEKB/Belle	SuperKEKB/Belle II
operation	1999–2010	2018–
peak luminosity	$2.11 \times 10^{34}\,\mathrm{cm^{-2}s^{-1}}$	$8 \times 10^{35}\,\mathrm{cm^{-2}s^{-1}}$
integrated luminosity	$1023\,\mathrm{fb^{-1}}$ (772 million $B\overline{B}$ pairs)	$50\,\mathrm{ab^{-1}}$

(target values)

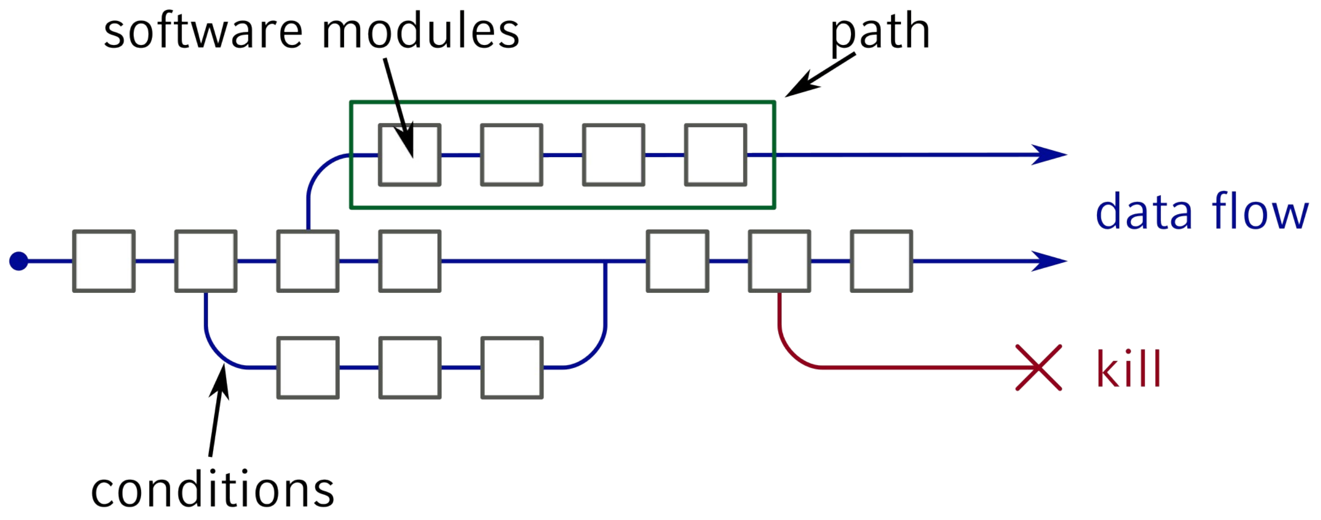
Typical events



- ~10 tracks/event
- ~100kB raw data
  → ~10kB mdst
- Relatively low track momenta
- Relatively high machine background

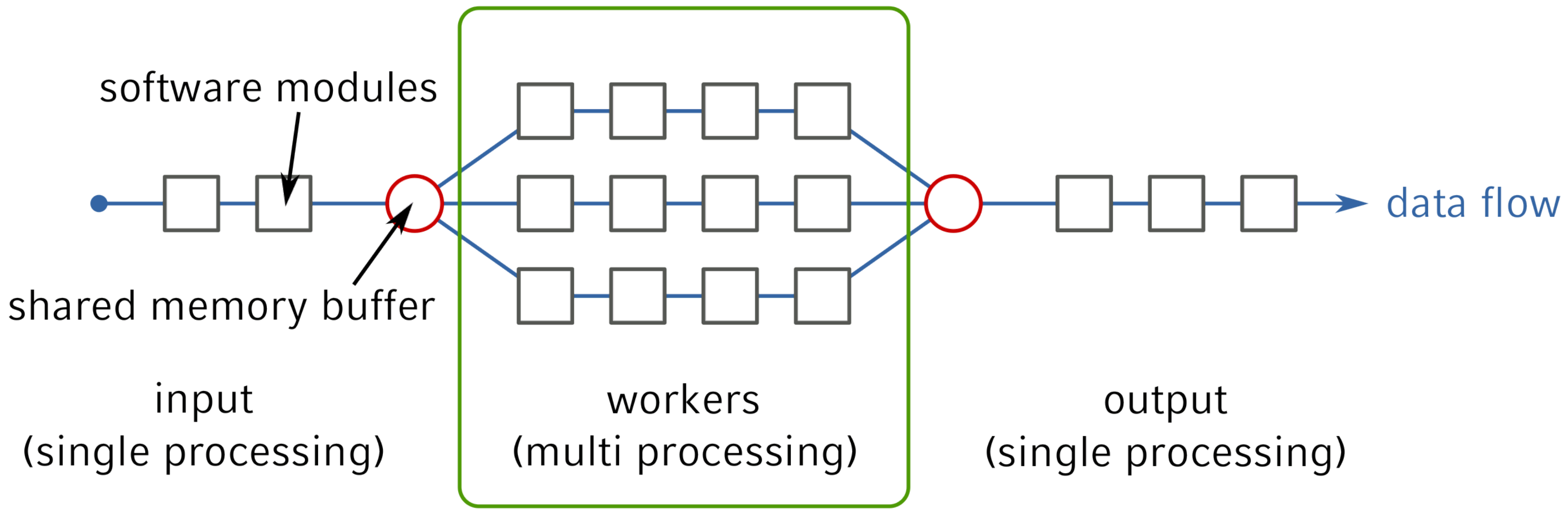
Belle 2 framework

- Written from scratch, using experience from Belle & others
- Many similarities with its contemporaries :
  - C++ computations, Python configuration & steering
  - ROOT-based I/O, Geant4-based simulation
  - Sequential chains of *modules* + control flow



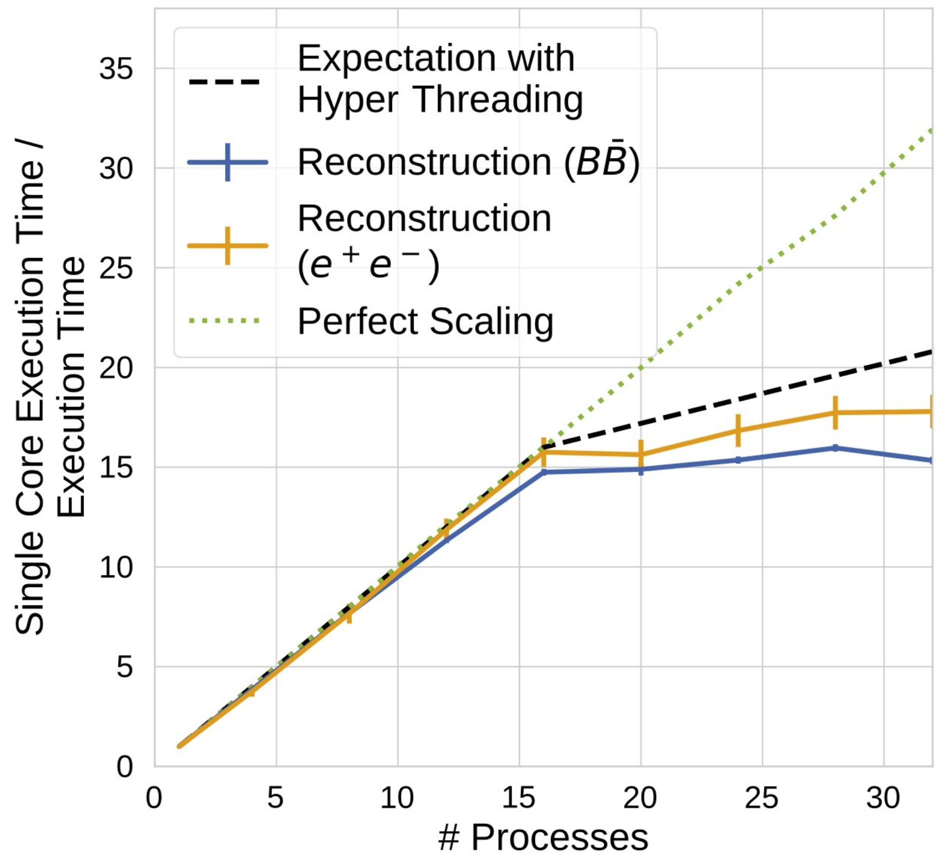
Parallelization strategy

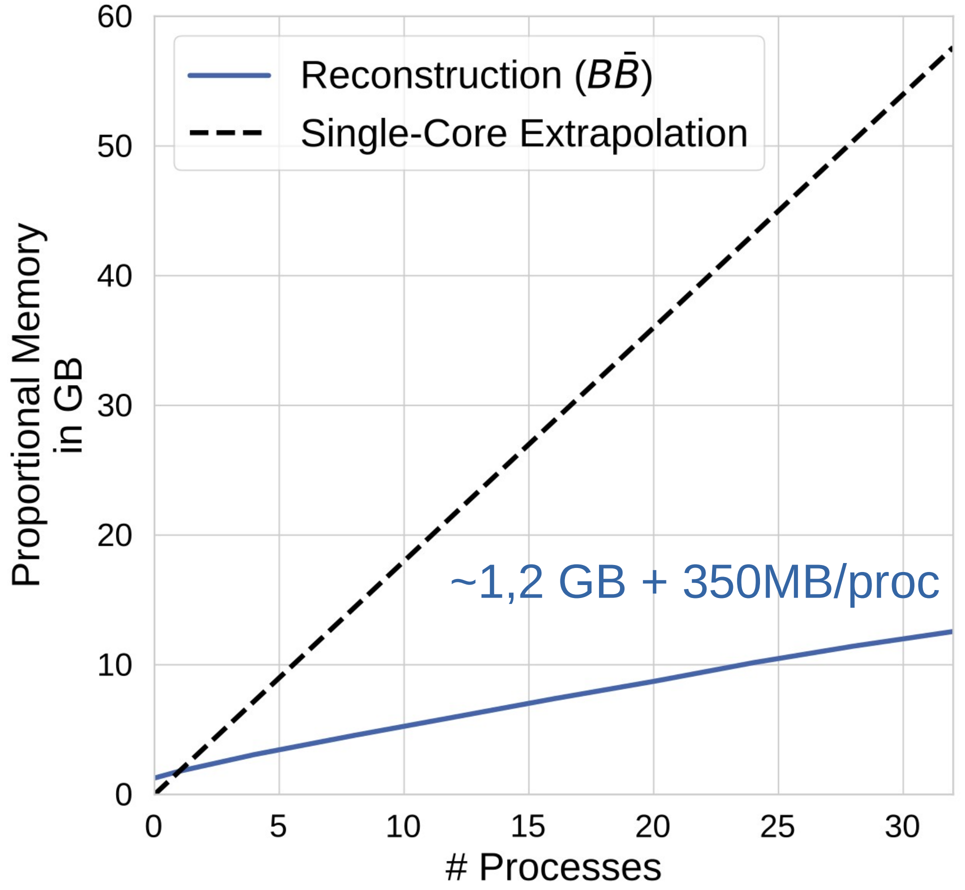
- Operating at event granularity
- (Seq. input → Parallel processing → Seq. output) pipeline
- Processes communicating via POSIX shared memory
- Memory sharing achieved through late process *fork*-ing



Parallel scaling

- Current solution scales well enough for the foreseeable future
  - A good thing: we couldn't afford a multi-threading migration





High-level analysis tools

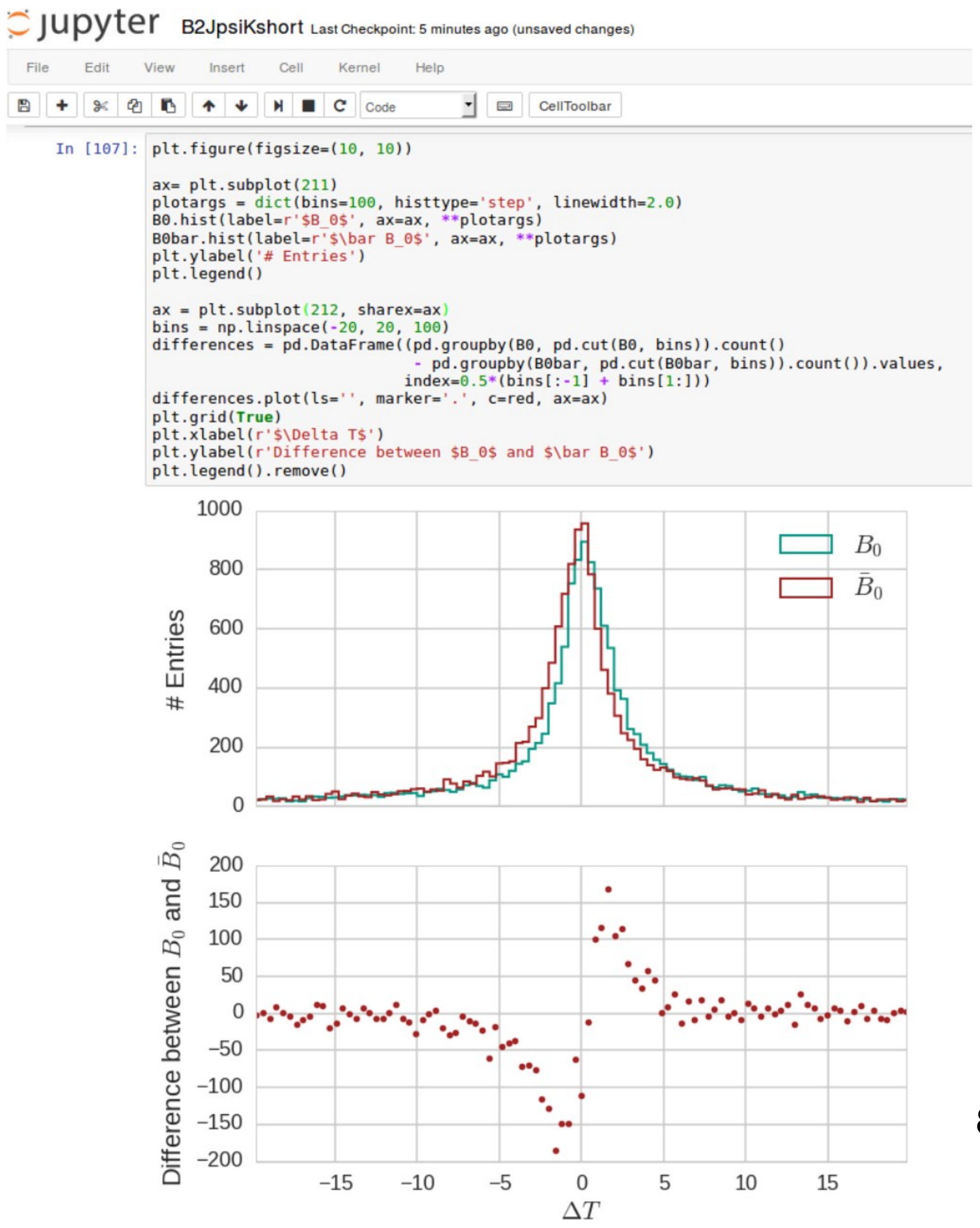
- DSL-based high-level analysis primitives
- Reduces the need for C++, aiming for “pure python”

Example: reconstruct $B^0 \to J/\psi(\to\mu^+\mu^-)\ K^0_S(\to\pi^+\pi^-)$

```
# create Ks -> pi+ pi- list from V0
# keep only candidates with 0.4 < M(pipi) < 0.6 GeV
fillParticleList('K_S0:pipi', '0.4 < M < 0.6')

# reconstruct J/psi -> mu+ mu- decay
# keep only candidates with 3.0 < M(mumu) < 3.2 GeV
reconstructDecay('J/psi:mumu -> mu+:loose mu-:loose', '3.0 < M < 3.2')

# reconstruct B0 -> J/psi Ks decay
# keep only candidates with 5.2 < M(J/PsiKs) < 5.4 GeV
reconstructDecay('B0:jspiks -> J/psi:mumu K_S0:pipi', '5.2 < M < 5.4')

# perform B0 kinematic vertex fit using only the mu+ mu-
# keep candidates only passing C.L. value of the fit > 0.0 (no cut)
vertexRave('B0:jspiks', 0.0, 'B0 -> [J/psi -> ^mu+ ^mu-] K_S0')

# build the rest of the event associated to the B0
buildRestOfEvent('B0:jspiks')

# perform MC matching (MC truth asociation). Always before TagV
matchMCTruth('B0:jspiks')

# calculate the Tag Vertex and Delta t (in ps)
# breco: type of MC association.
TagV('B0:jspiks', 'breco')
```

High-level analysis tools

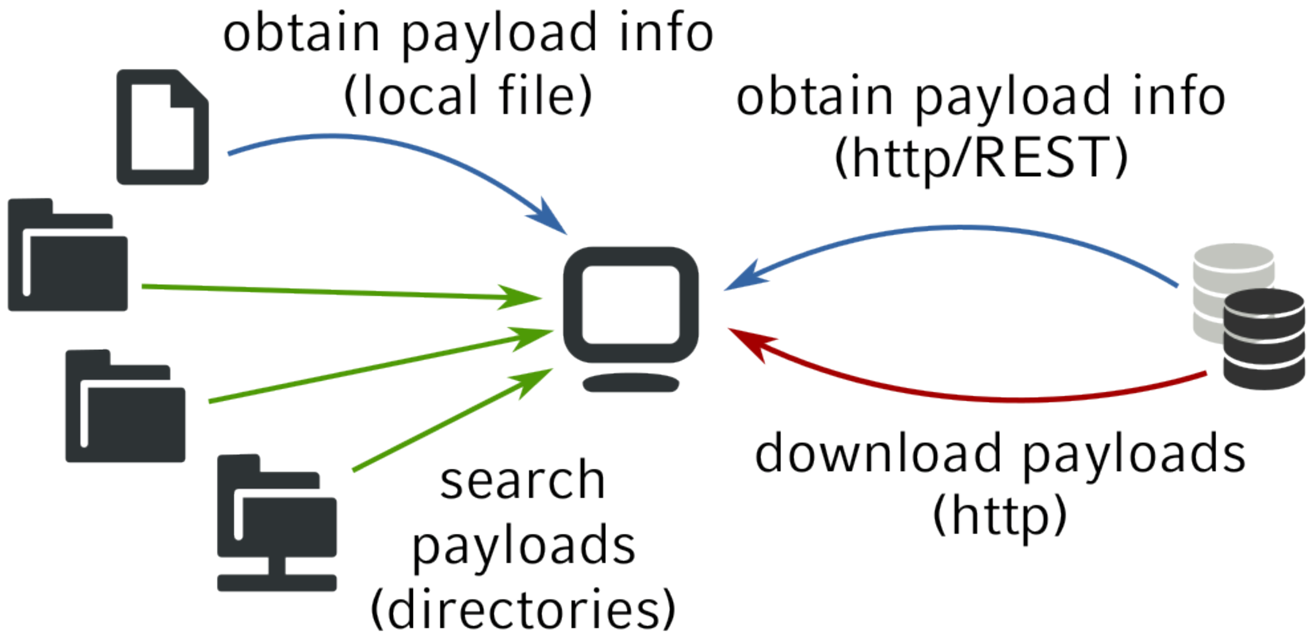
- DSL-based high-level analysis primitives
- Reduces the need for C++, aiming for “pure python”
- Jupyter integration:
  - Built-in to basf2 releases
  - Work done in subprocesses
  - Log and event data viewers
  - Progress bars…



```
plt.figure(figsize=(10, 10))

ax= plt.subplot(211)
plotargs = dict(bins=100, histtype='step', linewidth=2.0)
B0.hist(label=r'$B_0$', ax=ax, **plotargs)
B0bar.hist(label=r'$\bar B_0$', ax=ax, **plotargs)
plt.ylabel('# Entries')
plt.legend()

ax = plt.subplot(212, sharex=ax)
bins = np.linspace(-20, 20, 100)
differences = pd.DataFrame((pd.groupby(B0, pd.cut(B0, bins)).count()
                           - pd.groupby(B0bar, pd.cut(B0bar, bins)).count()).values,
                          index=0.5*(bins[:-1] + bins[1:]))
differences.plot(ls='', marker='.', c=red, ax=ax)
plt.grid(True)
plt.xlabel(r'$\Delta T$')
plt.ylabel(r'Difference between $B_0$ and $\bar B_0$')
plt.legend().remove()
```

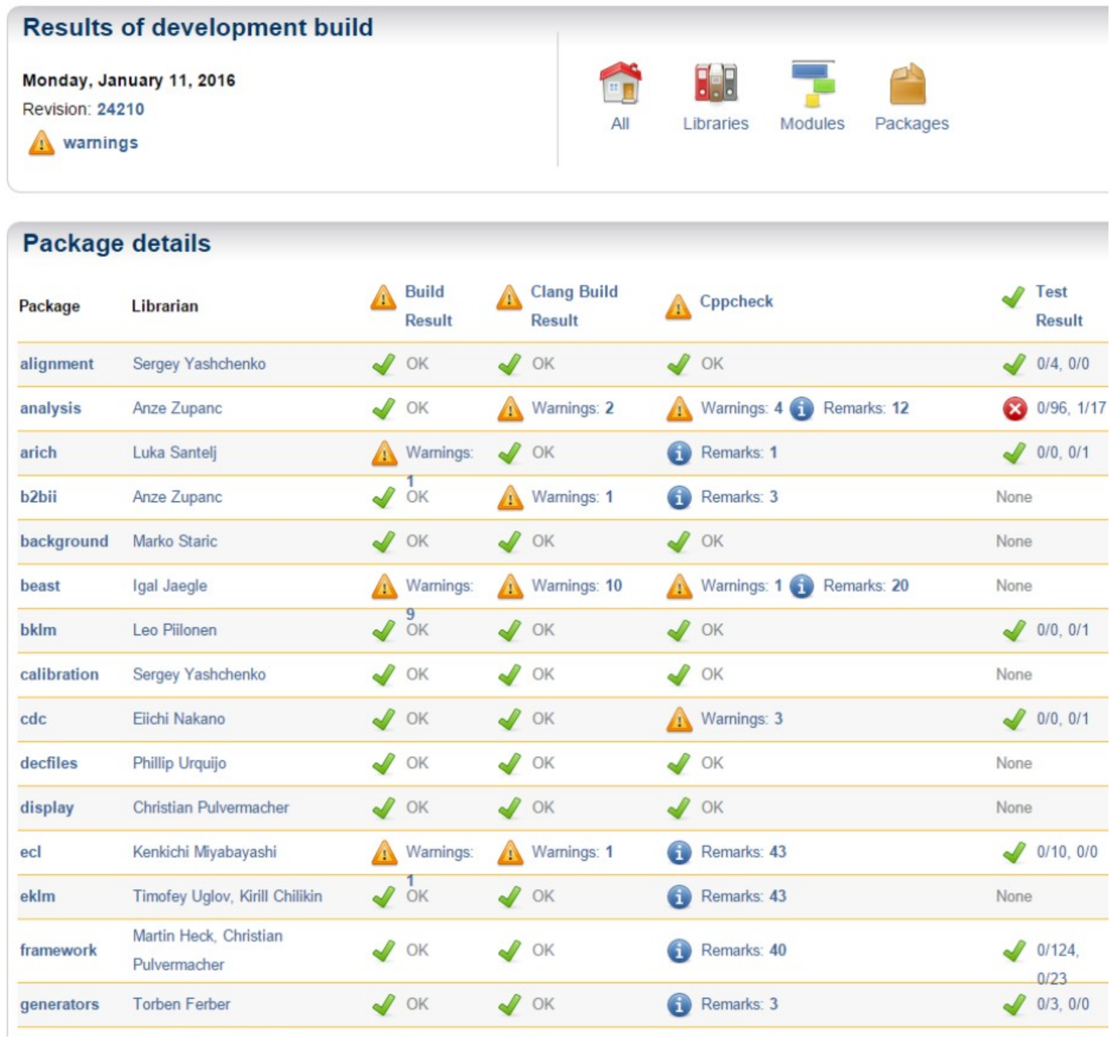
Condition database

- Metadata is queried from a REST server
- Payloads are usually ROOT files, fetched from CVMFS
- Full local caching → seamless offline work & benchmarking



Quality infrastructure

- Framework benefits from overall focus on SW quality :
  - Continuous integration
  - GCC + clang + icc builds
  - Aiming for zero warnings
  - cppcheck, clang static analyzer, memcheck
  - Software quality shifts
  - Nightly validation



Results of development build

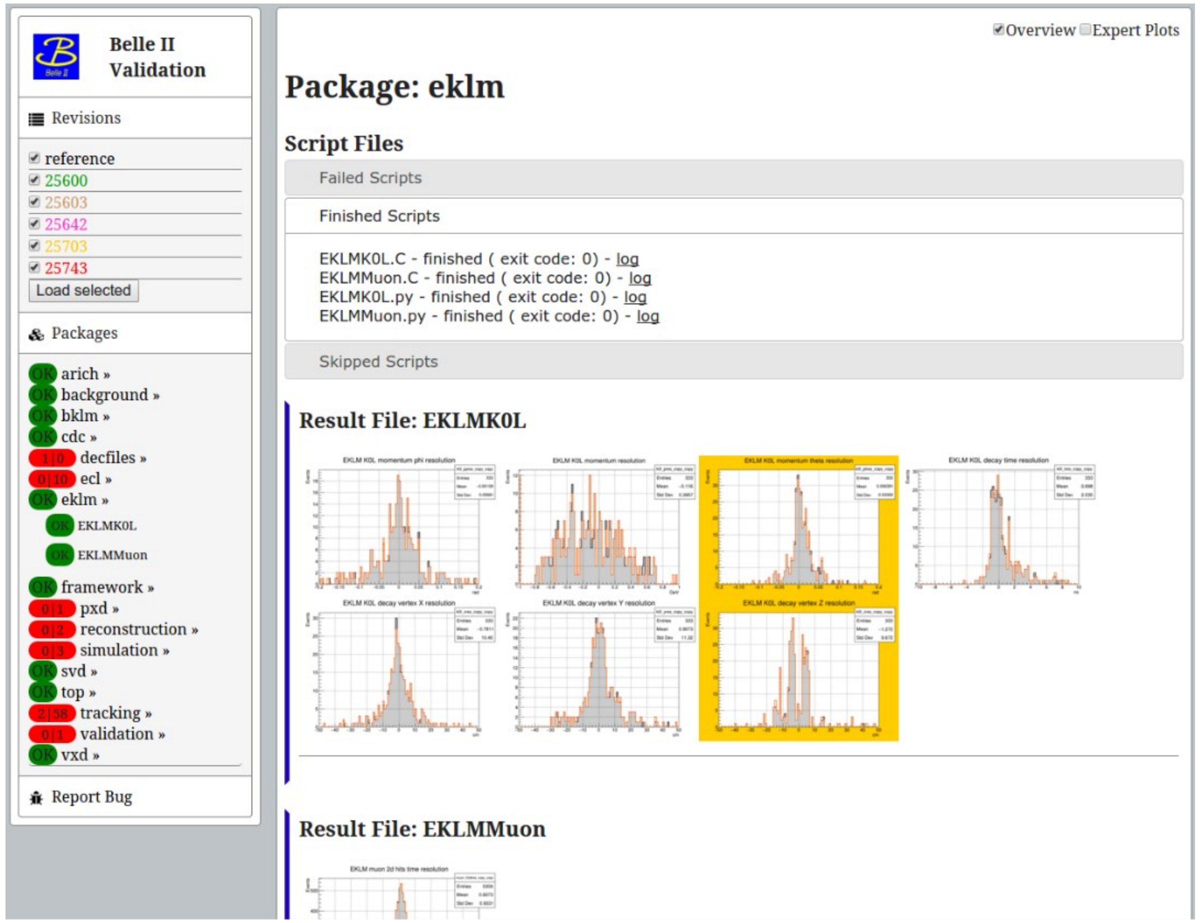
Monday, January 11, 2016
Revision: 24210
⚠ warnings

Package details

Package	Librarian	⚠ Build Result	⚠ Clang Build Result	⚠ Cppcheck	✓ Test Result
alignment	Sergey Yashchenko	✓ OK	✓ OK	✓ OK	✓ 0/4, 0/0
analysis	Anze Zupanc	✓ OK	⚠ Warnings: 2	⚠ Warnings: 4 ℹ Remarks: 12	✗ 0/96, 1/17
arich	Luka Santelj	⚠ Warnings: 1	✓ OK	ℹ Remarks: 1	✓ 0/0, 0/1
b2bii	Anze Zupanc	✓ OK	⚠ Warnings: 1	ℹ Remarks: 3	None
background	Marko Staric	✓ OK	✓ OK	✓ OK	None
beast	Igal Jaegle	⚠ Warnings: 9	⚠ Warnings: 10	⚠ Warnings: 1 ℹ Remarks: 20	None
bklm	Leo Piilonen	✓ OK	✓ OK	✓ OK	✓ 0/0, 0/1
calibration	Sergey Yashchenko	✓ OK	✓ OK	✓ OK	None
cdc	Eiichi Nakano	✓ OK	✓ OK	⚠ Warnings: 3	✓ 0/0, 0/1
decfiles	Phillip Urquijo	✓ OK	✓ OK	✓ OK	None
display	Christian Pulvermacher	✓ OK	✓ OK	✓ OK	None
ecl	Kenkichi Miyabayashi	⚠ Warnings: 1	⚠ Warnings: 1	ℹ Remarks: 43	✓ 0/10, 0/0
eklm	Timofey Uglov, Kirill Chilikin	✓ OK	✓ OK	ℹ Remarks: 43	None
framework	Martin Heck, Christian Pulvermacher	✓ OK	✓ OK	ℹ Remarks: 40	✓ 0/124, 0/23
generators	Torben Ferber	✓ OK	✓ OK	ℹ Remarks: 3	✓ 0/3, 0/0

Validation infrastructure

- Physics plots every night
- Auto. dependency handling
- Auto. checks vs reference
- Nice web visualization
- Easy local runs
- Easy bisect tooling



Package: eklm

Script Files

Failed Scripts

Finished Scripts

EKLMK0L.C - finished (exit code: 0) - log
EKLMMuon.C - finished (exit code: 0) - log
EKLMK0L.py - finished (exit code: 0) - log
EKLMMuon.py - finished (exit code: 0) - log

Skipped Scripts

Result File: EKLMK0L

Result File: EKLMMuon

Platform support

- We provide pre-built binaries for…
  - RHEL/SLC 6, RHEL/CentOS 7 + 8 coming soon
  - Ubuntu 16.04, 18.04 + 20.04 coming soon
- For many other distributions, we provide…
  - Automatic externals build deps installation
  - Automatic (Make-based) externals compilation
- Belle 2 specific software is built via SCons

Conclusion

- The Belle 2 framework is a “wget” of HEP software frameworks
  - Solid implementation of classical approaches
  - Works well enough for people to forget it’s there
- Mostly in maintenance mode these days
  - Main ongoing R&D: replace POSIX shm IPC with ZeroMQ