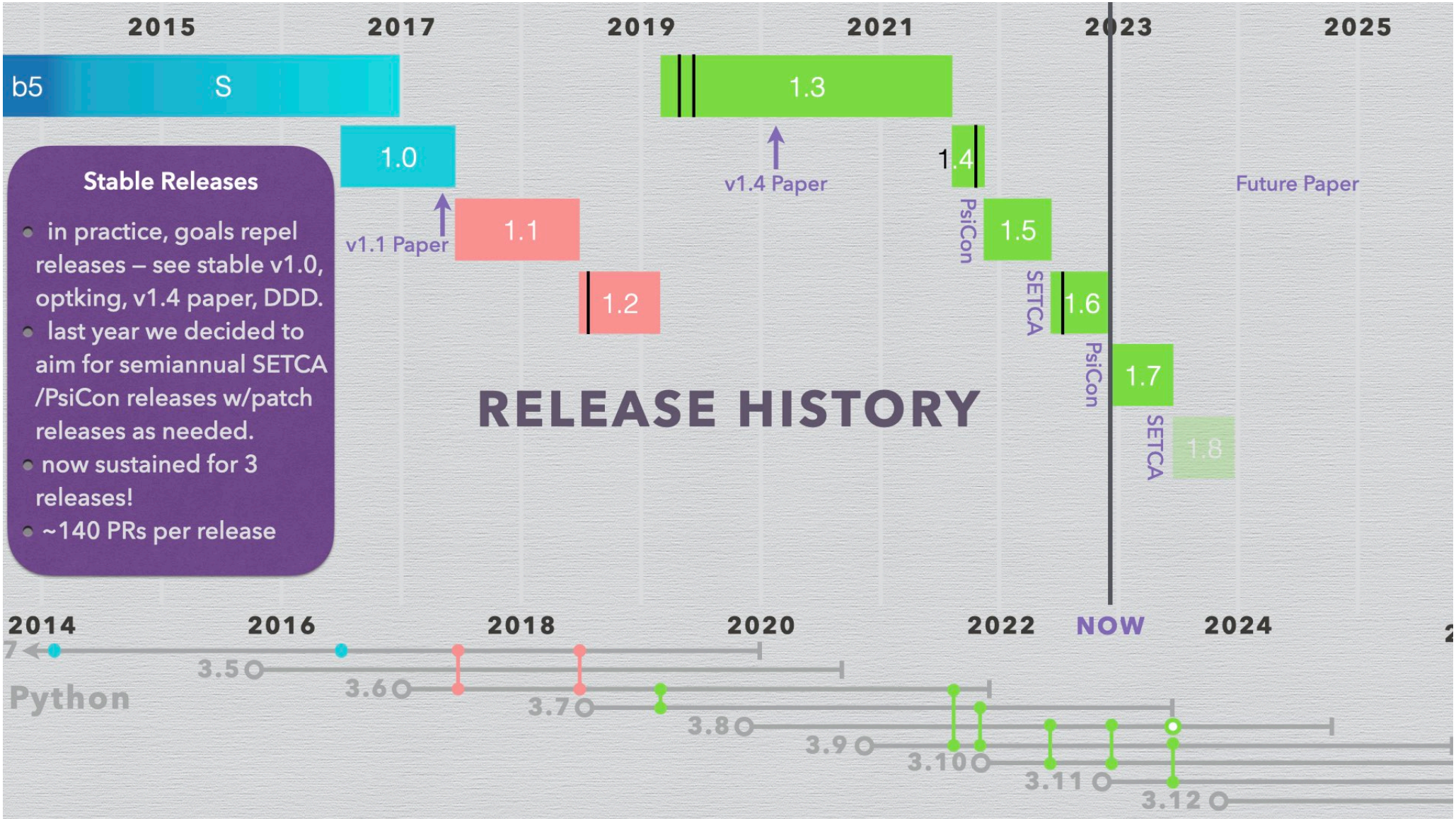
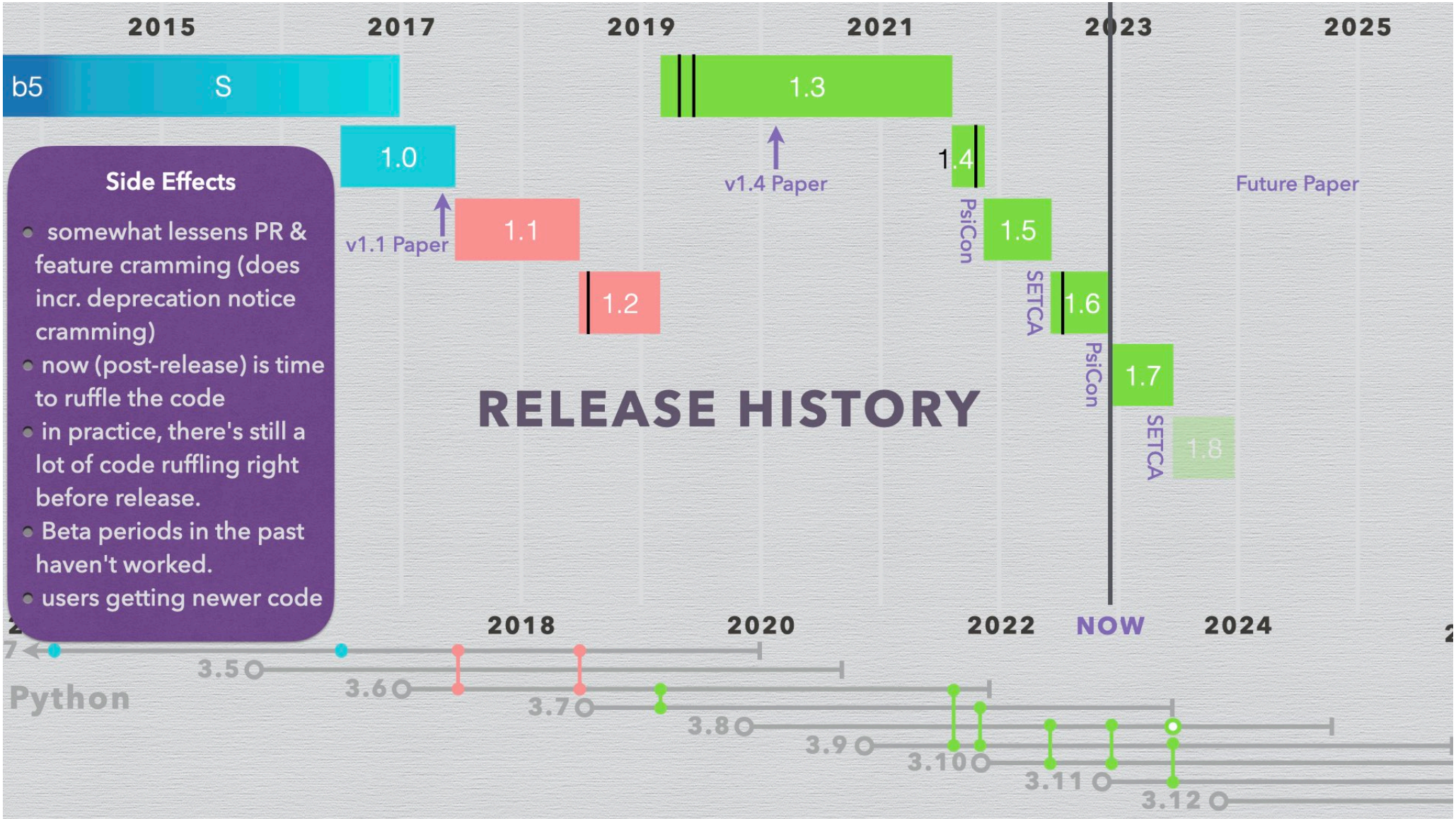
2022 PSI4 ORDNANCE SURVEY

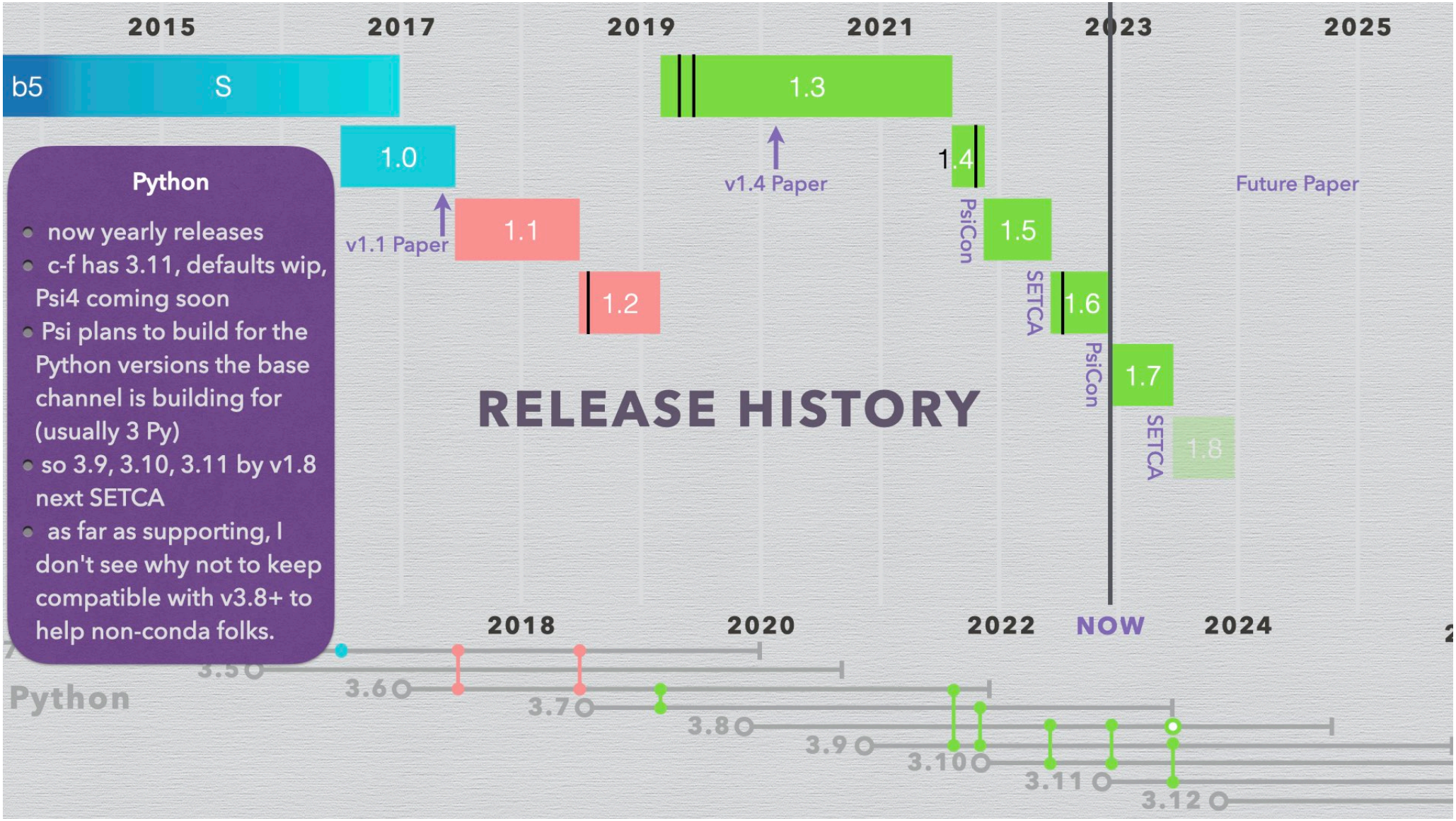
LORI A. BURNS

PSICON 2022, BLACKSBURG, VA

9 DECEMBER 2021


2015
2017
2019
2021
2023
2025
b5
S
1.0
1.1
1.2
1.3
1.4
1.5
1.6
1.7
1.8
v1.1 Paper
v1.4 Paper
PsiCon
SETCA
PsiCon
SETCA
Future Paper
Stable Releases
in practice, goals repel releases – see stable v1.0, optking, v1.4 paper, DDD.
last year we decided to aim for semiannual SETCA /PsiCon releases w/patch releases as needed.
now sustained for 3 releases!
~140 PRs per release
RELEASE HISTORY
2014
2016
2018
2020
2022
NOW
2024
Python
2
3.5
3.6
3.7
3.8
3.9
3.10
3.11
3.12

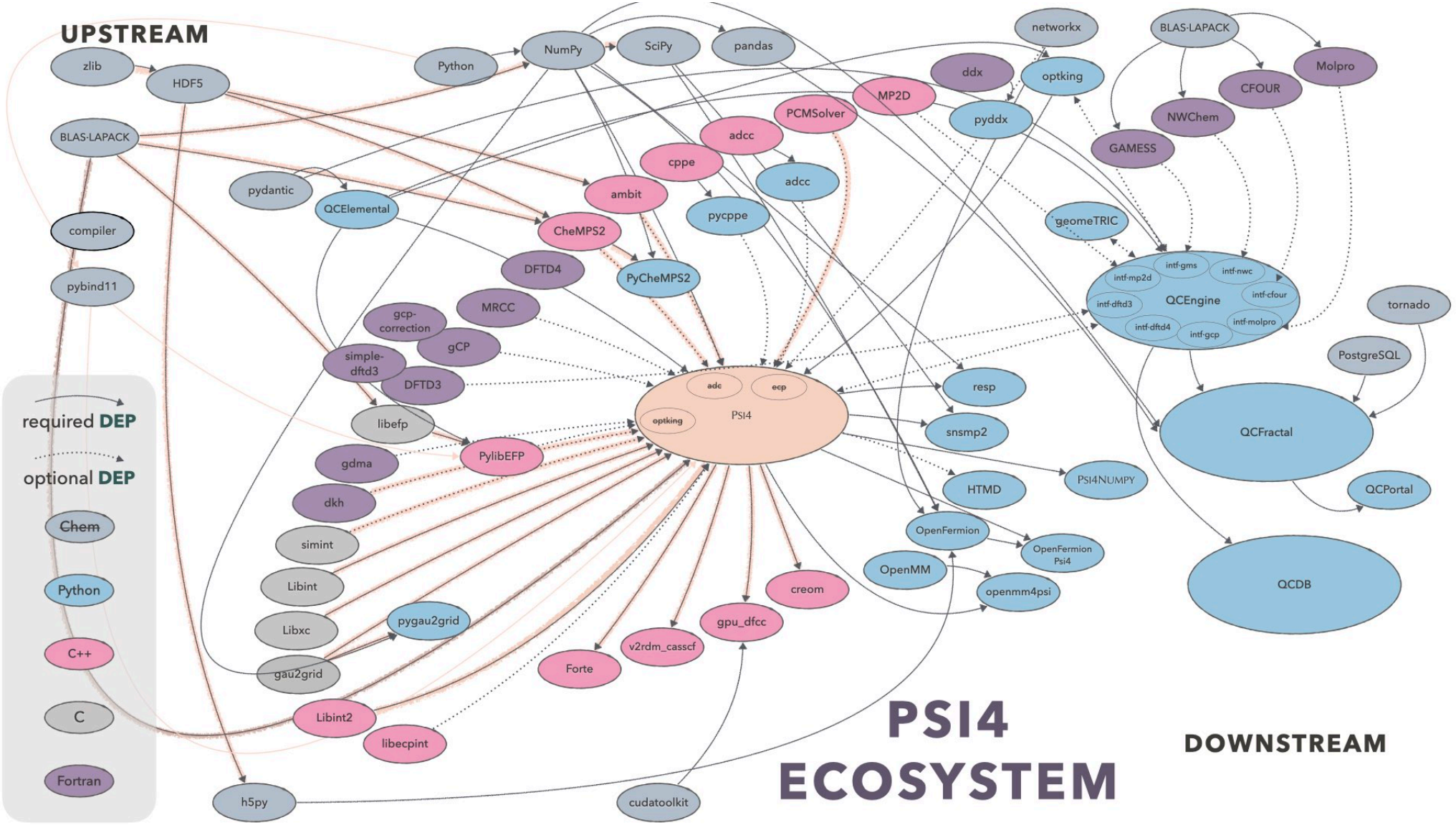

2015
2017
2019
2021
2023
2025
b5
S
1.0
1.1
1.2
1.3
1.4
1.5
1.6
1.7
1.8
v1.1 Paper
v1.4 Paper
Future Paper
PsiCon
SETCA
PsiCon
SETCA
RELEASE HISTORY
Side Effects
somewhat lessens PR & feature cramming (does incr. deprecation notice cramming)
now (post-release) is time to ruffle the code
in practice, there's still a lot of code ruffling right before release.
Beta periods in the past haven't worked.
users getting newer code
2018
2020
2022
NOW
2024
Python
3.5
3.6
3.7
3.8
3.9
3.10
3.11
3.12

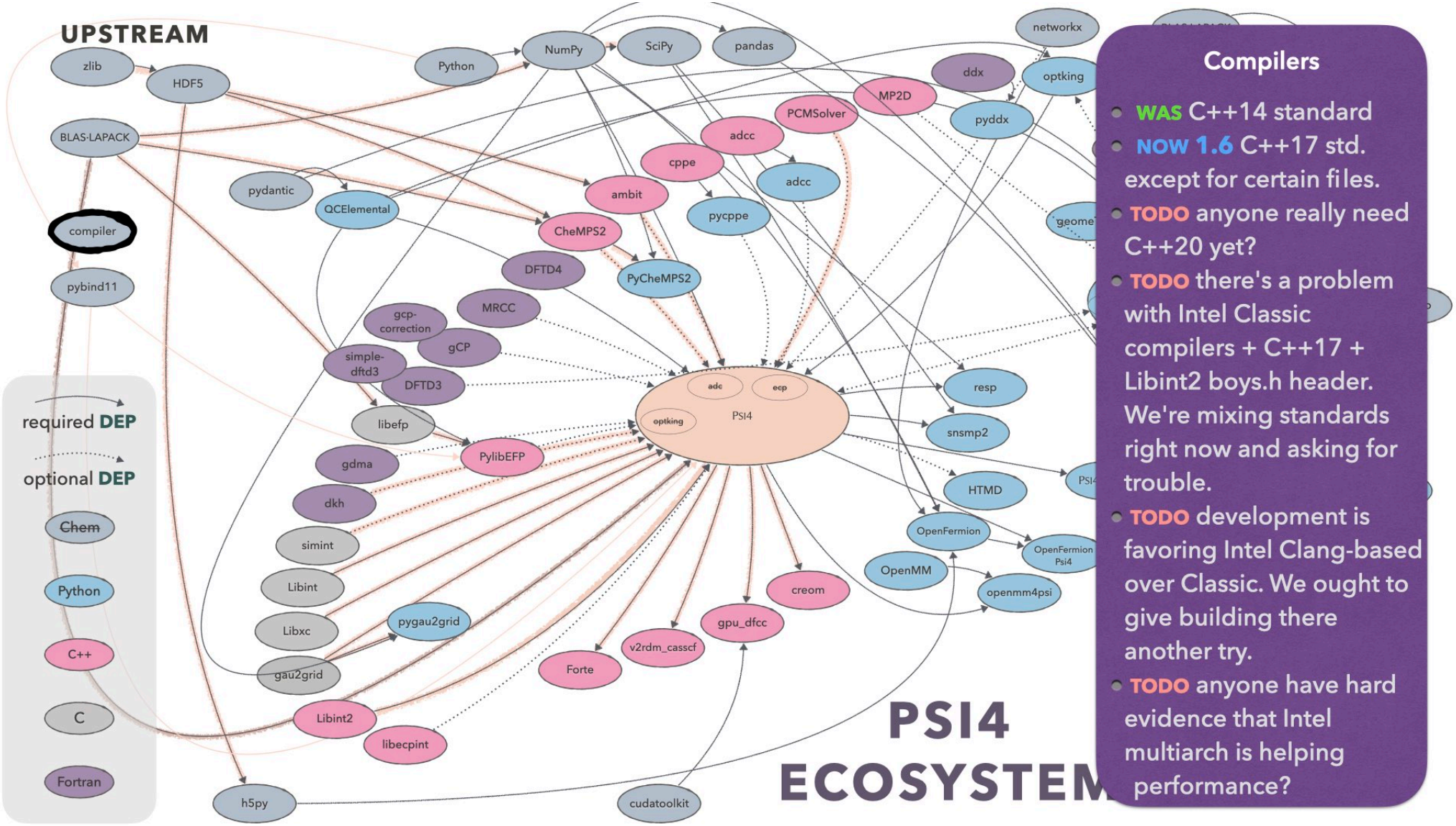

2015
2017
2019
2021
2023
2025
b5
S
1.0
1.1
1.2
1.3
1.4
1.5
1.6
1.7
1.8
v1.1 Paper
v1.4 Paper
Future Paper
PsiCon
SETCA
PsiCon
SETCA
RELEASE HISTORY
Python
now yearly releases
c-f has 3.11, defaults wip, Psi4 coming soon
Psi plans to build for the Python versions the base channel is building for (usually 3 Py)
so 3.9, 3.10, 3.11 by v1.8 next SETCA
as far as supporting, I don't see why not to keep compatible with v3.8+ to help non-conda folks.
2018
2020
2022
NOW
2024
Python
3.5
3.6
3.7
3.8
3.9
3.10
3.11
3.12

Psi4 1.7.0

Now Available

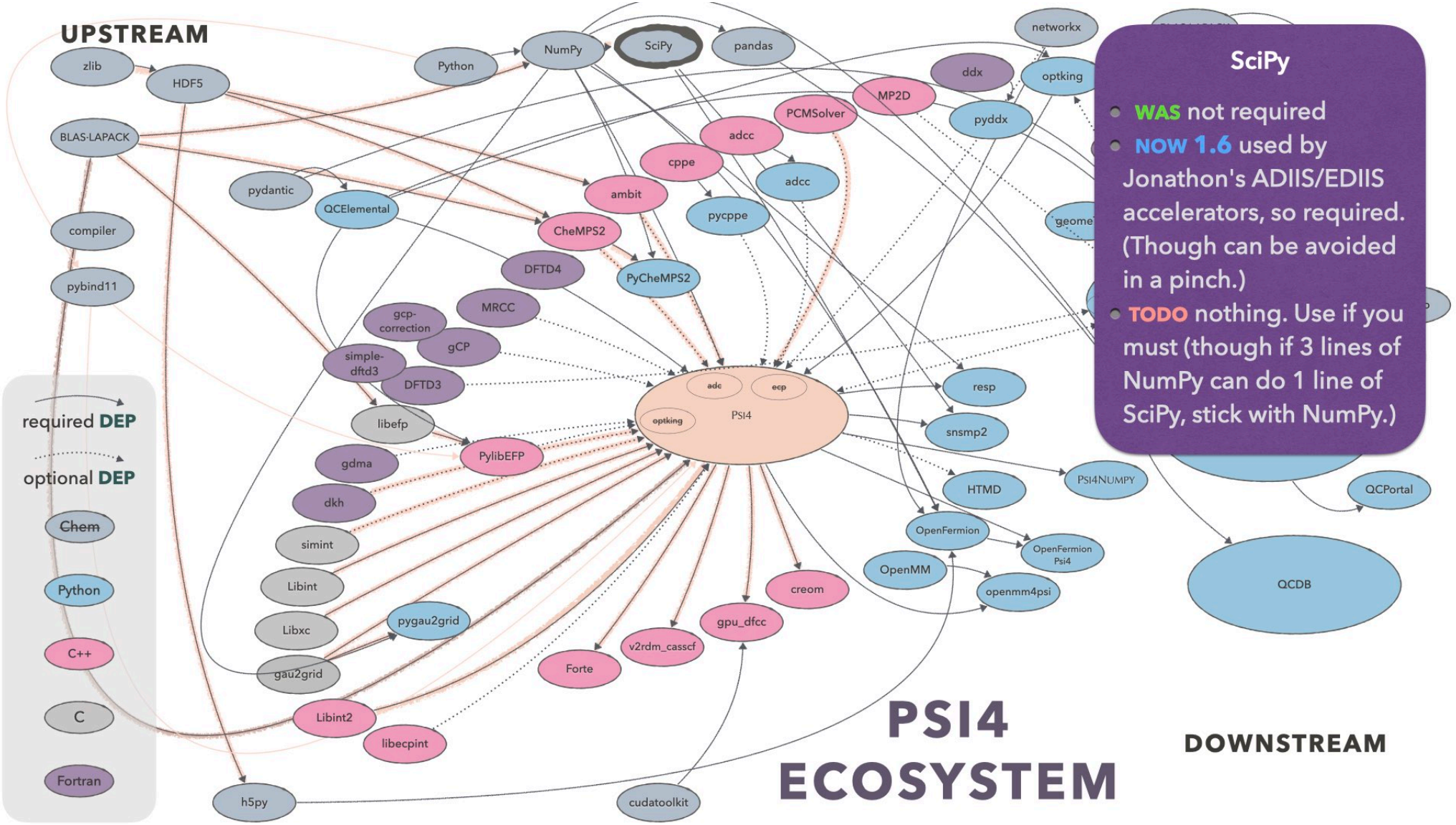
6 December 2022

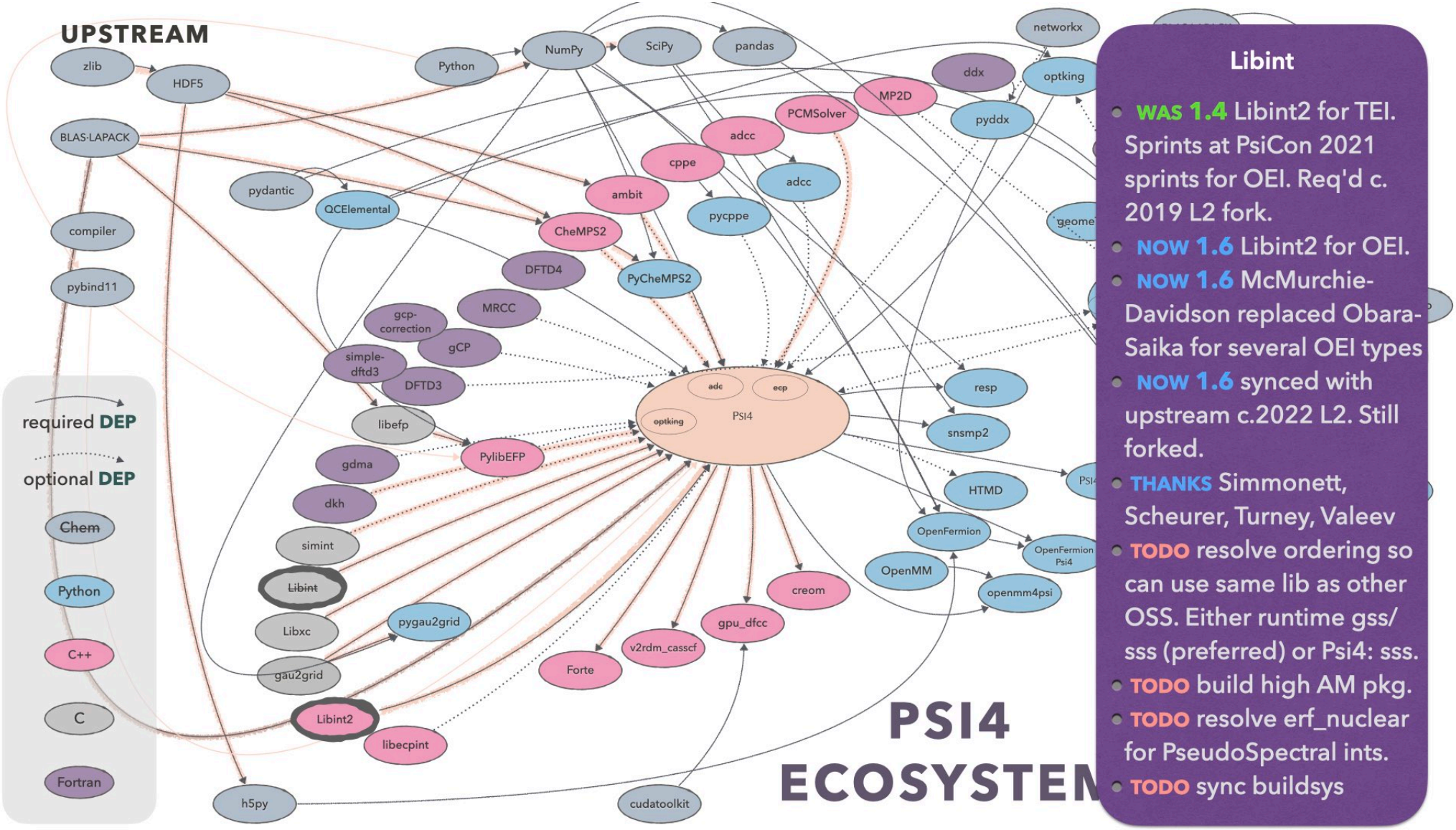

UPSTREAM
zlib
HDF5
BLAS-LAPACK
compiler
pybind11
Python
NumPy
SciPy
pandas
pydantic
QCElemental
networkx
optking
ddx
MP2D
pyddx
PCMSolver
adcc
cppe
adcc
ambit
pycppe
CheMPS2
DFTD4
PyCheMPS2
MRCC
gcp-correction
gCP
simple-dftd3
DFTD3
libefp
PylibEFP
gdma
dkh
simint
Libint
Libxc
gau2grid
pygau2grid
Libint2
libecpint
h5py
Forte
v2rdm_casscf
gpu_dfcc
creom
cudatoolkit
adc
ecp
optking
PSI4
resp
snsmp2
HTMD
OpenFermion
OpenMM
openmm4psi
OpenFermion Psi4
PSI4NUMPY
BLAS-LAPACK
Molpro
CFOUR
NWChem
GAMESS
geomeTRIC
intf-gms
intf-nwc
intf-mp2d
intf-cfour
intf-dftd3
QCEngine
intf-dftd4
intf-molpro
intf-gcp
tornado
PostgreSQL
QCFractal
QCPortal
QCDB
required DEP
optional DEP
Chem
Python
C++
C
Fortran
PSI4
ECOSYSTEM
DOWNSTREAM



Compilers

- **WAS** C++14 standard
- **NOW 1.6** C++17 std. except for certain files.
- **TODO** anyone really need C++20 yet?
- **TODO** there's a problem with Intel Classic compilers + C++17 + Libint2 boys.h header. We're mixing standards right now and asking for trouble.
- **TODO** development is favoring Intel Clang-based over Classic. We ought to give building there another try.
- **TODO** anyone have hard evidence that Intel multiarch is helping performance?


UPSTREAM
zlib
HDF5
BLAS-LAPACK
compiler
pybind11
Python
NumPy
SciPy
pandas
pydantic
QCElemental
networkx
optking
ddx
MP2D
PCMSolver
pyddx
adcc
cppe
adcc
ambit
pycppe
CheMPS2
PyCheMPS2
DFTD4
MRCC
gcp-correction
gCP
simple-dftd3
DFTD3
libefp
PylibEFP
gdma
dkh
simint
Libint
Libxc
gau2grid
pygau2grid
Libint2
libecpint
h5py
cudatoolkit
Forte
v2rdm_casscf
gpu_dfcc
creom
adc
ecp
optking
PSI4
resp
snsmp2
HTMD
PSI4NUMPY
QCPortal
OpenFermion
OpenFermion Psi4
OpenMM
openmm4psi
QCDB
required DEP
optional DEP
Chem
Python
C++
C
Fortran
SciPy
WAS not required
NOW 1.6 used by Jonathon's ADIIS/EDIIS accelerators, so required. (Though can be avoided in a pinch.)
TODO nothing. Use if you must (though if 3 lines of NumPy can do 1 line of SciPy, stick with NumPy.)
PSI4 ECOSYSTEM
DOWNSTREAM



Libint

- **WAS 1.4** Libint2 for TEI. Sprints at PsiCon 2021 sprints for OEI. Req'd c. 2019 L2 fork.
- **NOW 1.6** Libint2 for OEI.
- **NOW 1.6** McMurchie-Davidson replaced Obara-Saika for several OEI types
- **NOW 1.6** synced with upstream c.2022 L2. Still forked.
- **THANKS** Simmonett, Scheurer, Turney, Valeev
- **TODO** resolve ordering so can use same lib as other OSS. Either runtime gss/sss (preferred) or Psi4: sss.
- **TODO** build high AM pkg.
- **TODO** resolve erf_nuclear for PseudoSpectral ints.
- **TODO** sync buildsys

LIBINT2

progress in 2022

Steps toward Psi4 using a upstream Libint2 release in common with other open-source QC programs:

- step 2 — 20 Nov 2020 — Libint2 and shell screening #1721
- step 3 — 11 Mar 2022 — Libint2 one electron integrals #2388
- post 3 — 16 Mar 2022 — fix L2 tarball for current post-l2-osi master #2474, pin L2 for l2-oei #2482, ack, missed a L2 #2484
- step C — 21 March 2022 — Three-center overlap integrals using Libint2 #2489
- step C — 21 March 2022 — McMurchie-Davidson integrals for angular momentum #2483
- step B — 23 March 2022 — Update CMake fork of Libint2, c. 2019 -> c. 2022 #2413
- post B — 23 March 2022 — fix up tests and bump win L2 #2494
- step C — 25 March 2022 — Interfaced Yukawa Libint2 ERI Kernel #2386
- step C — 25 March 2022 — Arbitrary-order multipole integrals (and gradients) with McMurchie-Davidson #24
- step C — 30 March 2022 — Multipole Potential Integrals with McMurchie-Davidson #2504
- step C — 31 March 2022 — Goodbye, `osrecur`! 👋 #2517
- step C — 31 March 2022 — Implement McMurchie Davidson Integrals #2414
- step C — Remove libint1 and erd integral engines. #2503
- step C — Bring in F12 integrals from libint2 #2502
- step A — change solid harmonic ordering from gaussian to standard #2537 —
- step C — Refactor `PseudospectralInts` using Libint2 #2473
- step B — WIP — new cmake harness, round 3 evaleev/libint#233
- step B — WIP — new cmake harness evaleev/libint#205
- coordinate with packagers and other QC programs for a AM and integrals types l2 config options set that is adequate all

Algorithms used for One Electron Integrals

Integral	Class	Implementation	Comment
Three-Center Overlap	`ThreeCenterOverlapInt`	Libint2	using `libint2::Operator::delta` for 4-center integrals
Angular Momentum	`AngularMomentumInt`	M-D	
Dipole	`DipoleInt`	Libint2	no derivatives supported
Electric Field	`ElectricFieldInt`	Libint2	using first derivative of `libint2::Operator::nuclear`
Coulomb Potential	`ElectrostaticInt`	Libint2	evaluated for a single origin and unity charge
Kinetic	`KineticInt`	Libint2	
Multipole Potential	`MultipolePotentialInt`	M-D	arbitrary order derivative of 1/R supported
Multipole Moments	`MultipoleInt`	M-D	arbitrary order multipoles supported, including nuclear gradients
Nabla Operator	`NablaInt`	Libint2	using first derivative of `libint2::Operator::overlap`
Overlap	`OverlapInt`	Libint2	
Nuclear Coulomb Potential	`PotentialInt`	Libint2	assumes nuclear centers/charges as the potential
PCM Potential	`PCMPotentialInt`	Libint2	parallelized over charge points
Quadrupole	`QuadrupoleInt`	Libint2	
Traceless Quadrupole	`TracelessQuadrupoleInt`	Libint2	
Relativistic Potential	`RelPotentialInt`	Libint2	

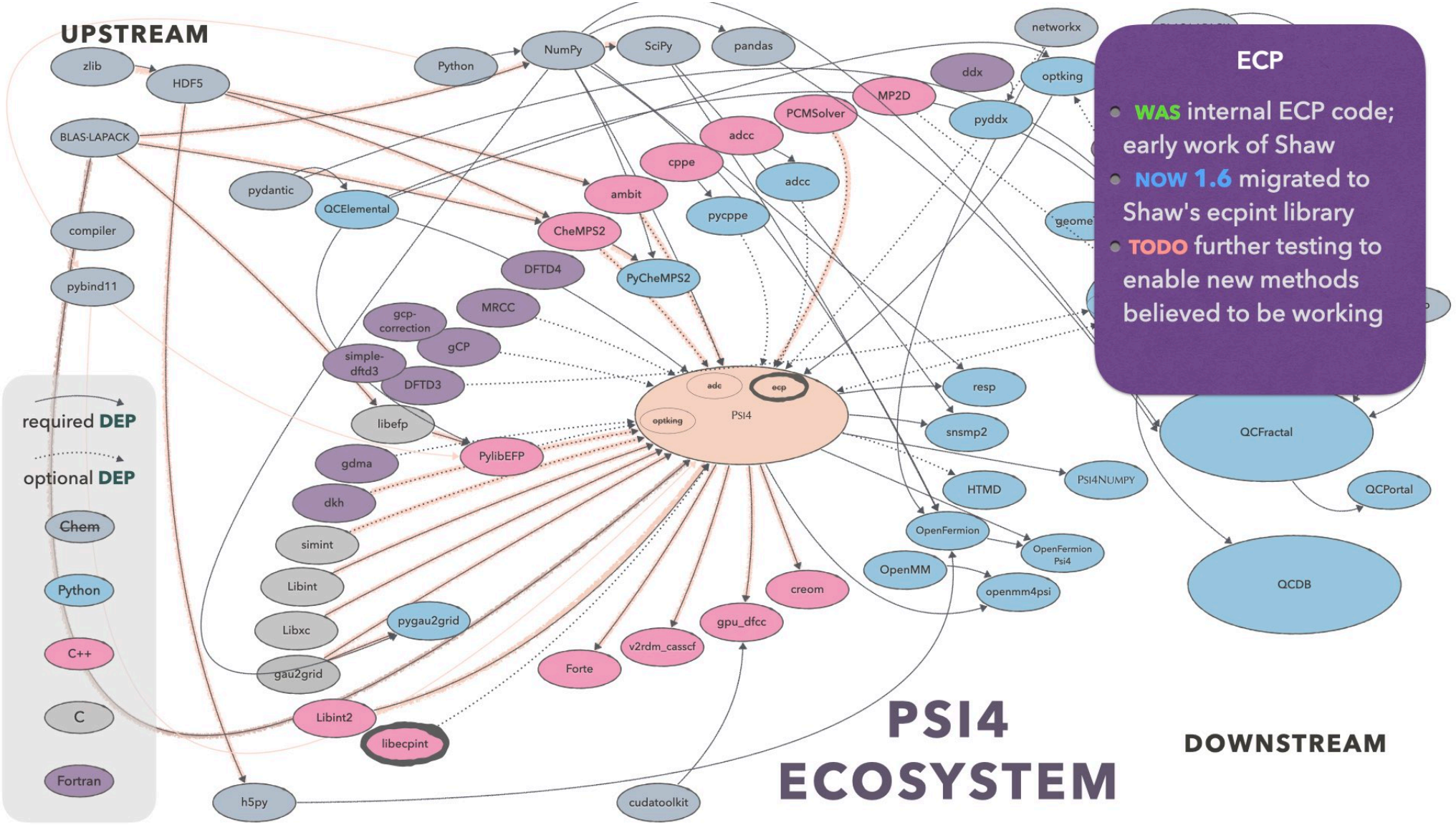
Open **Libint2 2022 Strategy** #2442

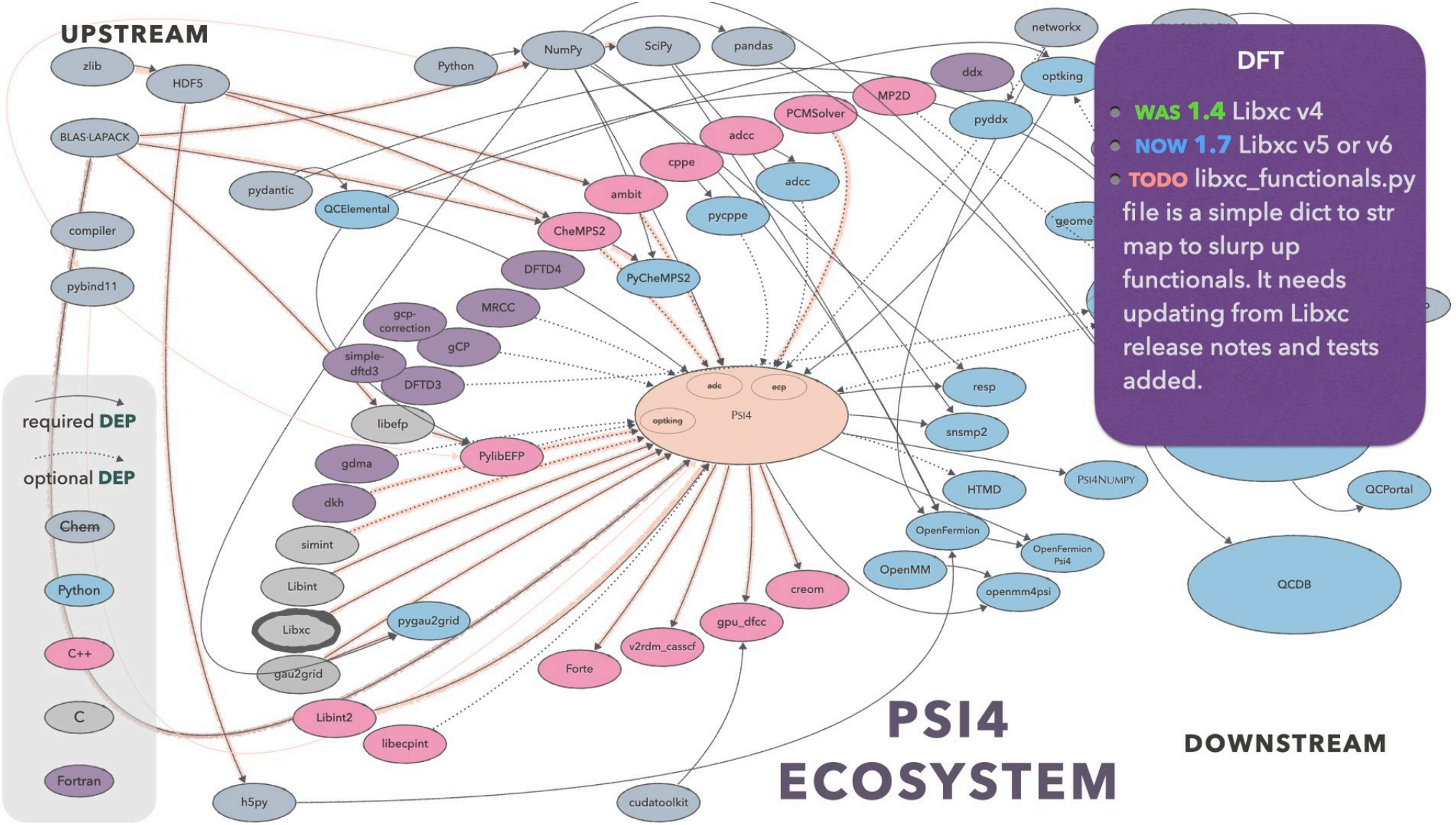
loriab opened this issue on Feb 14 · 2 comments

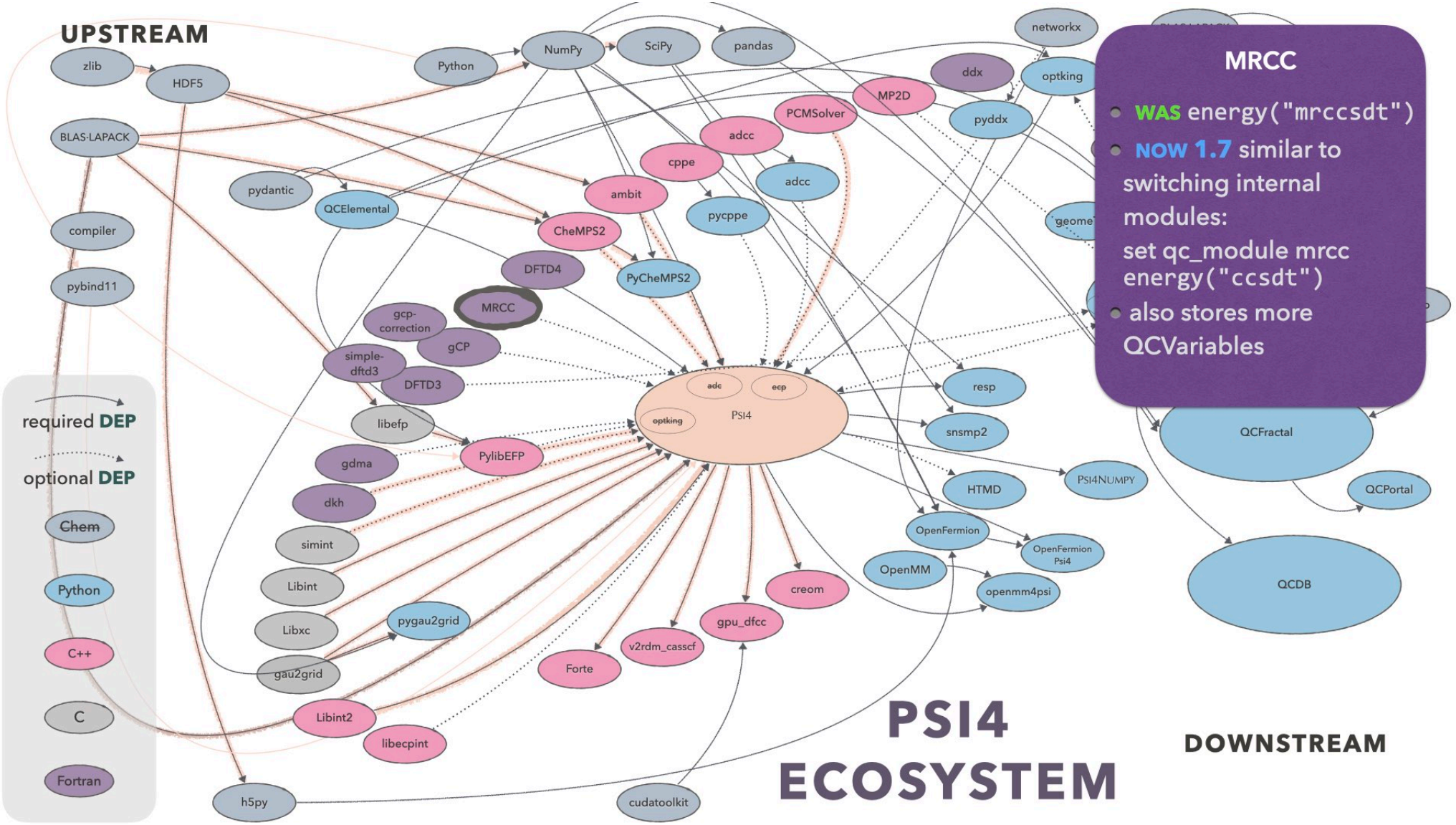
step	status	libint ver & branch	Psi4	tarball[1]
1[3]	longstanding L1	L1 evaleev:5c89451	v1.3	—
2[4]	TEI L2	loriab:l2cmake evaleev/libint#148	20Nov20, after #1721, v1.4, 1.5	L: `7-7-4-7-7-5_1`, MW: `5-4-3-6-5-4_1`
3[5]	OEI L2	ditto step 2	11Mar22, after #2388	L: `5-4-3-6-5-4_mm25f12ob2`, MW: `5-4-3-6-5-4_mm4ob2`
B[6]	upstream L2 cmake	loriab:new-cmake-harness-lab-rb1 evaleev/libint#233	23Mar22, after #2413, v1.6	`5-4-3-6-5-4_mm4f12ob2.tgz`
C[2]	McMurchie Davidson	any	31Mar22, after #2414, v1.6	
A[7]	standardize ordering	ditto step B	#2537	ditto step B

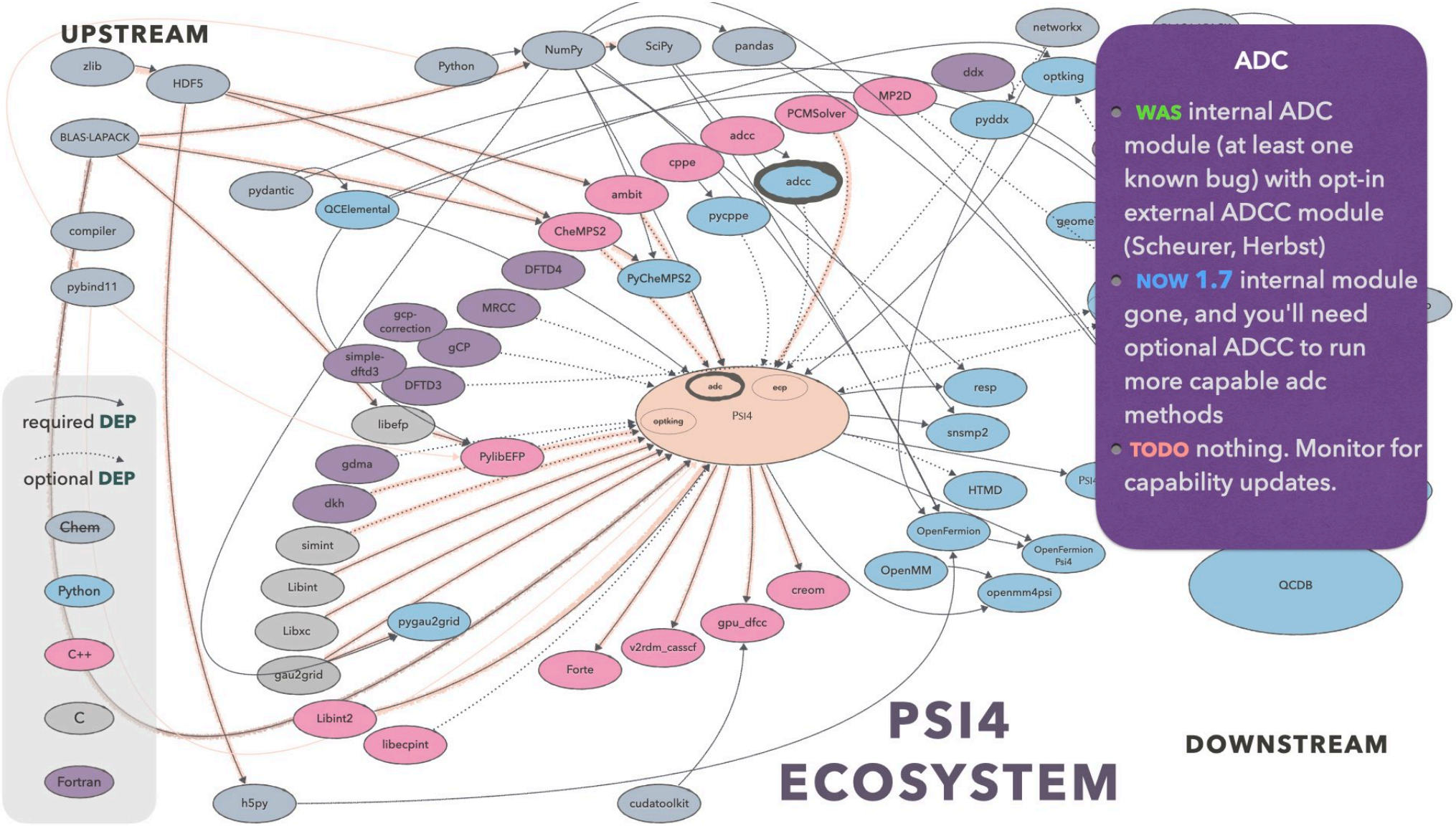
Libint

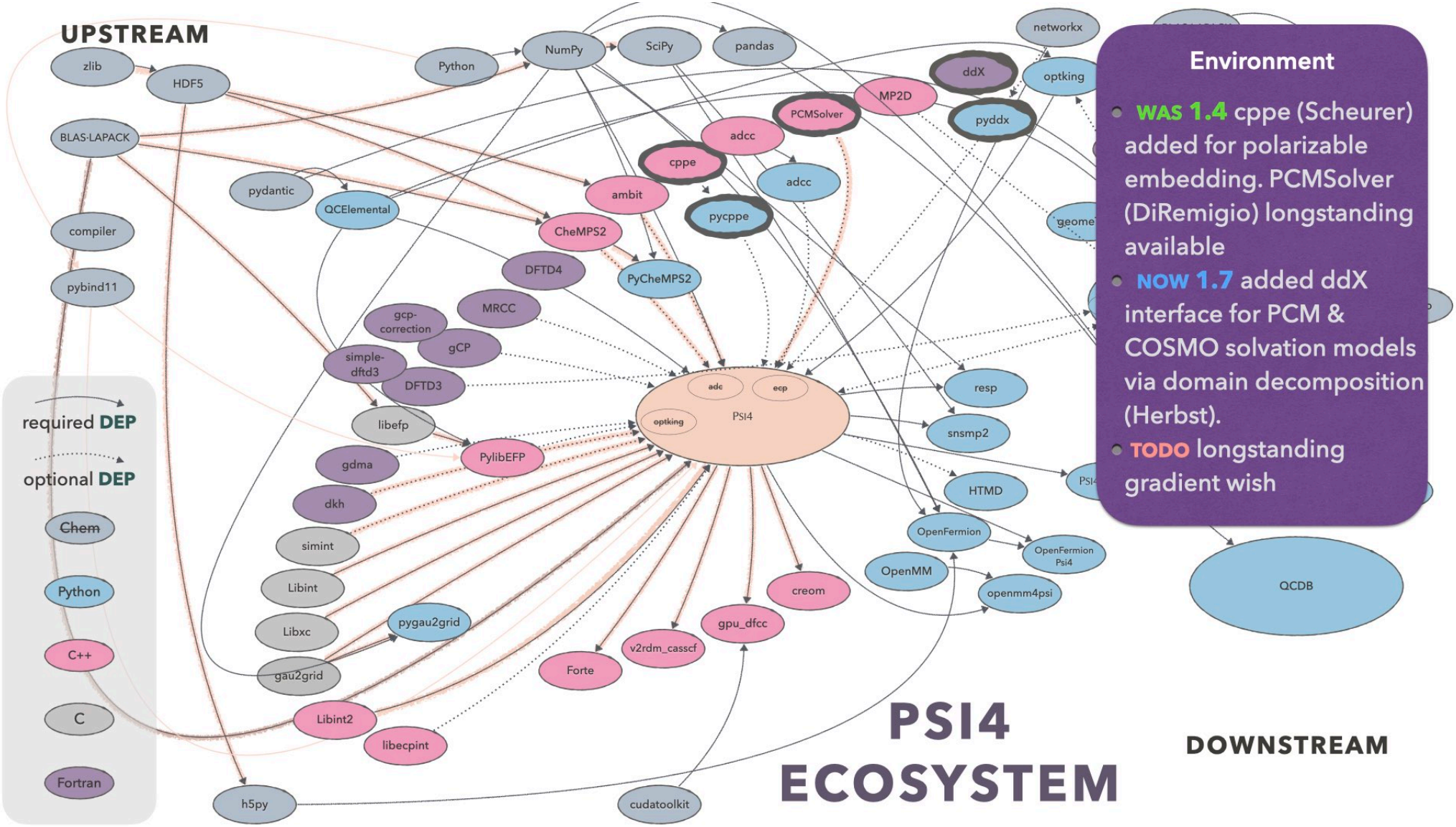
- WAS 1.4 Libint2 for TEI. Sprints at PsiCon 2021 sprints for OEI. Req'd c. 2019 L2 fork.
- NOW 1.6 Libint2 for OEI.
- NOW 1.6 McMurchie-Davidson replaced Obara-Saika for several OEI types
- NOW 1.6 synced with upstream c.2022 L2. Still forked.
- THANKS Simmonett, Scheurer, Turney, Valeev
- TODO resolve ordering so can use same lib as other OSS. Either runtime gss/sss (preferred) or Psi4: sss.
- TODO build high AM pkg.
- TODO resolve erf_nuclear for PseudoSpectral ints.
- TODO sync buildsys

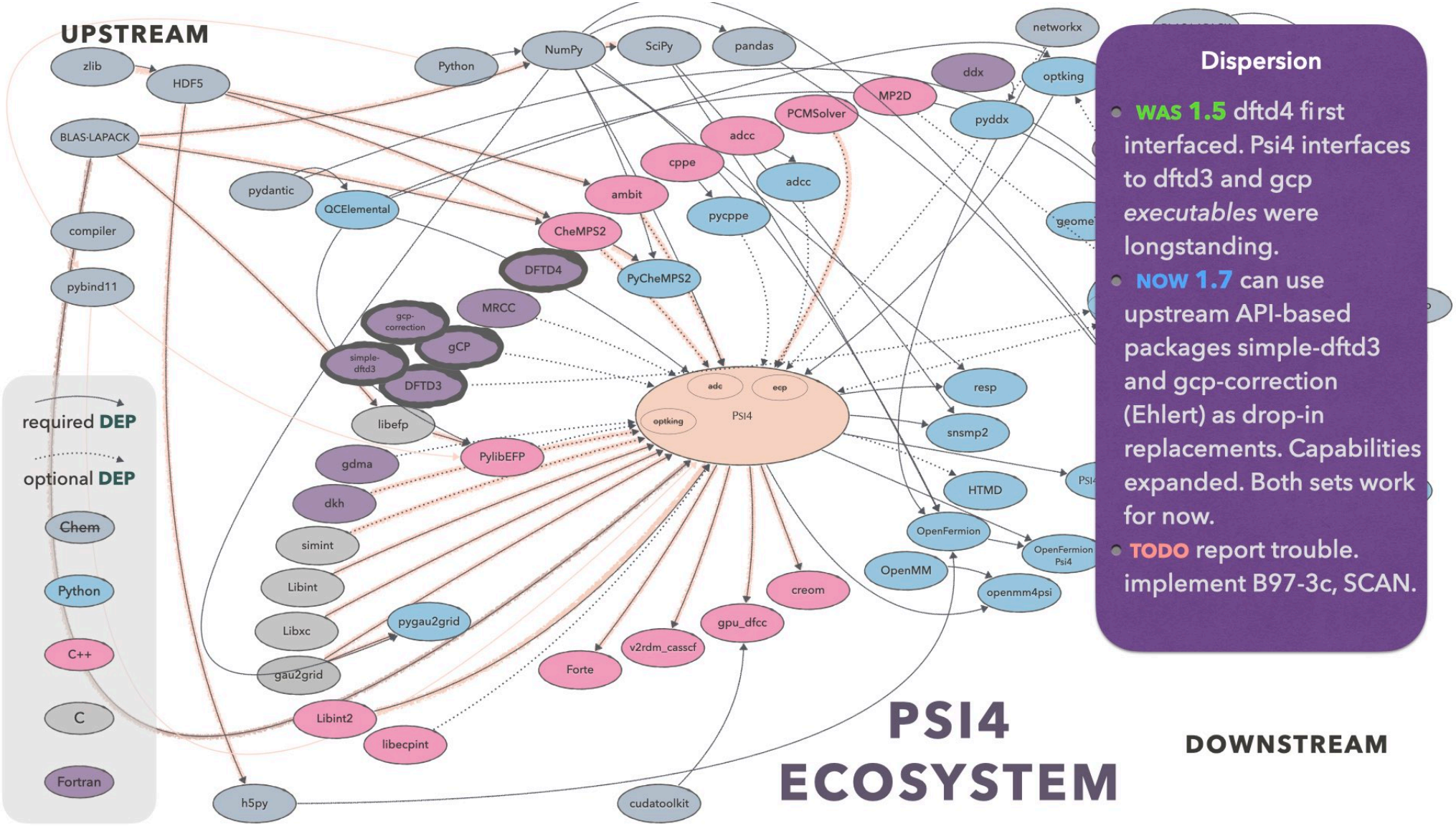

UPSTREAM
zlib
HDF5
BLAS-LAPACK
compiler
pybind11
Python
NumPy
SciPy
pandas
pydantic
QCElemental
networkx
optking
ddx
MP2D
PCMSolver
pyddx
adcc
cppe
adcc
ambit
pycppe
CheMPS2
PyCheMPS2
DFTD4
MRCC
gcp-correction
gCP
simple-dftd3
DFTD3
libefp
PylibEFP
gdma
dkh
simint
Libint
Libxc
gau2grid
pygau2grid
Libint2
libecpint
h5py
Forte
v2rdm_casscf
gpu_dfcc
creom
cudatoolkit
adc
ecp
optking
PSI4
resp
snsmp2
HTMD
OpenFermion
OpenFermion Psi4
OpenMM
openmm4psi
PSI4NUMPY
QCFractal
QCPortal
QCDB
ECP
WAS internal ECP code; early work of Shaw
NOW 1.6 migrated to Shaw's ecpint library
TODO further testing to enable new methods believed to be working
required DEP
optional DEP
Chem
Python
C++
C
Fortran
PSI4 ECOSYSTEM
DOWNSTREAM


UPSTREAM
zlib
HDF5
BLAS-LAPACK
compiler
pybind11
Python
NumPy
SciPy
pandas
pydantic
QCElemental
networkx
optking
ddx
MP2D
pyddx
PCMSolver
adcc
cppe
adcc
ambit
pycppe
CheMPS2
PyCheMPS2
DFTD4
MRCC
gcp-correction
gCP
simple-dftd3
DFTD3
libefp
PylibEFP
gdma
dkh
simint
Libint
Libxc
pygau2grid
gau2grid
Libint2
libecpint
h5py
cudatoolkit
Forte
v2rdm_casscf
gpu_dfcc
creom
adc
ecp
optking
PSI4
resp
snsmp2
HTMD
PSI4NUMPY
QCPortal
OpenFermion
OpenFermion Psi4
OpenMM
openmm4psi
QCDB
required DEP
optional DEP
Chem
Python
C++
C
Fortran
DFT
WAS 1.4 Libxc v4
NOW 1.7 Libxc v5 or v6
TODO libxc_functionals.py file is a simple dict to str map to slurp up functionals. It needs updating from Libxc release notes and tests added.
PSI4 ECOSYSTEM
DOWNSTREAM


UPSTREAM
zlib
HDF5
BLAS-LAPACK
compiler
pybind11
Python
NumPy
SciPy
pandas
pydantic
QCElemental
networkx
optking
ddx
MP2D
PCMSolver
pyddx
adcc
cppe
adcc
ambit
pycppe
CheMPS2
DFTD4
PyCheMPS2
MRCC
gcp-correction
gCP
simple-dftd3
DFTD3
libefp
PylibEFP
gdma
dkh
simint
Libint
Libxc
gau2grid
pygau2grid
Libint2
libecpint
h5py
Forte
v2rdm_casscf
gpu_dfcc
creom
cudatoolkit
adc
ecp
optking
PSI4
resp
snsmp2
HTMD
PSI4NUMPY
OpenFermion
OpenFermion Psi4
OpenMM
openmm4psi
QCFractal
QCPortal
QCDB
MRCC
WAS energy("mrccsdt")
NOW 1.7 similar to switching internal modules:
set qc_module mrcc
energy("ccsdt")
also stores more QCVariables
required DEP
optional DEP
Chem
Python
C++
C
Fortran
PSI4 ECOSYSTEM
DOWNSTREAM


UPSTREAM
zlib
HDF5
BLAS-LAPACK
compiler
pybind11
Python
NumPy
SciPy
pandas
pydantic
QCElemental
networkx
ddx
optking
MP2D
PCMSolver
pyddx
adcc
cppe
adcc
ambit
pycppe
CheMPS2
DFTD4
PyCheMPS2
MRCC
gcp-correction
gCP
simple-dftd3
DFTD3
libefp
PylibEFP
gdma
dkh
simint
Libint
Libxc
gau2grid
pygau2grid
Libint2
libecpint
h5py
Forte
v2rdm_casscf
gpu_dfcc
creom
cudatoolkit
adc
ecp
optking
PSI4
resp
snsmp2
HTMD
OpenFermion
OpenFermion Psi4
OpenMM
openmm4psi
QCDB
ADC
WAS internal ADC module (at least one known bug) with opt-in external ADCC module (Scheurer, Herbst)
NOW 1.7 internal module gone, and you'll need optional ADCC to run more capable adc methods
TODO nothing. Monitor for capability updates.
required DEP
optional DEP
Chem
Python
C++
C
Fortran
PSI4 ECOSYSTEM
DOWNSTREAM

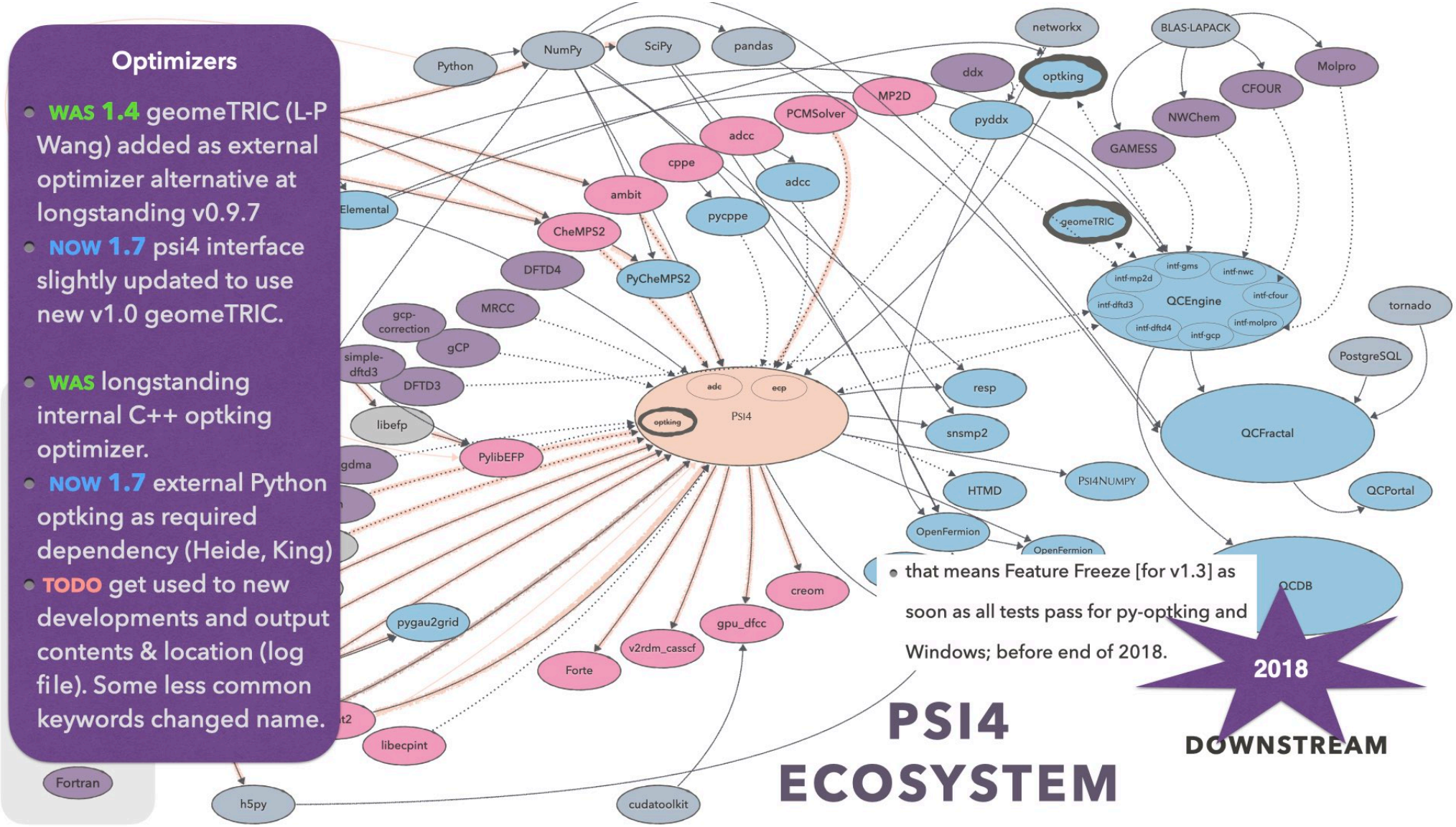

UPSTREAM
zlib
HDF5
BLAS-LAPACK
compiler
pybind11
Python
NumPy
SciPy
pandas
pydantic
QCElemental
networkx
optking
ddX
MP2D
PCMSolver
pyddx
adcc
cppe
adcc
ambit
pycppe
CheMPS2
DFTD4
PyCheMPS2
MRCC
gcp-correction
gCP
simple-dftd3
DFTD3
libefp
PylibEFP
gdma
dkh
simint
Libint
Libxc
gau2grid
pygau2grid
Libint2
libecpint
h5py
cudatoolkit
Forte
v2rdm_casscf
gpu_dfcc
creom
adc
ecp
optking
PSI4
resp
snsmp2
HTMD
OpenFermion
OpenFermion Psi4
OpenMM
openmm4psi
QCDB
required DEP
optional DEP
Chem
Python
C++
C
Fortran
Environment
WAS 1.4 cppe (Scheurer) added for polarizable embedding. PCMSolver (DiRemigio) longstanding available
NOW 1.7 added ddX interface for PCM & COSMO solvation models via domain decomposition (Herbst).
TODO longstanding gradient wish
PSI4 ECOSYSTEM
DOWNSTREAM


UPSTREAM
zlib
HDF5
BLAS-LAPACK
compiler
pybind11
Python
NumPy
SciPy
pandas
pydantic
QCElemental
CheMPS2
DFTD4
PyCheMPS2
MRCC
gcp-correction
gCP
simple-dftd3
DFTD3
libefp
PylibEFP
gdma
dkh
simint
Libint
Libxc
gau2grid
pygau2grid
Libint2
libecpint
h5py
cudatoolkit
Forte
v2rdm_casscf
gpu_dfcc
creom
adcc
cppe
ambit
pycppe
adcc
MP2D
PCMSolver
ddx
pyddx
networkx
optking
resp
snsmp2
HTMD
OpenFermion
OpenFermion Psi4
OpenMM
openmm4psi
adc
ecp
optking
PSI4
required DEP
optional DEP
Chem
Python
C++
C
Fortran
PSI4 ECOSYSTEM
DOWNSTREAM
Dispersion
WAS 1.5 dftd4 first interfaced. Psi4 interfaces to dftd3 and gcp executables were longstanding.
NOW 1.7 can use upstream API-based packages simple-dftd3 and gcp-correction (Ehlert) as drop-in replacements. Capabilities expanded. Both sets work for now.
TODO report trouble. implement B97-3c, SCAN.

Optimizers

- **WAS 1.4** geomeTRIC (L-P Wang) added as external optimizer alternative at longstanding v0.9.7
- **NOW 1.7** psi4 interface slightly updated to use new v1.0 geomeTRIC.

- **WAS** longstanding internal C++ optking optimizer.
- **NOW 1.7** external Python optking as required dependency (Heide, King)
- **TODO** get used to new developments and output contents & location (log file). Some less common keywords changed name.



- that means Feature Freeze [for v1.3] as soon as all tests pass for py-optking and Windows; before end of 2018.

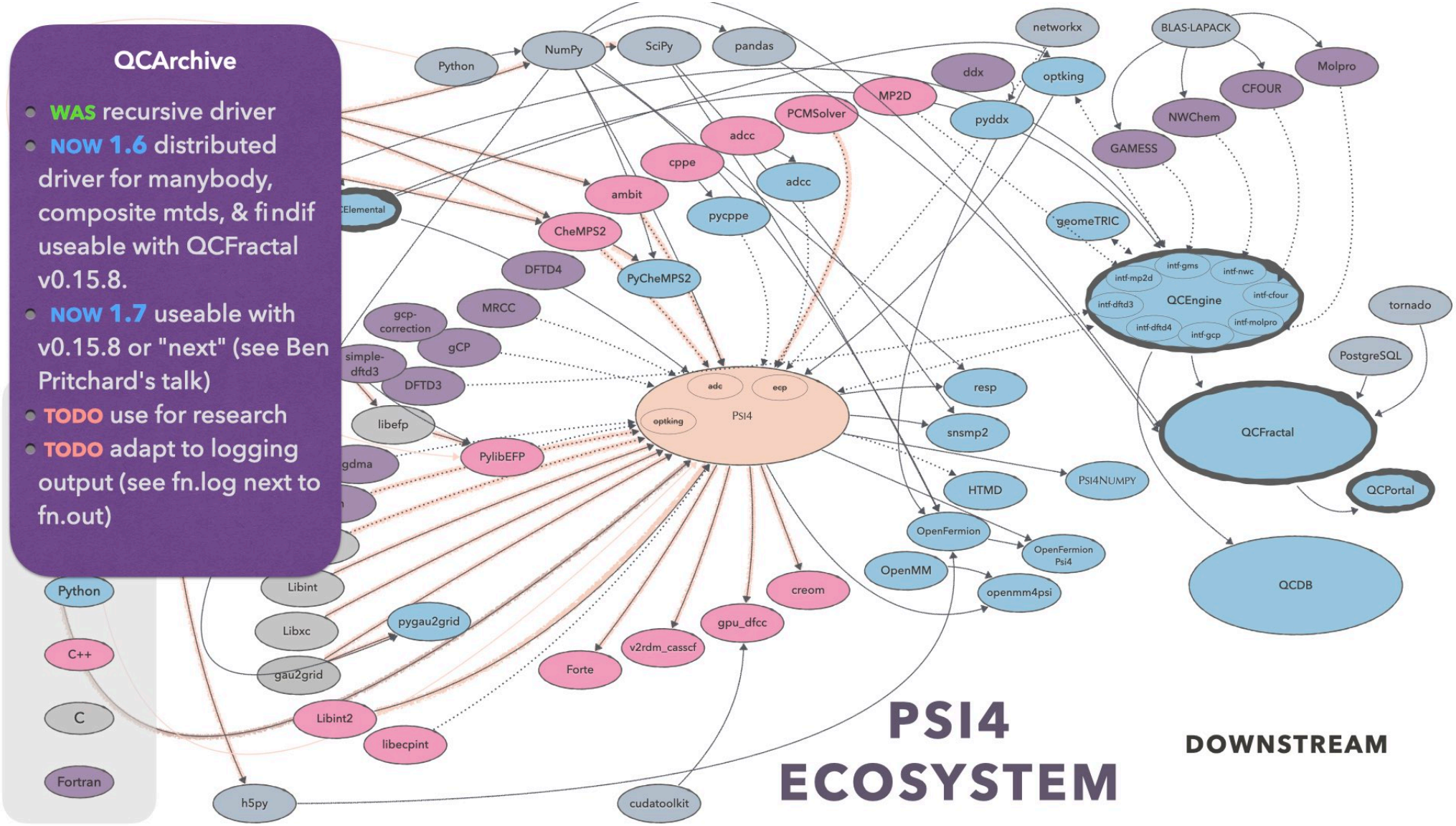
2018

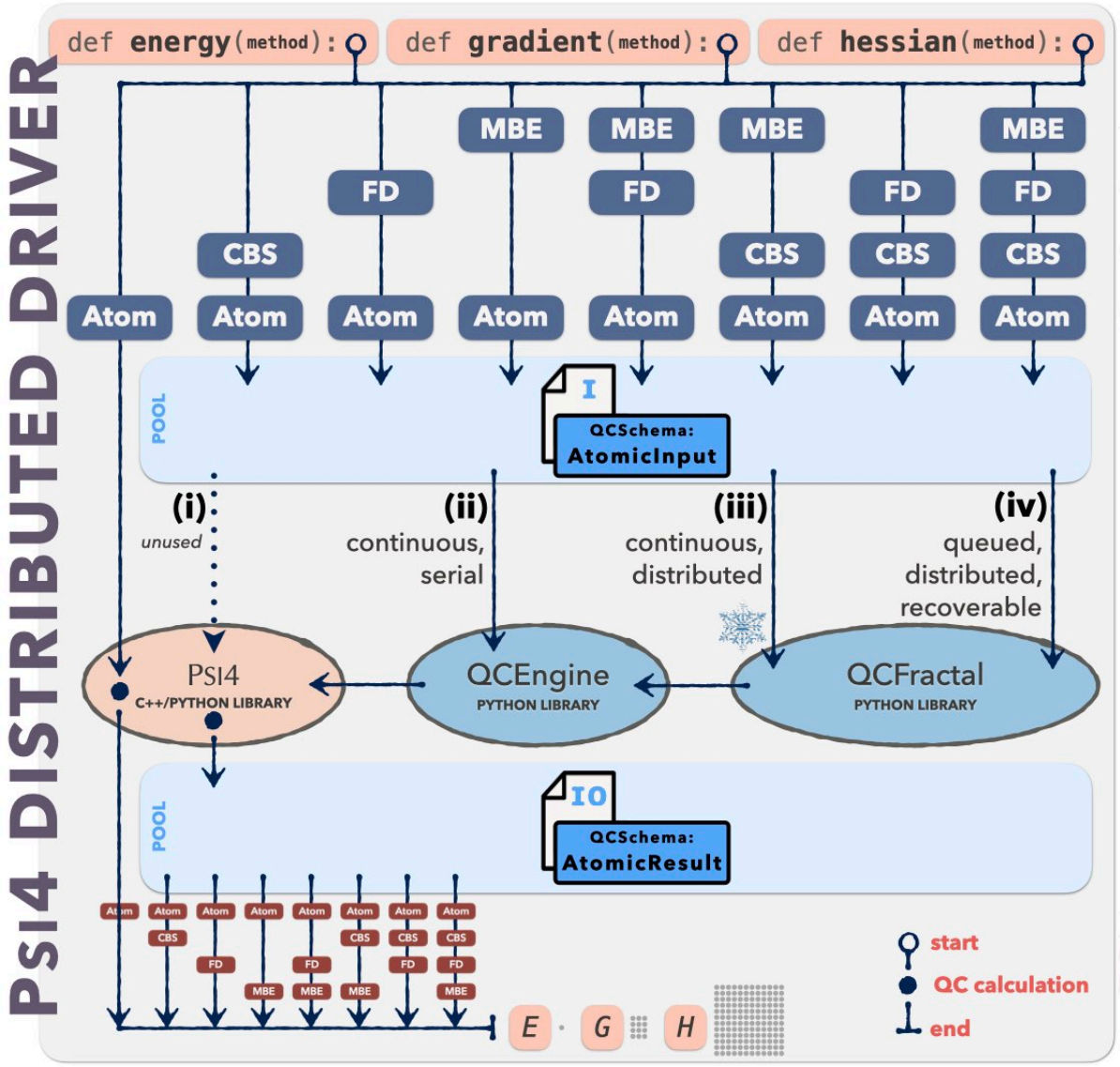
PSI4 ECOSYSTEM

DOWNSTREAM

QCArchive

- **WAS** recursive driver
- **NOW 1.6** distributed driver for manybody, composite mtds, & findif useable with QCFractal v0.15.8.
- **NOW 1.7** useable with v0.15.8 or "next" (see Ben Pritchard's talk)
- **TODO** use for research
- **TODO** adapt to logging output (see fn.log next to fn.out)






```
class ManyBodyComputer ():
```

PLAN Separate **molecule** into subsystems. CP, noCP, VMFC basis. **method** unchanged.

```
for frag in fragments: return qcschema
```

ASM Assemble n-body & interaction results from fragments.

```
class FiniteDifferenceComputer ():
```

PLAN Displace **molecule** according to stencil. Reference **molecule** & **method** unchanged.

```
for disp in displacements: return qcschema
```

ASM Assemble derivative results from displacements.

```
class CompositeComputer ():
```

PLAN Separate **method** into method, basis, & extrapolations. **molecule** unchanged.

```
mp2/cc-pv[tq]z  →  MP2 TOTAL ENERGY/cc-pVTZ
                   MP2 TOTAL ENERGY/cc-pVQZ
for mc in modelchems: return qcschema
```

ASM Assemble extrapolations & total results from modelchems.

```
class AtomicComputer ():
```

PLAN **molecule** & **method** unchanged. `return qcschema`

ASM Return analytic energy, gradient, or Hessian.

QCEngine: Value Added

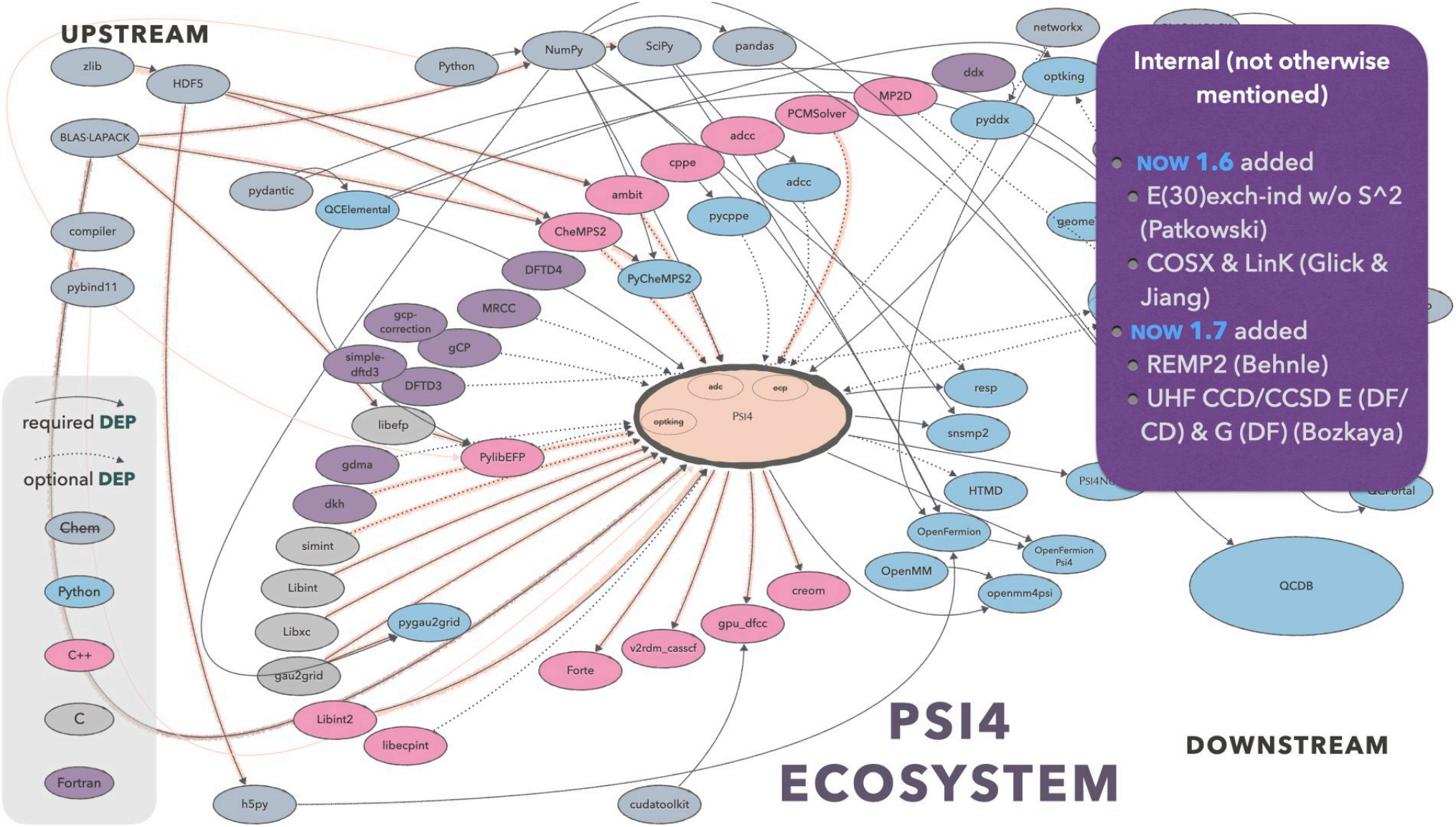
results (obviously), runtime info, structured results, error classification

- **ERROR** QCEngine attempts classification based on restartability.
  - **`InputError`** incorrect user parameters, not recoverable.
  - **`ResourceError`** not enough memory, cores, disk present. Recoverable on different box.
  - **`RandomError`** likely recoverable like segfaults or disk I/O.
  - **`UnknownError`** cannot classify error type.
- **EXTRAS** free communication outside or pre-QCSchema.
- **PROPERTIES** secondary scalar or array results harvested in atomic units.
- **PROTOCOLS** configure data layout (e.g., suppress stdout or include orbital energies).
- **PROVENANCE** program identity and version, runtime cpu, memory, walltime.
- **RETURN_RESULT** primary scalar or array result cooresponding to **DRIVER**.
- **STDOUT/STDERR** customary human logfile output.
- **WAVEFUNCTION** basis-based array results in CCA ordering.

```
{'driver': <DriverEnum.energy: 'energy'>,
 'error': None,
 'extras': {'psiapi': True,
            'psiapi_evaluated': True,
            'qcvars': {'CURRENT DIPOLE': array([0., 0., 0.]),
                       'CURRENT ENERGY': -2.807913354492939,
                       'CURRENT REFERENCE ENERGY': -2.807913354492939,
                       'HF TOTAL ENERGY': -2.807913354492939,
                       'NUCLEAR REPULSION ENERGY': 0.0,
                       'ONE-ELECTRON ENERGY': -3.8634969002750466,
                       'SCF DIPOLE': array([0., 0., 0.]),
                       'SCF DIPOLE X': 0.0,
                       'SCF DIPOLE Y': 0.0,
                       'SCF DIPOLE Z': 0.0,
                       'SCF ITERATION ENERGY': -2.807913354492939,
                       'SCF ITERATIONS': 2.0,
                       'SCF TOTAL ENERGY': -2.807913354492939,
                       'TWO-ELECTRON ENERGY': 1.0555835457821074}},
 'id': None,
 'keywords': {},
 'model': {'basis': 'sto-3g', 'method': 'hf'},
 'molecule': {'atom_labels': array([''], dtype='<U1'),
              'atomic_numbers': array([2], dtype=int16),
              'fix_com': False,
              'fix_orientation': False,
              'fragment_charges': [0.0],
              'fragment_multiplicities': [1],
              'fragments': [array([0], dtype=int32)],
              'geometry': array([[0., 0., 0.]]),
              'mass_numbers': array([4], dtype=int16),
              'masses': array([4.00260325]),
              'molecular_charge': 0.0,
              'molecular_multiplicity': 1,
              'name': 'He',
              'provenance': {'creator': 'QCElemental',
                             'routine': 'qcelemental.molparse.from_string',
                             'version': 'v0.16.0'},
              'real': array([ True]),
              'schema_name': 'qcschema_molecule',
              'schema_version': 2,
              'symbols': array(['He'], dtype='<U2'),
              'validated': True},
```

```
 'properties': {'calcinfo_nalpha': 1,
                'calcinfo_natom': 1,
                'calcinfo_nbasis': 1,
                'calcinfo_nbeta': 1,
                'calcinfo_nmo': 1,
                'nuclear_repulsion_energy': 0.0,
                'return_energy': -2.807913354492939,
                'scf_dipole_moment': [0.0, 0.0, 0.0],
                'scf_iterations': 2,
                'scf_one_electron_energy': -3.8634969002750466,
                'scf_total_energy': -2.807913354492939,
                'scf_two_electron_energy': 1.0555835457821074},
 'protocols': {'stdout': True},
 'provenance': {'cpu': 'Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz',
                'creator': 'Psi4',
                'hostname': 'psinet',
                'memory': 0.524,
                'module': 'scf',
                'nthreads': 1,
                'qcengine_version': 'v0.16.0+2.gba6b49b',
                'routine': 'psi4.schema_runner.run_qcschema',
                'username': 'psilocaluser',
                'version': '1.4a2.dev1180',
                'wall_time': 0.6296589374542236},
 'return_result': -2.807913354492939,
 'schema_name': 'qcschema_output',
 'schema_version': 1,
 'stderr': None,
 'stdout': '\n'
           '    '
           '---------------------------------------------------------\n'
           ' Psi4: An Open-Source Ab Initio Electronic Structure Package\n'
           '                      Psi4 1.4a2.dev1180 \n'
           ' . . . \n'

           '\n'
           '    Psi4 stopped on: Tuesday, 15 September 2020 04:36AM\n'
           '    Psi4 wall time for execution: 0:00:00.37\n'
           '\n'
           '*** Psi4 exiting successfully. Buy a developer a beer!\n',
 'success': True,
 'wavefunction': None}
```


UPSTREAM
zlib
HDF5
BLAS-LAPACK
compiler
pybind11
Python
NumPy
SciPy
pandas
pydantic
QCElemental
networkx
ddx
optking
MP2D
PCMSolver
pyddx
adcc
cppe
adcc
ambit
pycppe
CheMPS2
PyCheMPS2
DFTD4
MRCC
gcp-correction
gCP
simple-dftd3
DFTD3
libefp
PylibEFP
gdma
dkh
simint
Libint
Libxc
gau2grid
pygau2grid
Libint2
libecpint
h5py
cudatoolkit
Forte
v2rdm_casscf
gpu_dfcc
creom
adc
ecp
optking
PSI4
resp
snsmp2
HTMD
OpenFermion
OpenFermion Psi4
OpenMM
openmm4psi
QCDB
required DEP
optional DEP
Chem
Python
C++
C
Fortran
Internal (not otherwise mentioned)
NOW 1.6 added
E(30)exch-ind w/o S^2 (Patkowski)
COSX & LinK (Glick & Jiang)
NOW 1.7 added
REMP2 (Behnle)
UHF CCD/CCSD E (DF/CD) & G (DF) (Bozkaya)
PSI4 ECOSYSTEM
DOWNSTREAM

WAR ON GLOBALS

major offensive in 2022

process.h c. v1.0

```
class Process
{
public:
    class Environment
    {
        std::map<std::string, std::string> environment_;
        unsigned long int memory_;
        int nthread_;

        std::shared_ptr<Molecule> molecule_;
        SharedMatrix gradient_;
        std::shared_ptr<efp::EFP> efp_;
        SharedMatrix efp_torque_;
        std::shared_ptr<Vector> frequencies_;
        std::shared_ptr<PointGroup> parent_symmetry_;

        std::shared_ptr<Molecule> legacy_molecule_;
        std::shared_ptr<Wavefunction> legacy_wavefunction_;
    public:
        void initialize();

        /// The symmetry of the molecule, before any displacements have been made
        std::shared_ptr<PointGroup> parent_symmetry() { return parent_symmetry_; }
        /// Set the "parent" symmetry
        void set_parent_symmetry(std::shared_ptr<PointGroup> pg) { parent_symmetry_ = pg; }
        const std::string& operator()(const std::string& key) const;
        std::string operator()(const std::string& key);
        const std::string set(const std::string& key, const std::string& value);

        /// Set active molecule
        void set_molecule(const std::shared_ptr<Molecule>& molecule);
        /// Return active molecule
        std::shared_ptr<Molecule> molecule() const;


        /// Temporary slots for legacy as a stop-gap
        /// Set active molecule
        void set_legacy_molecule(const std::shared_ptr<Molecule>& molecule);
        /// Return active molecule
        std::shared_ptr<Molecule> legacy_molecule() const;

        /// Set wavefunction
        void set_legacy_wavefunction(const std::shared_ptr<Wavefunction>& wavefunction);
        /// Get wavefunction
        std::shared_ptr<Wavefunction> legacy_wavefunction() const;


        /// Set gradient manually
        void set_gradient(const SharedMatrix g) { gradient_ = g; }
        /// Get gradient manually
        SharedMatrix gradient() const { return gradient_; }

        /// Set frequencies manually
        void set_frequencies(const std::shared_ptr<Vector> f) { frequencies_ = f; }
        /// Get frequencies manually
        std::shared_ptr<Vector> frequencies() const { return frequencies_; }

        /// Set EFP
        void set_efp(const std::shared_ptr<psi::efp::EFP>& efp) { efp_ = efp; }
        /// Get EFP
        std::shared_ptr<psi::efp::EFP> get_efp() const { return efp_; }

        /// Set EFP gradient manually
        void set_efp_torque(const SharedMatrix g) { efp_torque_ = g; }
        /// Get EFP gradient manually
        SharedMatrix efp_torque() const { return efp_torque_; }

        /// Map containing current energies
        std::map<std::string, double> globals;

        /// Map containing current arrays
        std::map<std::string, std::shared_ptr<Matrix> > arrays;

        /// Number of threads per process
        int get_n_threads() const;
        void set_n_threads(int nthread);

        /// Memory in bytes
        unsigned long int get_memory() const;
        void set_memory(unsigned long int m);

        /// "Global" liboptions object.
        Options options;
    };

    static Environment environment;

    static Environment get_environment();
```

v1.3

v1.4

v1.7

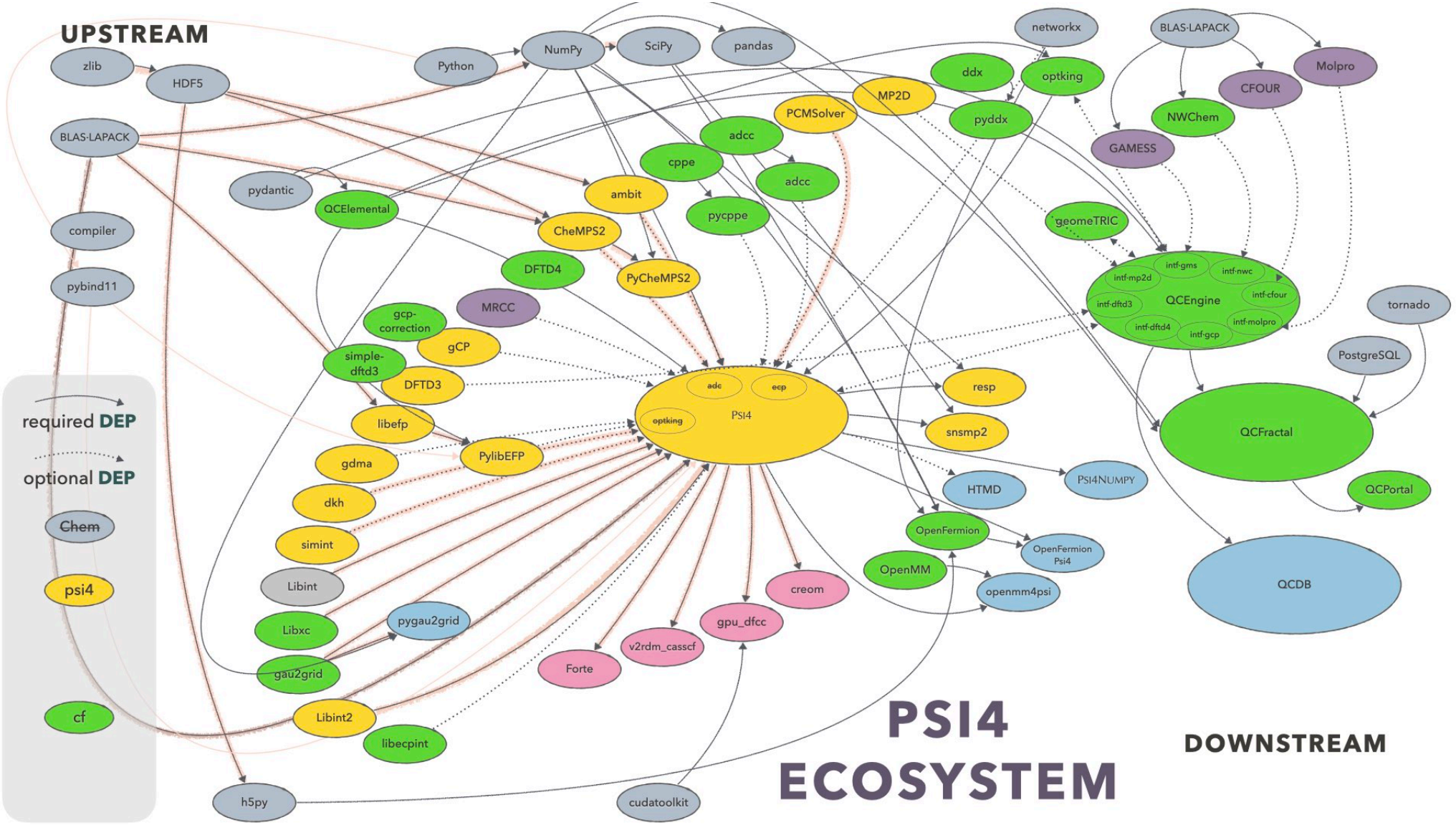
v1.4, findif / parent symmetry

v1.2, direct qcvariables

v1.7, optking gradient & mol, old plugin

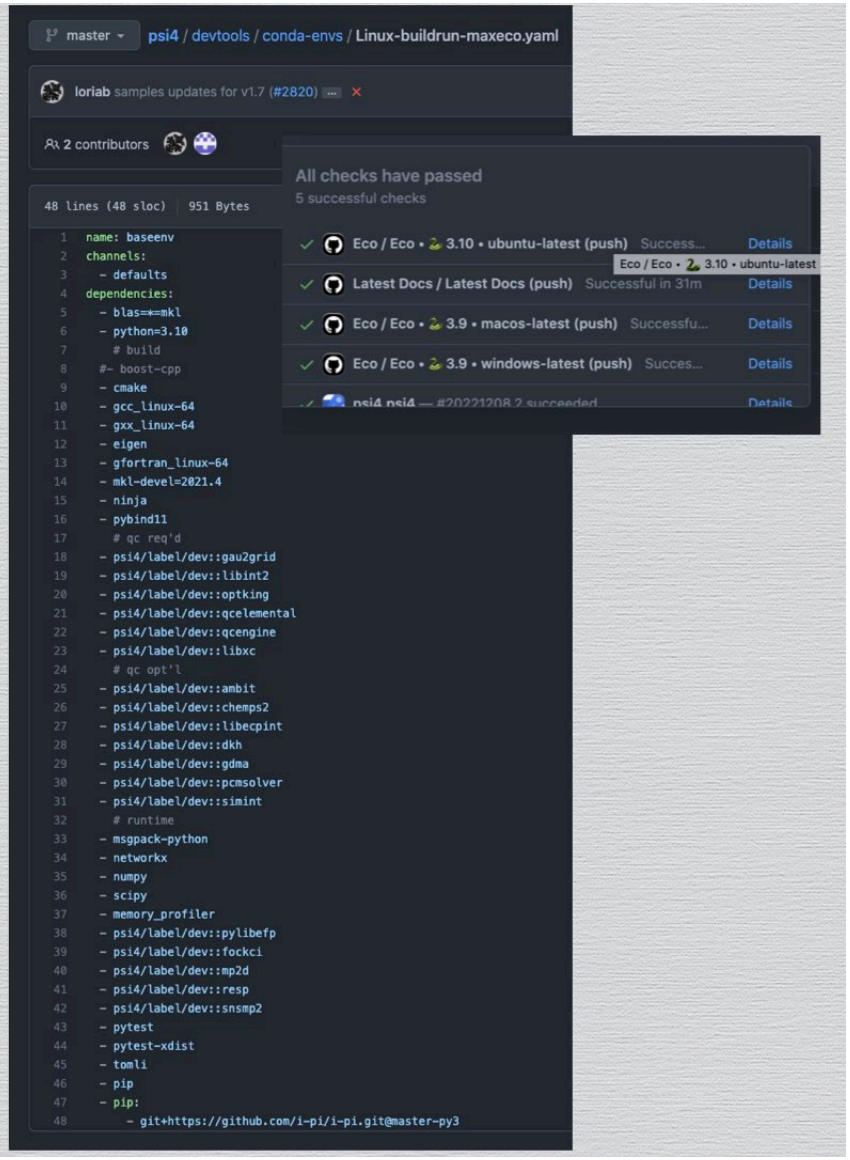
v1.3, efp upon mol parsing

- What's left?
- **MOLECULE** where "active" mol lives
  - only pseudo-global since C++ uses mol from wfn
  - can `energy(..., molecule=mol_instance)` so avoidable
- **GLOBALS & ARRAYS** where QCVariables live
  - increasingly a duplicate of wfn.variables() written back to global memory as a calc finishes
  - largely avoidable except for CC (WIP) and SAPT
- **MEMORY & THREADS** integers
- **OPTIONS** where current keyword state lives
  - very much still a global accessed by C++ modules
  - user-side, plan to allow `energy(..., keywords=kw_dict)` so calls self-contained. append or truncate?


UPSTREAM
zlib
HDF5
BLAS-LAPACK
compiler
pybind11
Python
NumPy
SciPy
pandas
pydantic
QCElemental
ambit
CheMPS2
DFTD4
PyCheMPS2
MRCC
gcp-correction
gCP
simple-dftd3
DFTD3
libefp
PylibEFP
gdma
dkh
simint
Libint
Libxc
pygau2grid
gau2grid
Libint2
libecpint
h5py
cudatoolkit
Forte
v2rdm_casscf
gpu_dfcc
creom
adcc
cppe
pycppe
adcc
PCMSolver
MP2D
ddx
pyddx
networkx
optking
BLAS-LAPACK
Molpro
CFOUR
NWChem
GAMESS
geomeTRIC
QCEngine
intf-gms
intf-nwc
intf-mp2d
intf-cfour
intf-dftd3
intf-molpro
intf-dftd4
intf-gcp
tornado
PostgreSQL
QCFractal
QCPortal
QCDB
adc
ecp
optking
PSI4
resp
snsmp2
HTMD
PSI4NUMPY
OpenFermion
OpenFermion Psi4
OpenMM
openmm4psi
required DEP
optional DEP
Chem
psi4
cf
PSI4 ECOSYSTEM
DOWNSTREAM

PACKAGING

- **PSI4 CHANNEL** has a lot of projects
  - **-C PSI4** is a nice one-stop shop for Psi4 end users
  - but projects **DOWNSTREAM** a robust environment from multiple channels is getting harder
  - even within psi4, solving a full **ECOSYSTEM** is getting harder
- **CONDA-FORGE** is community conda packaging
  - **LIBINT2** only suitable for psi4 channel at present.
  - **SIMPLE-DFTD3, GCP-CORRECTION, DFTD4-PYTHON, CPPE, GEOMETRIC, QCELEMENTAL, QCENGINE, QCFRACTAL, ADCC, DDX, OPTKING, GEOMETRIC, OPENFERMION, GAU2GRID, LIBXC** now on c-f. We copy over a few pure Py. Others we either have to duplicate or require an involved setup.
- **OS/ARCHITECTURES**
  - **WINDOWS** Psi4 already c-f based: `conda install psi4 -c psi4 -c conda-forge`
  - **OSX-ARM64** has fundamentals on c-f through cross-compilation. I suggest any Psi4 support be through this base.
  - **OSX-64** Psi4 maintains so that Psi4Education has a complete set and devs have dependencies available, not from any production-quality binaries argument. The deps could shift to c-f. Or kill off? macpsinet has died.
- **WINDOWS NIGHTLY** now conda package built on each commit to master through Azure and has been very reliable.
- **LINUX NIGHTLY** reccommended for GHA

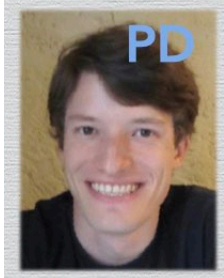


TEST SUITE

- **GREAT BOTHER** it is to compose tests. But worth it
  - **CONSOLIDATE** and define gains
  - **OUTSOURCE** fixing problems or discussing changes to PRs that introduce conflicts
  - **FACILITATE** refactoring and improving
- **TEST SUITE** proven pretty robust over pybind11/inversion, libint2, distributed driver
- **STANDARD SUITE** systematic testing of many QC methods
  - **COVERS** R/U/ROHF reference, CONV/DF/CD algorithms, analytic vs. findif
  - **ALGORITHM** all of CONV, DF, CD, plus checking DF near CONV
  - **CONTRACT** CURRENT variables and Wfn variables and returns
  - **NEWLY EXPANDED** expanded to more methods, gradients, modules
  - **NEWLY CHECKED** against other QC programs
  - **NEWLY EMPLOYED** to produce fine-grained capabilities tables for docs
  - **TODO** expand mols to corner cases like atoms, no beta electrons, ghost atoms
  - **TODO** expand to properties like dipoles (Jonathon has some systematic tests going)
  - **HELP NEW CONTRIBUTIONS** check correctness and integration with Psi4's plumbing
- **AT LONG LAST** one can run the whole test suite (1) in one pytest call and (2) from installed copy
- **REMEMBER** a few differences between ctest and pytest
  - **PARALLELISM** ctest `-j6` becomes pytest `-n6`
  - **SUBSET BY LABEL** ctest `-L quick` becomes pytest `-m quick` or logical `-m "quick and not scf"`
  - **SUBSET BY NAME** ctest `-R fnocc` becomes pytest `-k fnocc` or logical `-m "fnocc and fc and not rohf"`
  - **ADDONS** detected at runtime, not cmake configure time
  - **ERROR** and warning and deprecation pathways are easy to test

Summary capabilities of Psi4. "✓" runs analytically. "::" runs derivative with internal finite difference. Double underline "✓" selector (e.g., CC_TYPE) unspecified.

□	REFERENCE →	Restricted (RHF)										Unrestricted (UHF)									
name ↓ →	□	energy()						gradient()[8]				energy()						gradient()[8]			
□	type[6] ↓ →	CV		DF		CD		CV		DF		CD		CV		DF		CD		CV		DF		CD	
□	FREEZE_CORE[7] →	A	F	A	F	A	F	A	F	A	F	A	F	A	F	A	F	A	F	A	F	A	F	A	F
hf	SCF_TYPE	✓		✓		✓		✓		✓		::		✓		✓		✓		✓		✓		::	
mp2	MP2_TYPE	✓	✓	✓	✓	✓	✓	✓	::	✓	✓	::	::	✓	✓	✓	✓	✓	✓	✓	::	✓	✓	::	::
mp2.5	MP_TYPE	✓	✓	✓	✓	✓	✓	✓	::	✓	✓	::	::	✓	✓	✓	✓	✓	✓	✓	::	✓	✓	::	::
mp3	MP_TYPE	✓	✓	✓	✓	✓	✓	✓	::	✓	✓	::	::	✓	✓	✓	✓	✓	✓	✓	::	✓	✓	::	::
mp4(sdq)	MP_TYPE	✓	✓																						
mp4, mp*n*[9]	MP_TYPE	✓	✓																						
zapt2, zapt*n*[9]	MP_TYPE																								
cisd, ci*n*[9]	CI_TYPE	✓	✓																						
qcisd	CI_TYPE	✓	✓																						
qcisd(t)	CI_TYPE	✓	✓																						
fci	CI_TYPE	✓	✓																						
remp2	CC_TYPE	✓	✓	✓	✓	✓	✓							✓	✓	✓	✓	✓	✓						
lccd	CC_TYPE	✓	✓	✓	✓	✓	✓	✓	::	✓	✓	::	::	✓	✓	✓	✓	✓	✓	✓	::	✓	✓	::	::
lccsd, cepa(0)	CC_TYPE	✓	✓																						
cepa(1)	CC_TYPE	✓	✓																						
cepa(3)	CC_TYPE	✓	✓																						
acpf	CC_TYPE	✓	✓																						
aqcc	CC_TYPE	✓	✓																						
ccd	CC_TYPE			✓	✓	✓	✓			✓	✓	::	::			✓	✓	✓	✓			✓	✓	::	::
bccd	CC_TYPE	✓	✓											✓	✓										
cc2	CC_TYPE	✓	✓					✓	::					✓	✓					::	::				
ccsd	CC_TYPE	✓	✓	✓	✓	✓	✓	✓	::	✓	✓	::	::	✓	✓	✓	✓	✓	✓	✓	::	✓	✓	::	::
ccsd(t)	CC_TYPE	✓	✓	✓	✓	✓	✓	✓	::	✓	✓	::	::	✓	✓					✓	::				



Jerome Gonthier
Berkeley

Rob Parrish
Stanford

Holger Kruse
Czech Acad. Sci.

Rollin King
Bethel

Alexander Sokolov
Ohio State

David Sherrill
GaTech

Lori Burns
GaTech

Asim Alenaizan
GaTech

Maximilian Scheurer
Heidelberg

Zach Glick
GaTech

Jeff Schriber
GaTech

Francesco Evangelista
Emory

Eugene DePrince
FSU

Fritz Schaefer
UGA

Justin Turney
UGA

Daniel Crawford
VaTech

Daniel Smith
MolSSI

Konrad Patkowski
Auburn

Ben Pritchard
MolSSI

Jonathon Misiewicz
Emory

Ed Valeev
VaTech

Andy Simmonett
NIH

Ed Hohenstein
CCNY

Roberto Di Remigio
Tromso

Ugur Bozkaya
Hacettepe

Peter Kraus
Curtin

Susi Lehtola
MolSSI