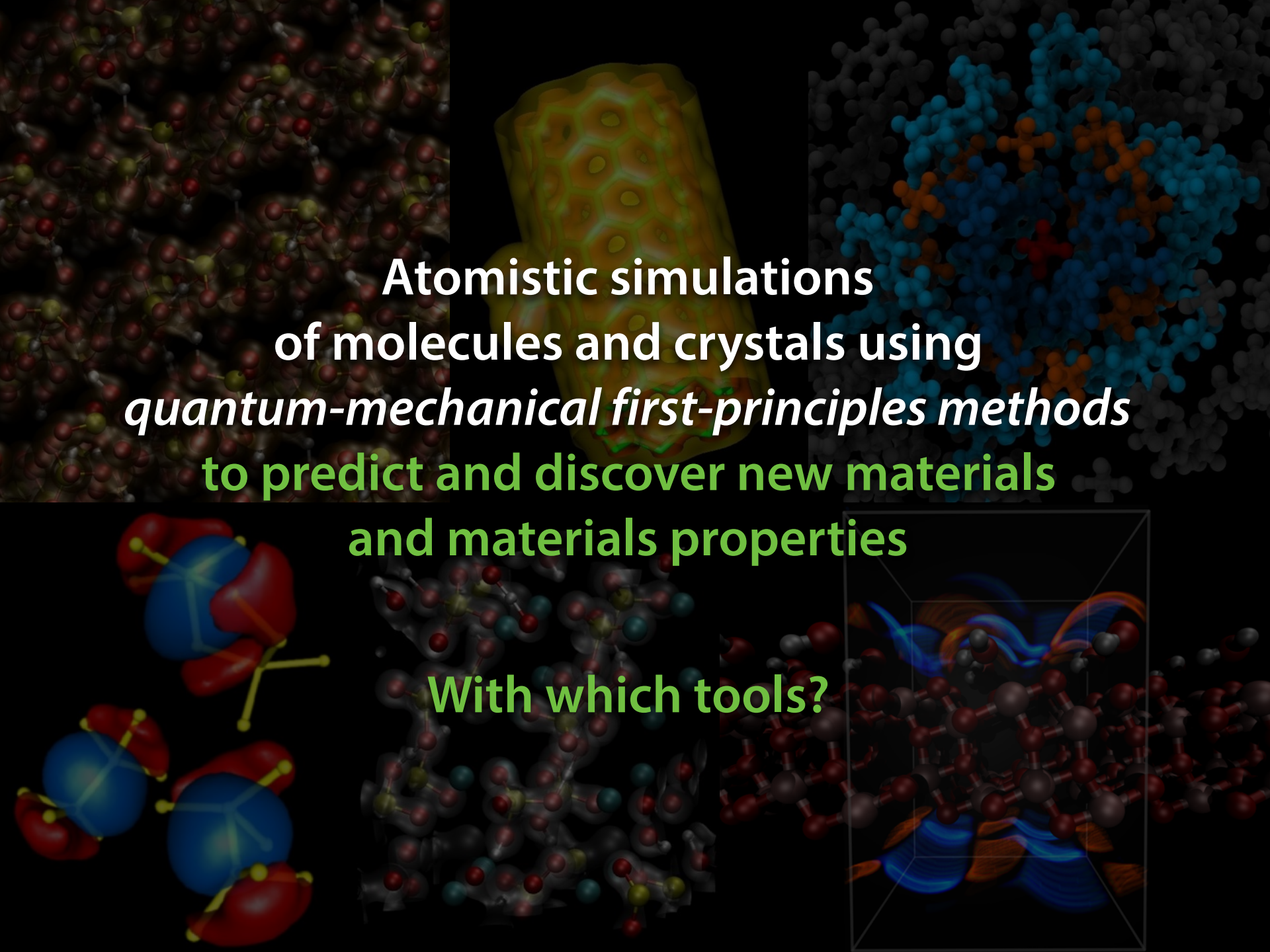
The ADES model for Computational Materials Science and its implementation in AiiDA

Giovanni Pizzi
Theory and Simulation of Materials, EPFL, Switzerland

THEOS
THEORY AND SIMULATION OF MATERIALS

MARVEL

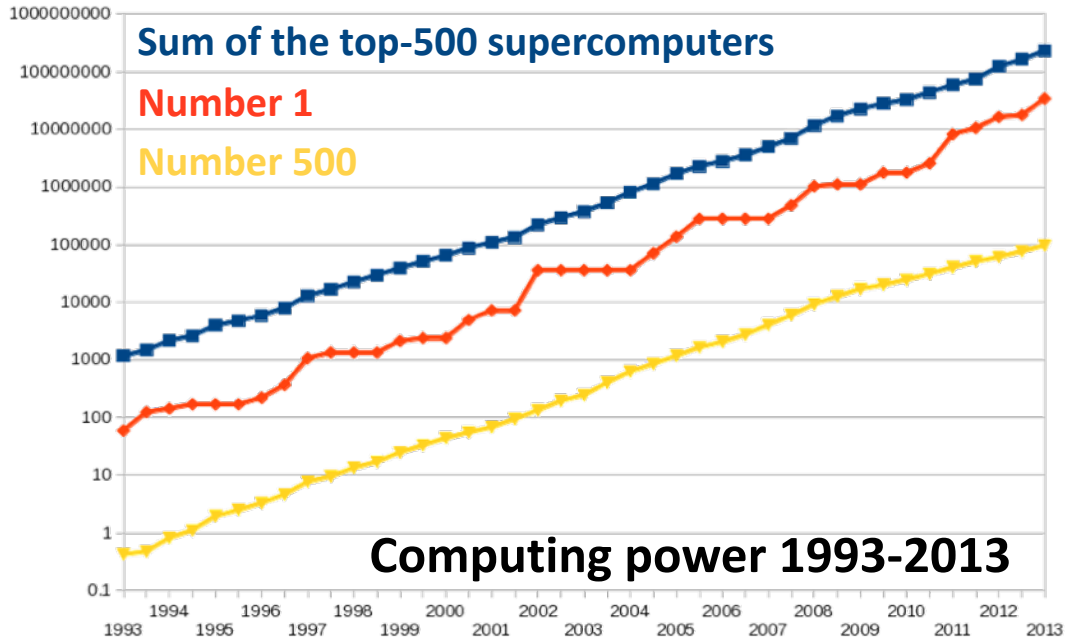
ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE



Atomistic simulations
of molecules and crystals using
quantum-mechanical first-principles methods
to predict and discover new materials
and materials properties

With which tools?

Accuracy and predictive power of quantum engines



150,000x increase in the past 20 years

1 month (1993)

↓

10 seconds (2015)

Result: materials design and discovery via

high-throughput computations

...but did we think at how to manage simulations?

We run *computer* laboratories:
we should manage *simulations* and *data* with a dedicated platform

But how should the platform be?

Computational science should be...

reproducible

Often not possible from the data reported in papers

searchable

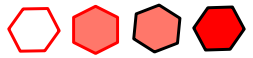
Find existing calculations, reuse and data-mine results

reliable

Results persisted in repositories, automated procedures to reduce errors and verify results

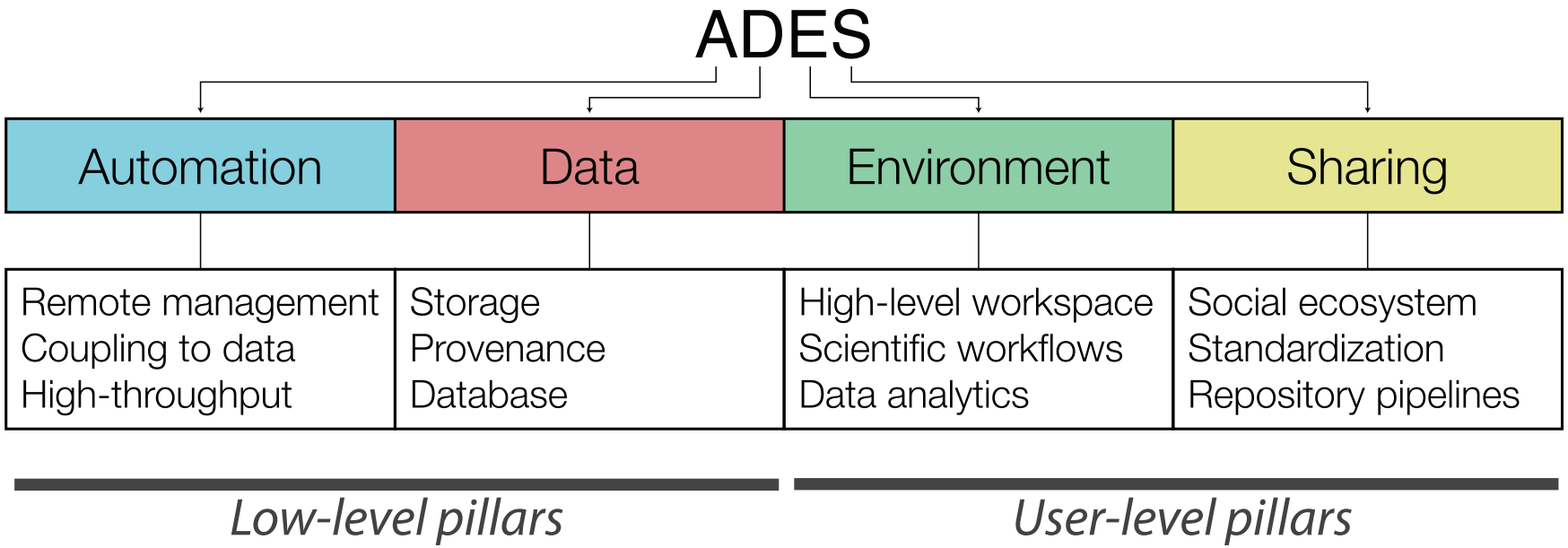
shareable

Community to share results, cross-validate them, and boost scientific discovery



THE ADES MODEL

We have encoded these requirements in the four pillars for a computational science infrastructure



G. Pizzi et al., Comp. Mat. Sci 111, 218-230 (2016)

AiiDA

AiiDA
Automated Interactive Infrastructure and Database for Computational Science

http://www.aiida.net

Oct 2012: First commit to current AiiDA repository

7tutorials from October 2014 to 2017 (Lausanne, Kyoto, Zurich, Berlin, Trieste)

Feb 2015: First public open release of AiiDA (0.4.0 on bitbucket + readthedocs)

Early codes by Boris Kozinsky

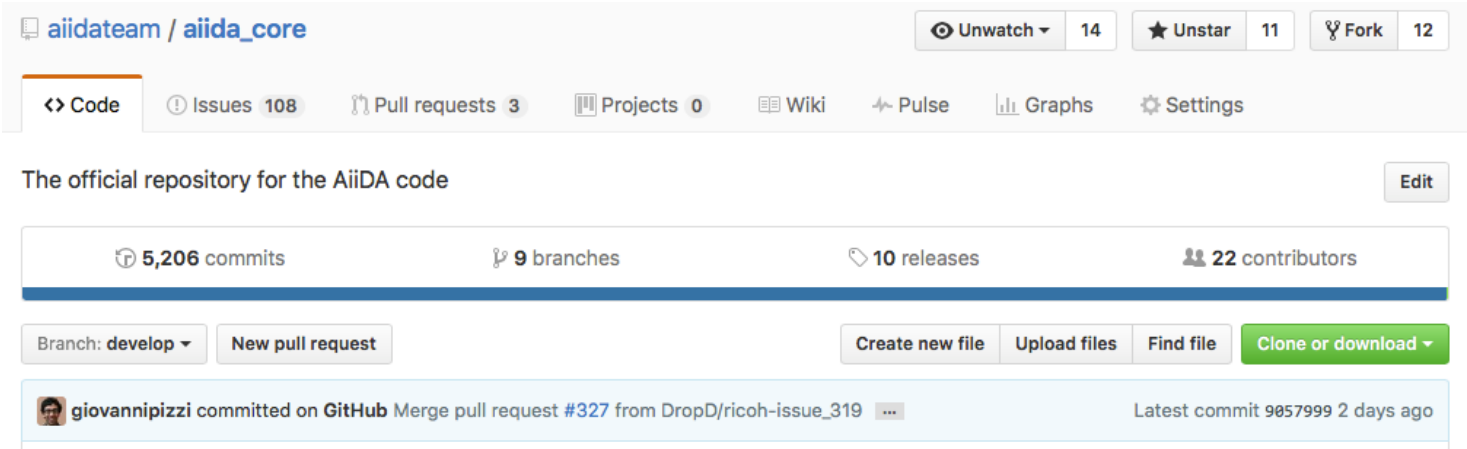
2012 *2013* *2014* *2015* *2016*

Mar 2012: First discussions about current framework

May 2014: Distributing beta versions to selected groups

Apr 2015: Paper on arXiv, AiiDA 0.4.1 released

Oct-Dec 2015: Paper published, AiiDA 0.5.0 released

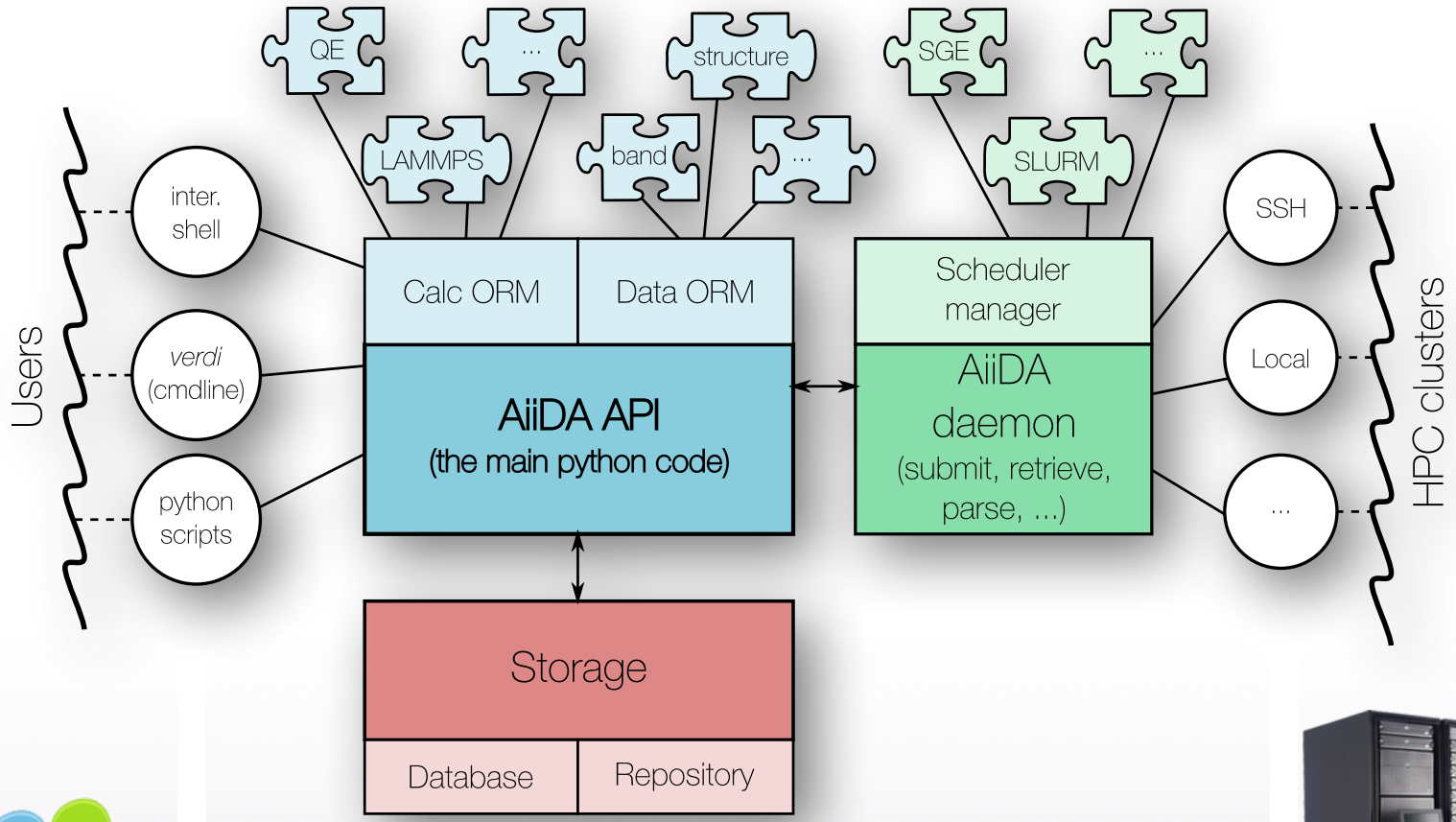


MIT License

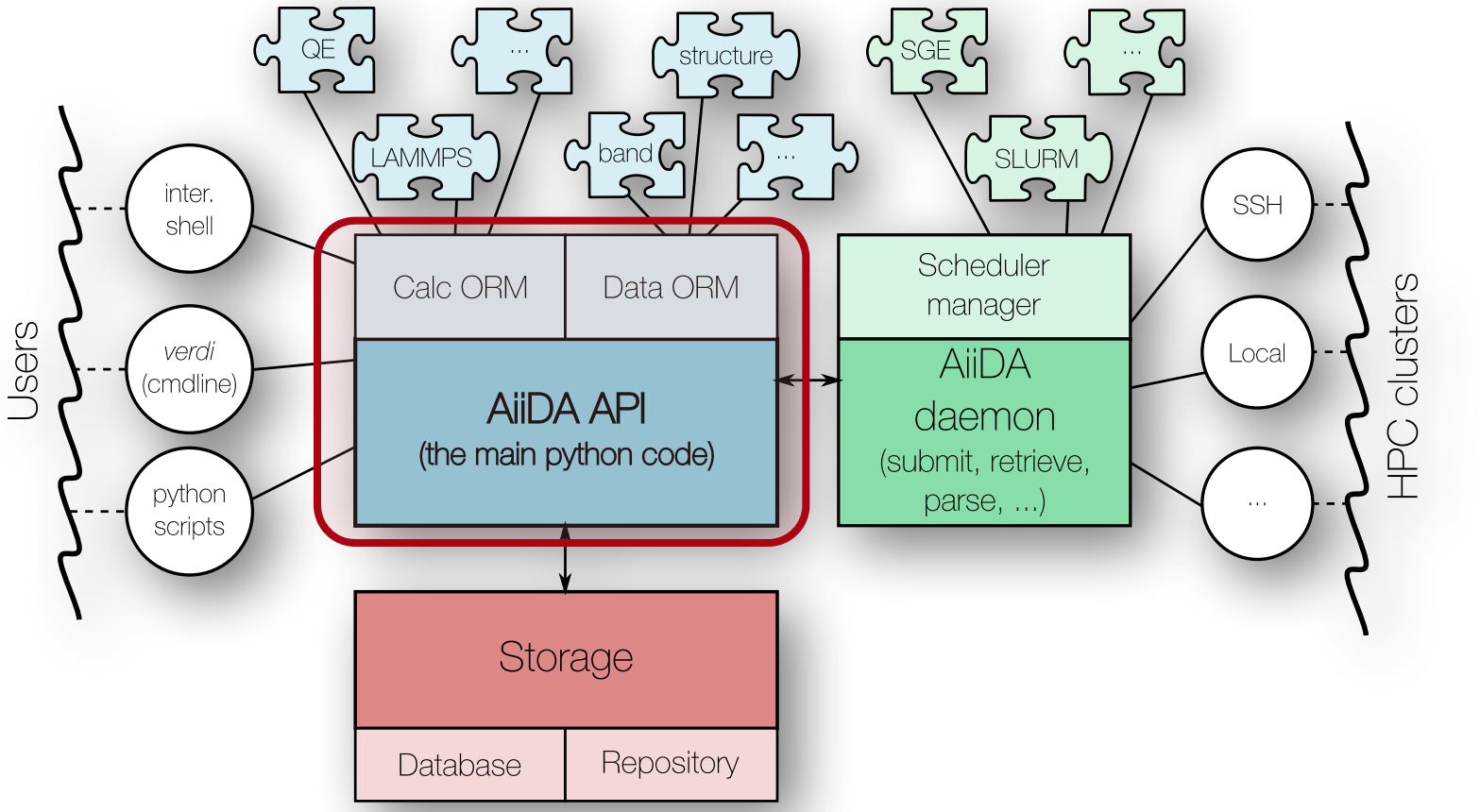
open source initiative

Approved License

What is AiiDA?

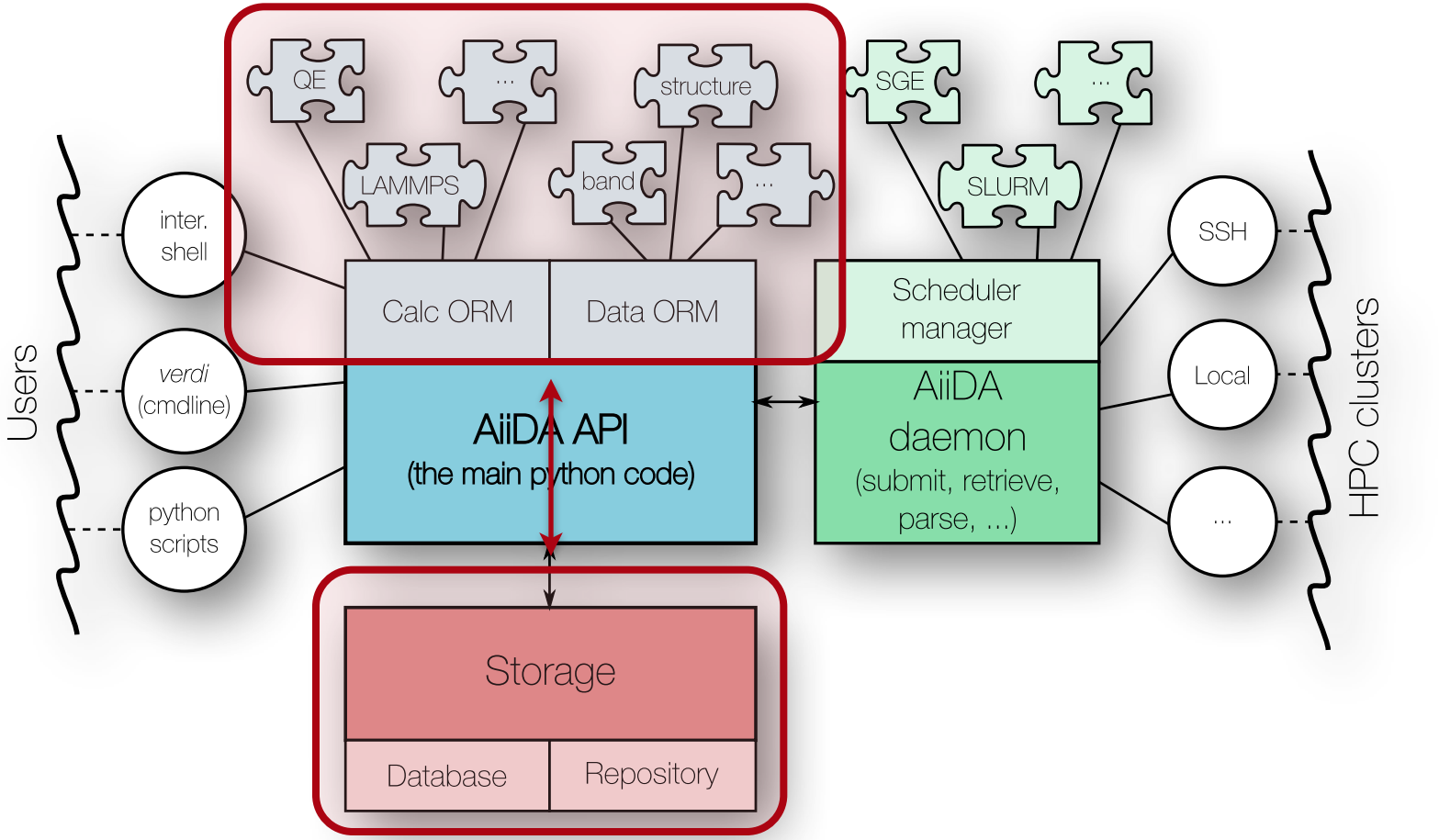


What is AiiDA?



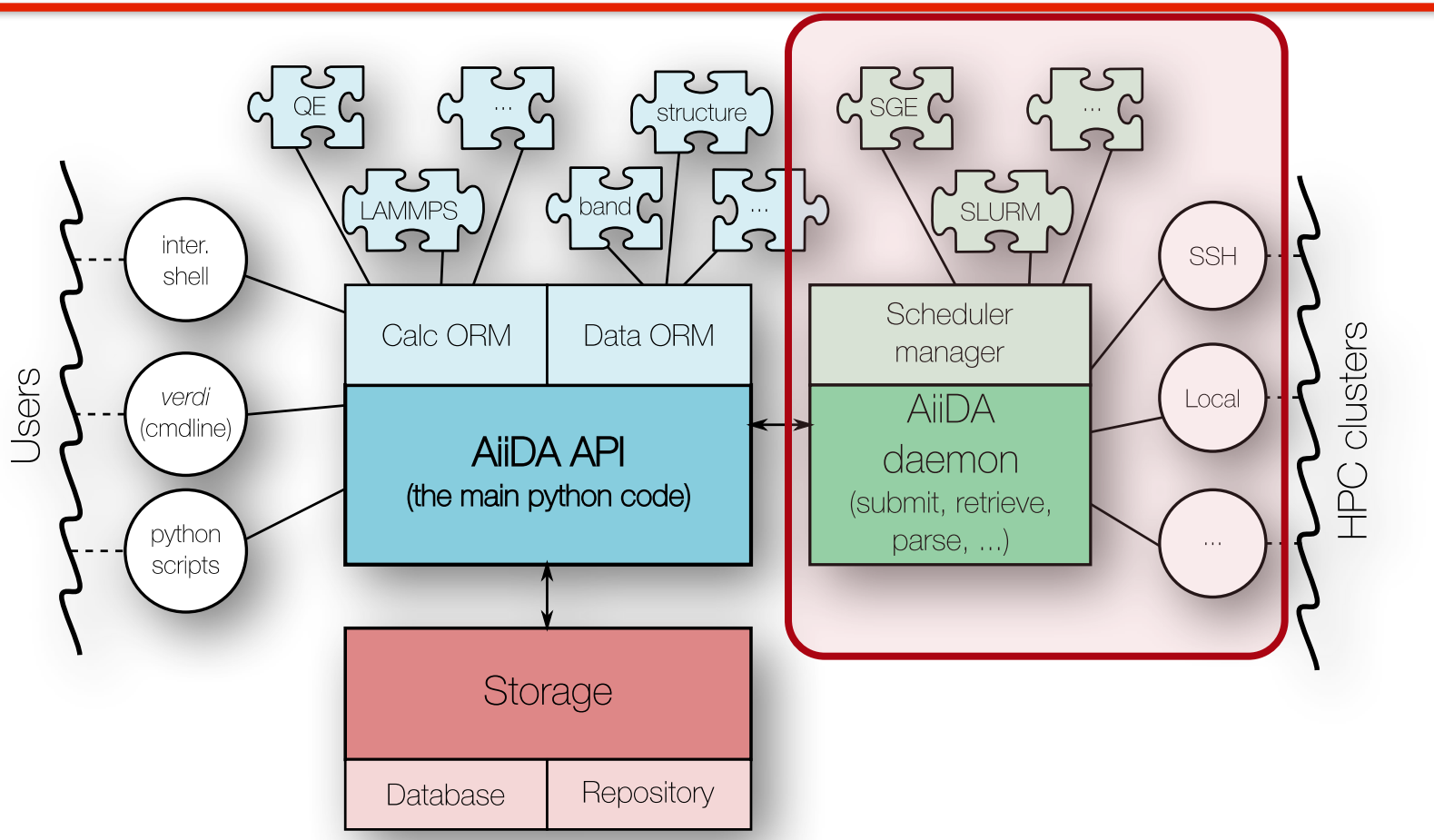
1. The core of the code is the **AiiDA API:** the main python code and classes

What is AiiDA?



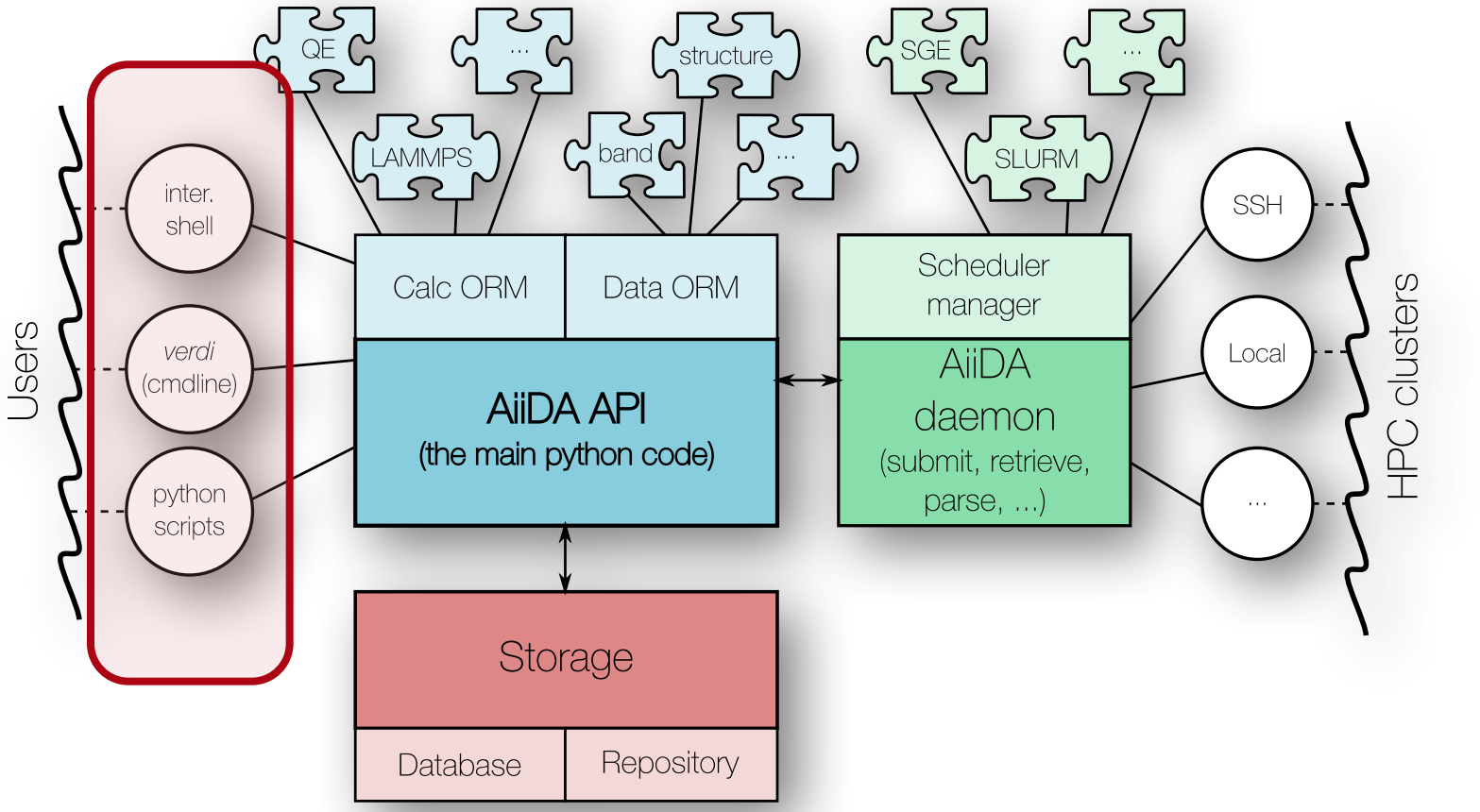
2. The **AiiDA Object-Relational Mapper** (ORM): transparently store data, codes and calculations in a database, *transparent to the user*

What is AiiDA?



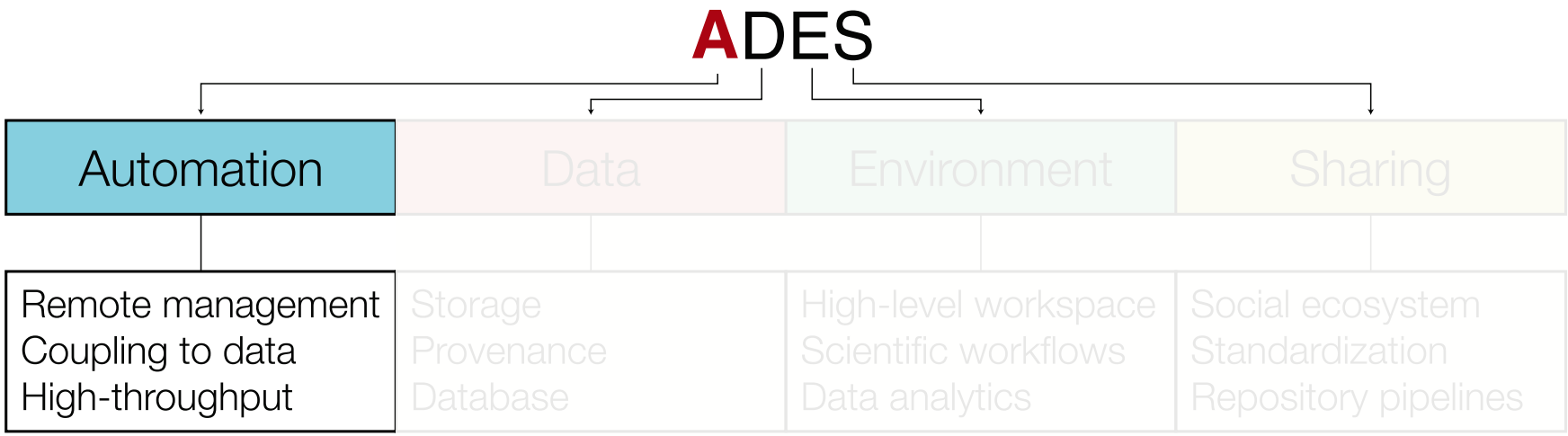
3. A **daemon** to manage interaction with remote computers without user intervention

What is AiiDA?



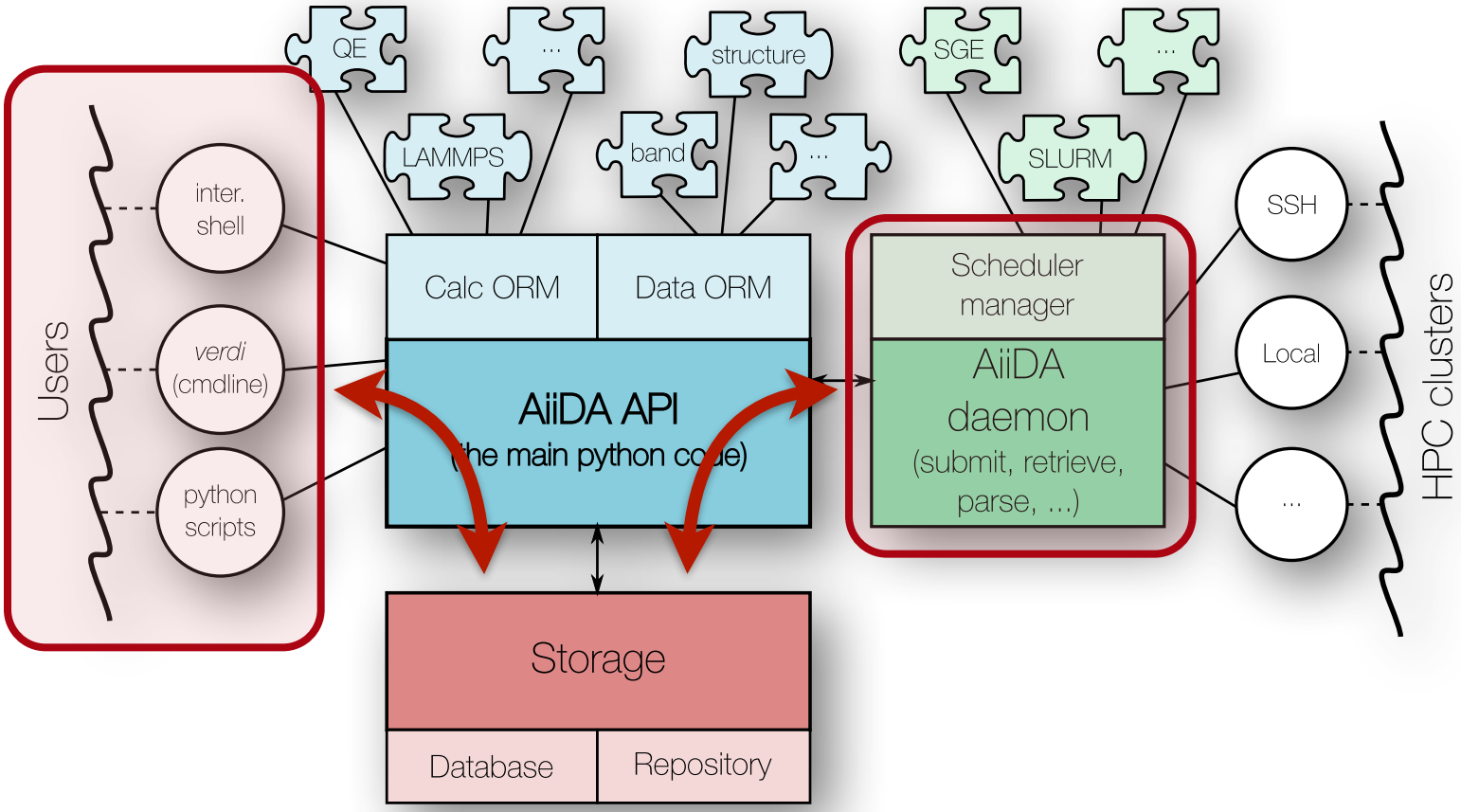
4. **User interaction** occurs via the command line tool **verdi**, the interactive shell or via Python scripts

Automation in AiiDA



G. Pizzi et al., Comp. Mat. Sci 111, 218-230 (2016)

Automation: coupling to data



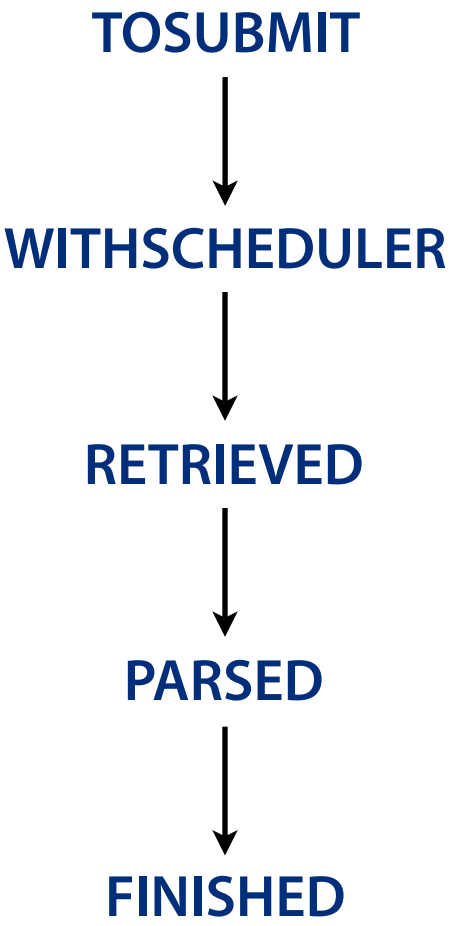
- ***Coupling* automation to data:**
  - *uniformity* of the input data, usage of codes and computers
  - full *reproducibility* of calculations (data is stored first)

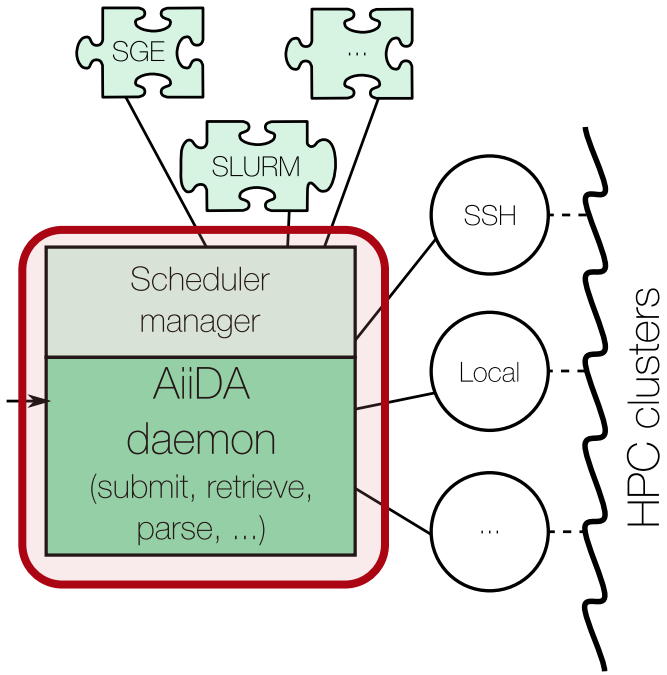
Automation: the daemon

A **daemon** runs in the background

Calculation state

- Process runs in the background
- Daemon processes managed using *celery* with a *django* backend, and *supervisor*
- Supports direct *connection* and various protocols: ssh, sftp, …
- *Schedulers* supported: SLURM, SGE, Torque, PBSPro, LSF, …





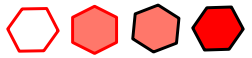
Abstraction into APIs: a single calculation

```
Parameter = DataFactory('parameter')
Structure = DataFactory('structure')

code = get_code('quantumespresso-pw@mycluster')
JobCalc = code.new_process()

attrs = {
    'max_wallclock_seconds': 3600,
    'resources': {"num_machines": 2},
    }

inp = {}
inp['structure'] = Structure(cif='silicon.cif')

inp['parameters'] = Parameter({
    'CONTROL': {
        'calculation': 'scf',
        'restart_mode': 'from_scratch',
    },
    'SYSTEM': {
        'ecutwfc': 40.,
    }})

f = async(JobCalc, _attributes=attrs, **inp)

print f.job.pk
```

Choose code and computer

Define all inputs

Take care of running the calculation **through the daemon**



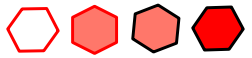
Abstraction into APIs: a single calculation

```
Parameter = DataFactory('parameter')
Structure = DataFactory('structure')

code = get_code('quantumespresso-pw@cluster2')
JobCalc = code.new_process()

attrs = {
    'max_wallclock_seconds': 3600,
    'resources': {"num_machines": 2},
    }

inp = {}
inp['structure'] = Structure(cif='silicon.cif')

inp['parameters'] = Parameter({
    'CONTROL': {
        'calculation': 'scf',
        'restart_mode': 'from_scratch',
    },
    'SYSTEM': {
        'ecutwfc': 40.,
    }})

f = submit(JobCalc, _attributes=attrs, **inp)

print f.job.pk
```

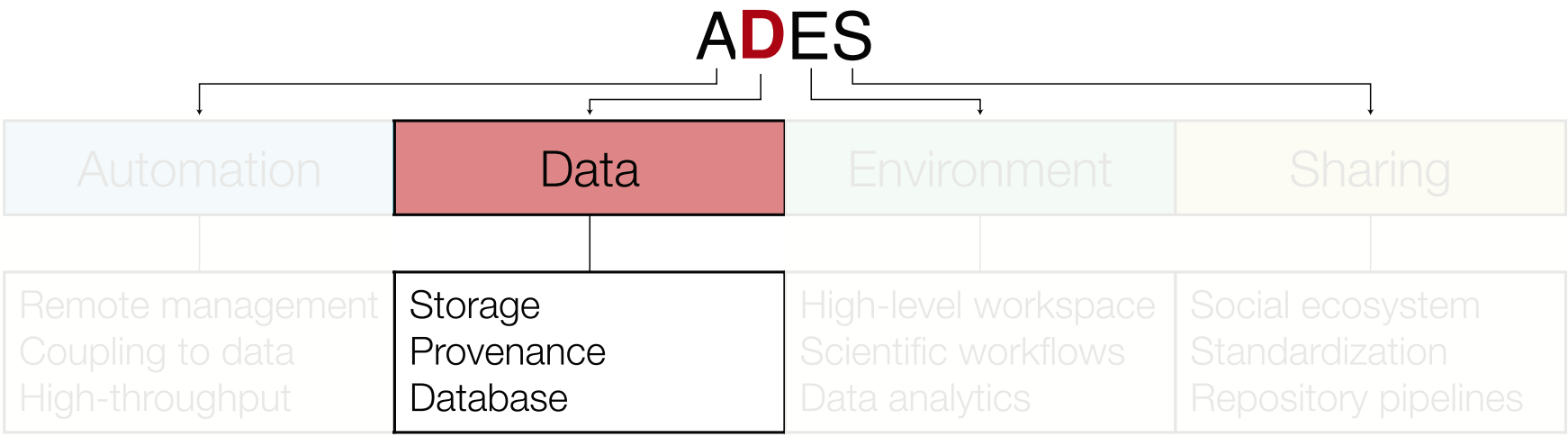
Just one line change to change computer (with different cluster, schedulers, version of codes, …)

Data gets stored in the DB during submission

Take care of running the calculation **through the daemon**



Data in AiiDA



G. Pizzi et al., Comp. Mat. Sci 111, 218-230 (2016)

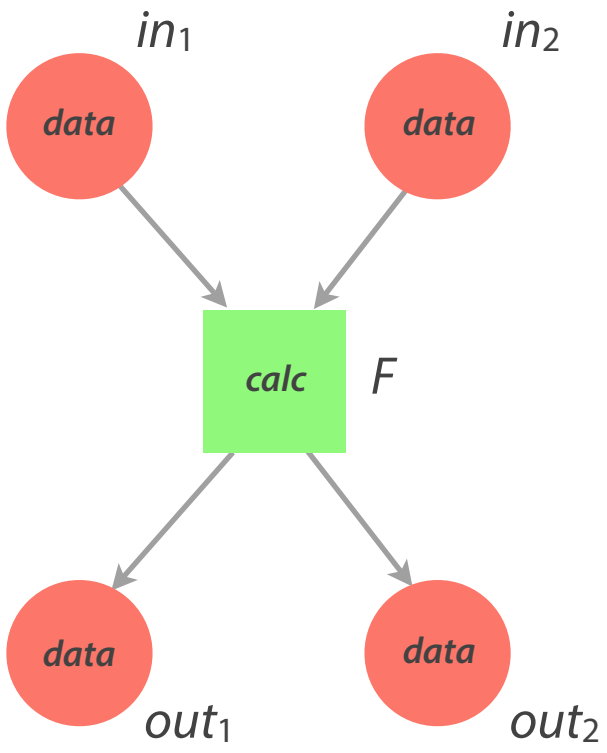
Storage and provenance

- *Calculated properties*: result of complex, connected calculations
- How do we store simulations **preserving the connected structure?**

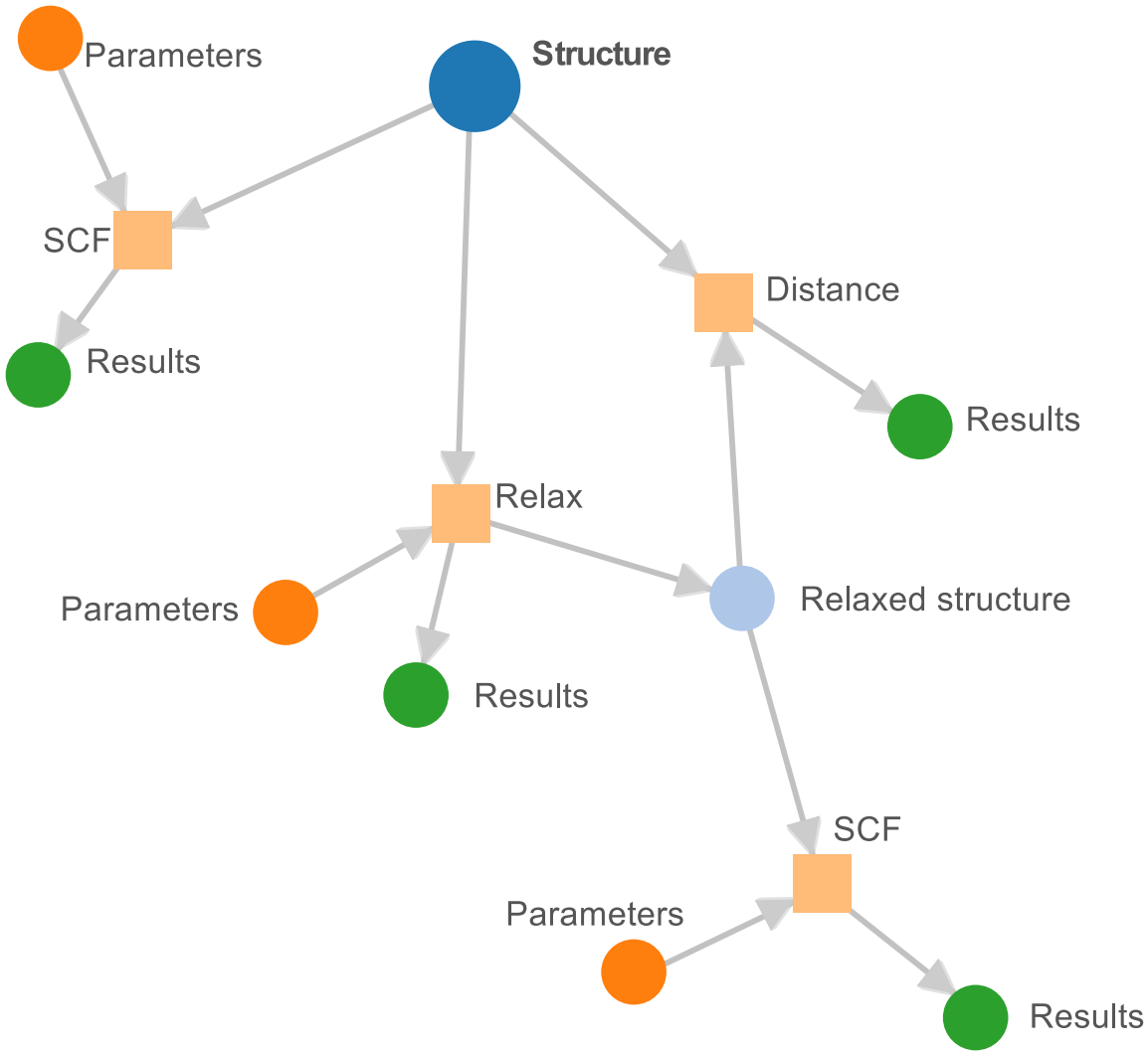
- Inspiration from the *open provenance model*
- **Any calculation: a function,** converting inputs to outputs:

$$out_1, out_2 = F(in_1, in_2)$$

- **Each object is a node in a graph,** connected by directional labeled links
- Output nodes can be used as inputs



Data provenance: Directed Acyclic Graphs



Data provenance: Directed Acyclic Graphs

Directed Acyclic Graphs:

appropriate representation

of calculations, data

and their **provenance**

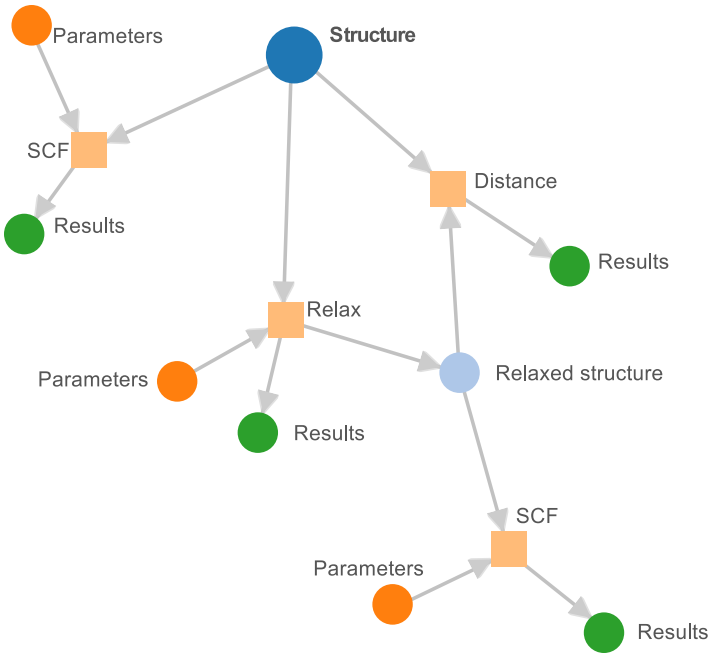
Next questions

What must we store?

How can we store it?

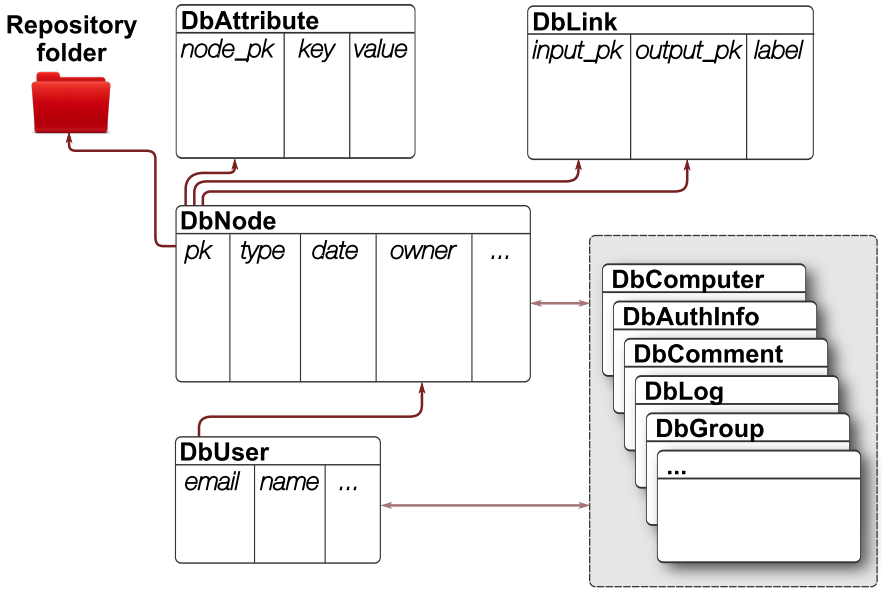
How can we query it?

Saving the DAGs: Nodes and Links



How to represent it… in SQL?

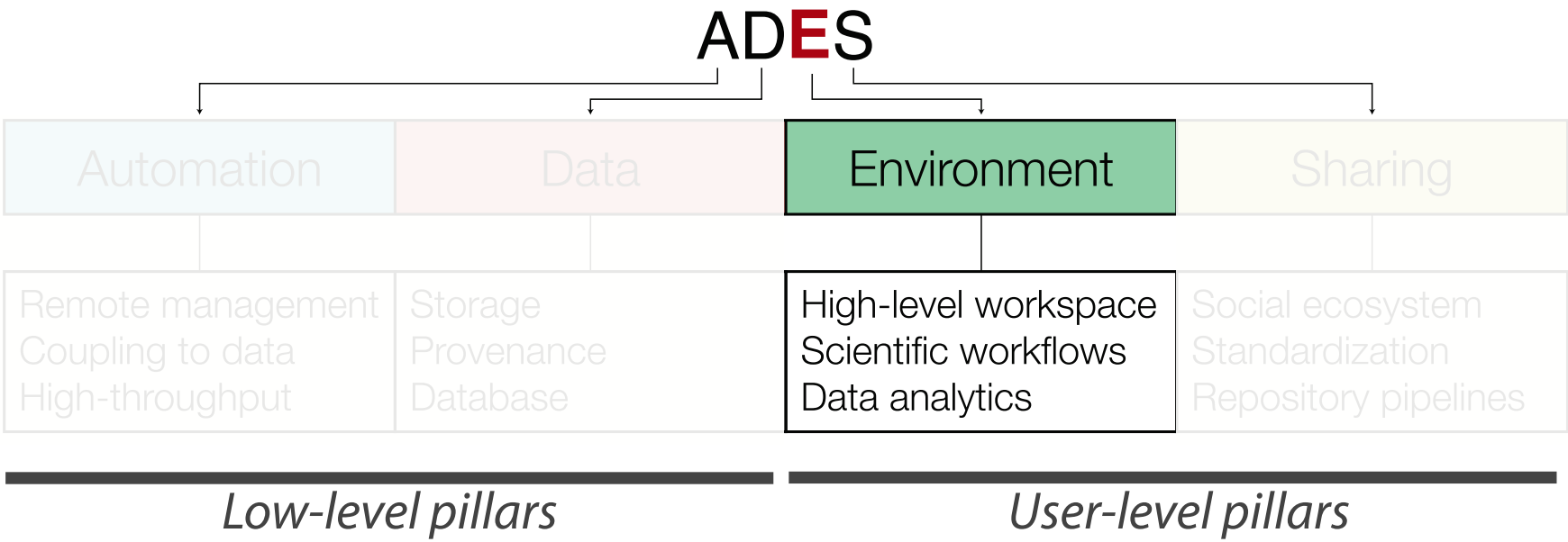
- *Each **node***: row in a SQL table
  - Additional data:
    - key-value **attributes**
    - **Files/folders**
- **Links** also stored in a SQL table
  ⇒ *jobs provenance*



- Multiple backends supported:
  - SQL (MySQL, PostgreSQL, …) and *attributes EAV table via Django*
  - SQL+JSONB (PostgreSQL > 9.4) via *SQLAlchemy*
  - Easy to extend to other backends (preliminary benchmarks with g*raph DBs* like Neo4j and TitanDB)

G. Pizzi et al., Comp. Mat. Sci 111, 218-230 (2016)

Environment in AiiDA



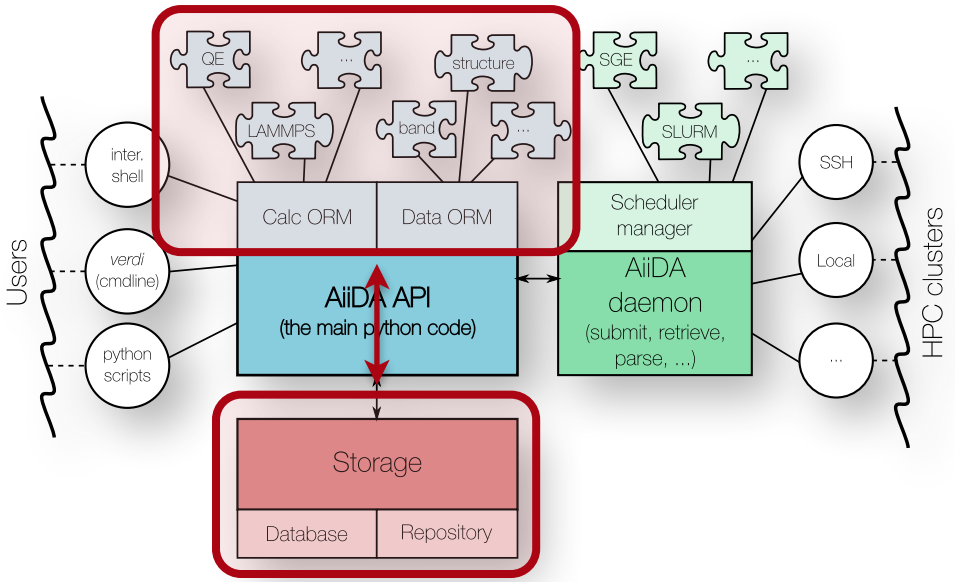
Is AiiDA easy to use?

Does it provide an advantage over files and custom-made scripts?

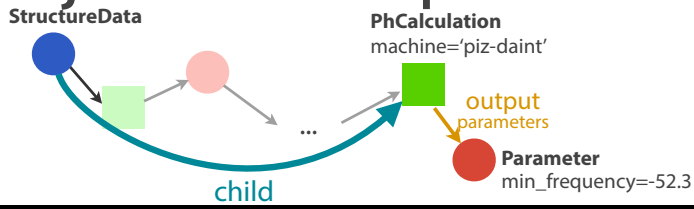
G. Pizzi et al., Comp. Mat. Sci 111, 218-230 (2016)

Environment in AiiDA: user interaction

High level Python interface

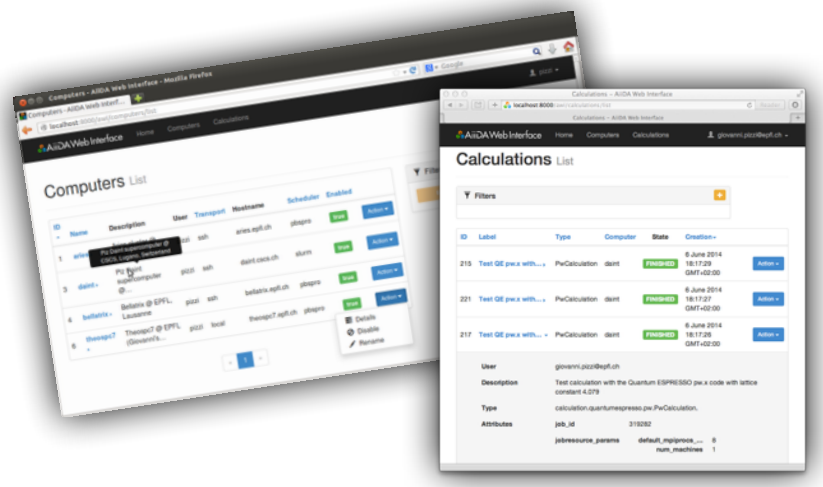


DB-agnostic query interface for AiiDA objects and their provenance




```
results = QueryBuilder(**{
  'path':[
    {'cls':PhCalculation, 'tag': 'ph'},
    {'cls':ParameterData, 'tag': 'param'}],
  'filters':{
    'ph': ['machine': "piz-daint"},
    'param': ['attributes.min_freq': {'<':-50.0}]},
  'project':{
   'param': ['attributes.min_freq'],}
}).iterall()
```

Seamless user interaction



Extensive documentation + tutorials

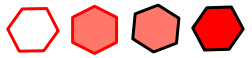
http://aiida-core.readthedocs.org/

Reusability and modularity: AiiDA plugins






Calculation	Data	Parser	Transport	Scheduler	Importers & exporters
Generation of input files for a given code	Management of data objects for input/ output	Parsing of code output & generation of new DB nodes	How to connect to a cluster	How to interact with the scheduler	Import structures, ... from external DBs
Quantum ESPRESSO, Phonopy, ASE, GPAW, Yambo, NWChem, …	*files&folders, parameter sets, remote data, structures, pseudos, ...*	*Quantum ESPRESSO, Phonopy, ASE, GPAW, Yambo, NWChem, ...*	*local connection, ssh, ...*	*PBSPro, Torque, SGE, SLURM, LSF, …*	*ICSD, COD, TCOD, MPOD, ...*

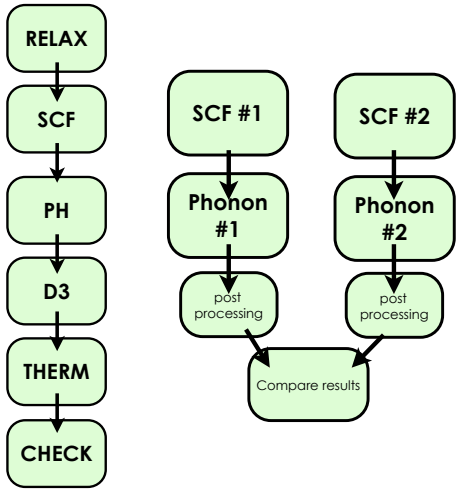


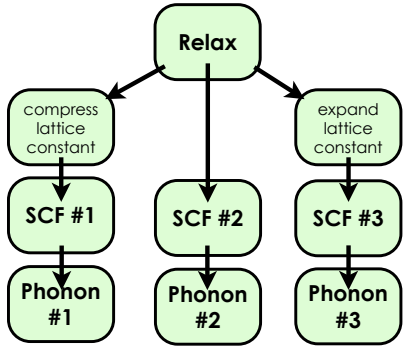
Environment in AiiDA: Scientific workflows

- Computed properties: result of complex **sequences of calculations**

- ***Not only storage***: also the management and encoding is important for reproducibility

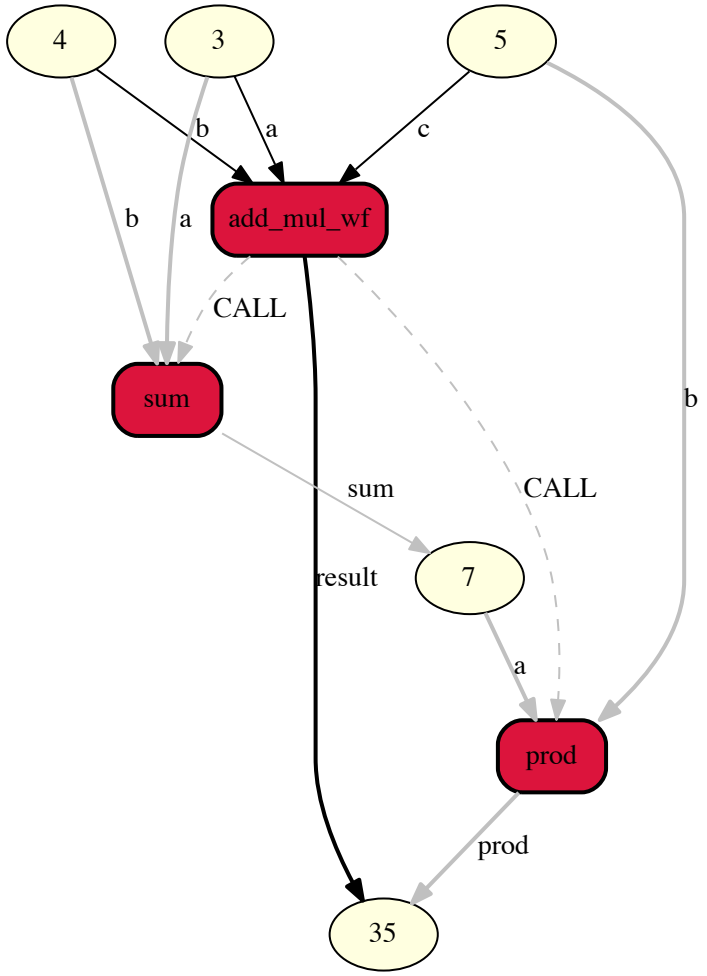
- We need to **encode** and to **share** "turn-key solutions" for:
  - the **calculation** of materials properties
  - the **automatic validation** of results





Workflows - a ‘Hello World’ example

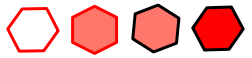
```
@wf
def sum(a, b):
    return {'sum': a+b}


@wf
def mul(a, b):
    return {'prod': a*b}


@wf
def add_mul_wf(a, b, c):
    return {'result':
        mul(add(a, b)[‘sum’], c)[‘prod’]}


final_value = add_mul_wf(
    a=Int(3),
    b=Int(4),
    c=Int(5))[‘result’]
```



final_value = (3+4) * 5 = 35

Workflows features

- Automatic tracking of provenance in the DB from *simple python functions*
  Inputs, outputs, call links, stored by simply adding a decorator
- Both serial and parallel (multithreaded) execution
  (using *async* calls and waiting for the result of *futures*) and combination of parallel and serial execution
- Checkpointing of workflows
  (you can shutdown your machine while it's waiting for a day-long job, and then continue from where you left) using a custom Workflow class
- Nesting and reuse of existing workflows
- Description of the inputs and outputs of a workflow
  for perspective provenance



A real-life workflow example: phonon dispersions

Main-Workflow

Sub-workflows

Structure Relaxation

Dynamical matrices

Single calculations

Interatomic force constants

Phonon dispersion

Relaxation #1

Relaxation #2

Relaxation #n

Structure cell converged

Restart management

PW vc-relax

Restart (wall-time exceeded, …)

PW vc-relax

PW vc-relax

several failure cases handled automatically

Initialize PH

PH on q-grid

Collect phonons

Parallelization

PH on $\mathbf{q}_1$

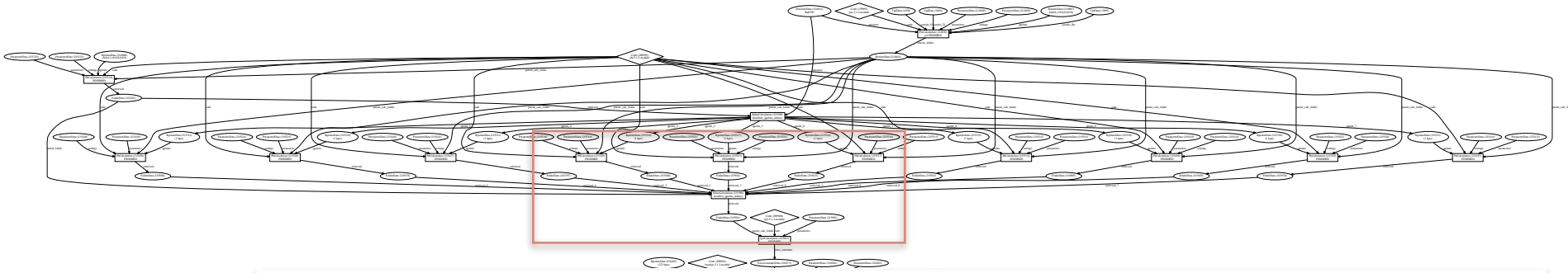
PH on $\mathbf{q}_2$

PH on $\mathbf{q}_n$

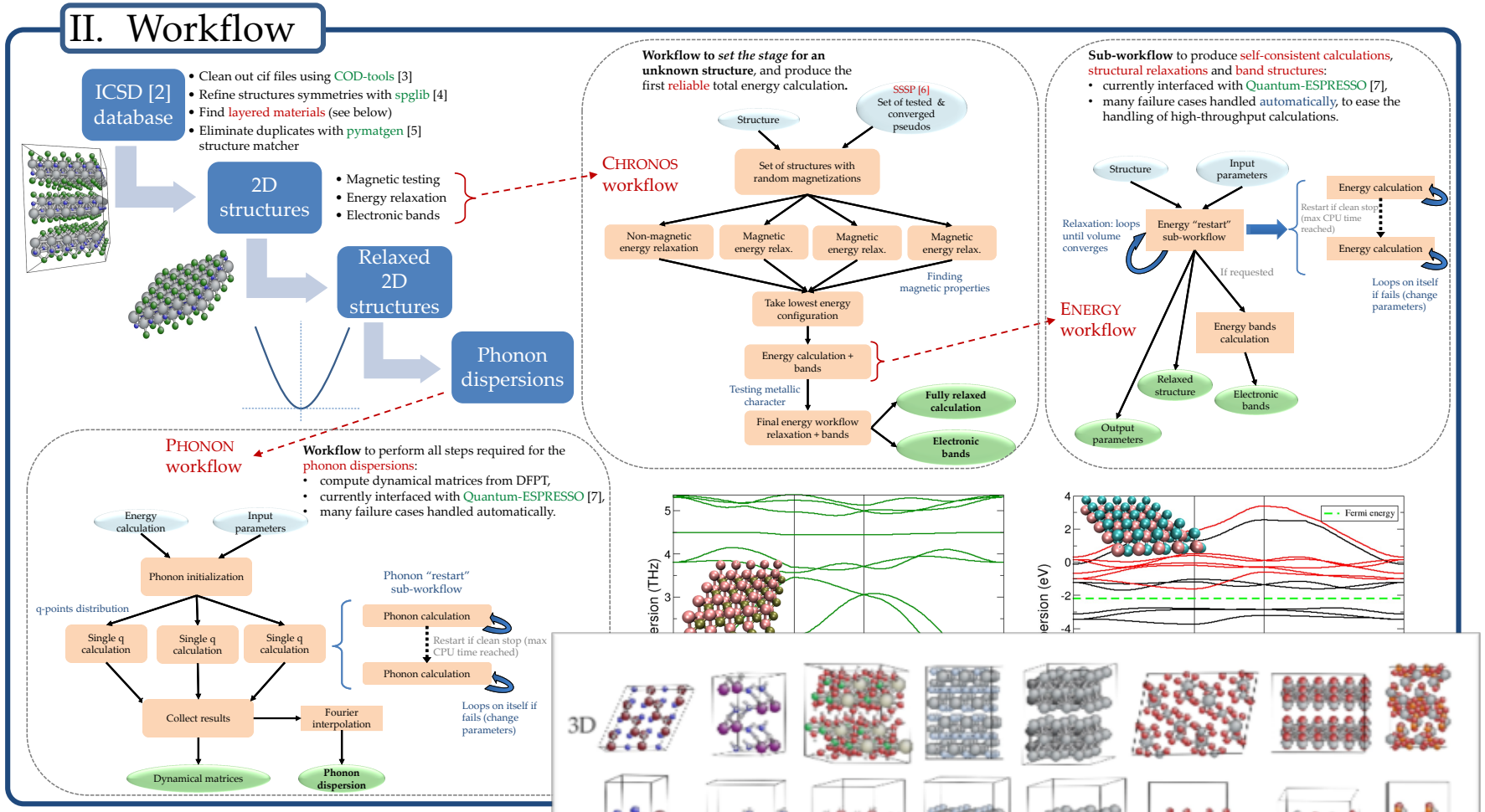
N. Mounet et al.

Visualising the outcome

```
params = {'input': {'kpoints_density': 0.2,
                    'convergence': 'tight'},
          'structure': structure,
          'pseudo_family': pseudo_family,
          'machinename': 'mycluster',
          'pw_input':{'volume_conv_threshold': 5e-2},
                      'pw_parameters': {
                         'SYSTEM': {'ecutwfc': 30.},
                         'ELECTRONS': {'conv_thr': 1.e-10}}
                    'ph_input':{'distance_kpoints_in_dispersion': 0.005,
                         'diagonalization': 'cg'}
                     }
wf = asyncd(PhBandsWorkflow, **params)
```



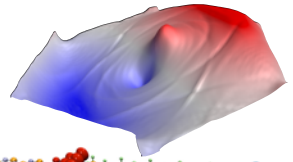
Discovering new 2D materials



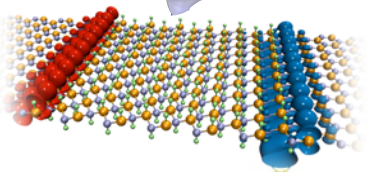
N. Mounet et al., *arXiv:1611.05234*

Some current applications

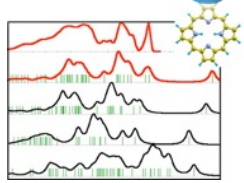
Phonon-phonon scattering in 2D — *Phonon hydrodynamics in 2D materials*



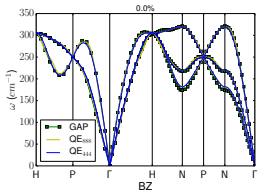
1D metallic wires at interfaces — *Engineering polar discontinuities in 2D*



Functional development — *Development of a Koopmans' compliant functional*

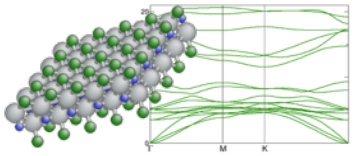


Neural Network potentials — *Generating databases for neural network potentials*

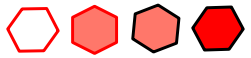


Pseudopotential database — *Creation of a Standard Solid State Pseudopotentials library*

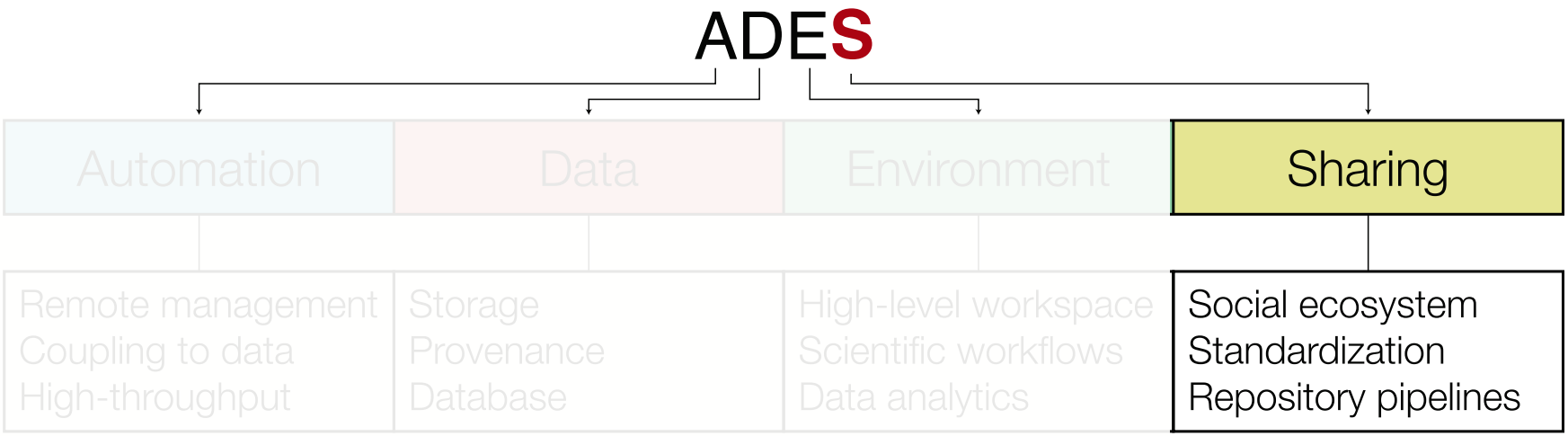
Thermodynamical properties of 2D materials — *Discovering 2D materials and creating a database of their thermodynamical properties*



Paraelectric-ferroelectric transition in perovskites, optical properties with GW, stability of structures, …



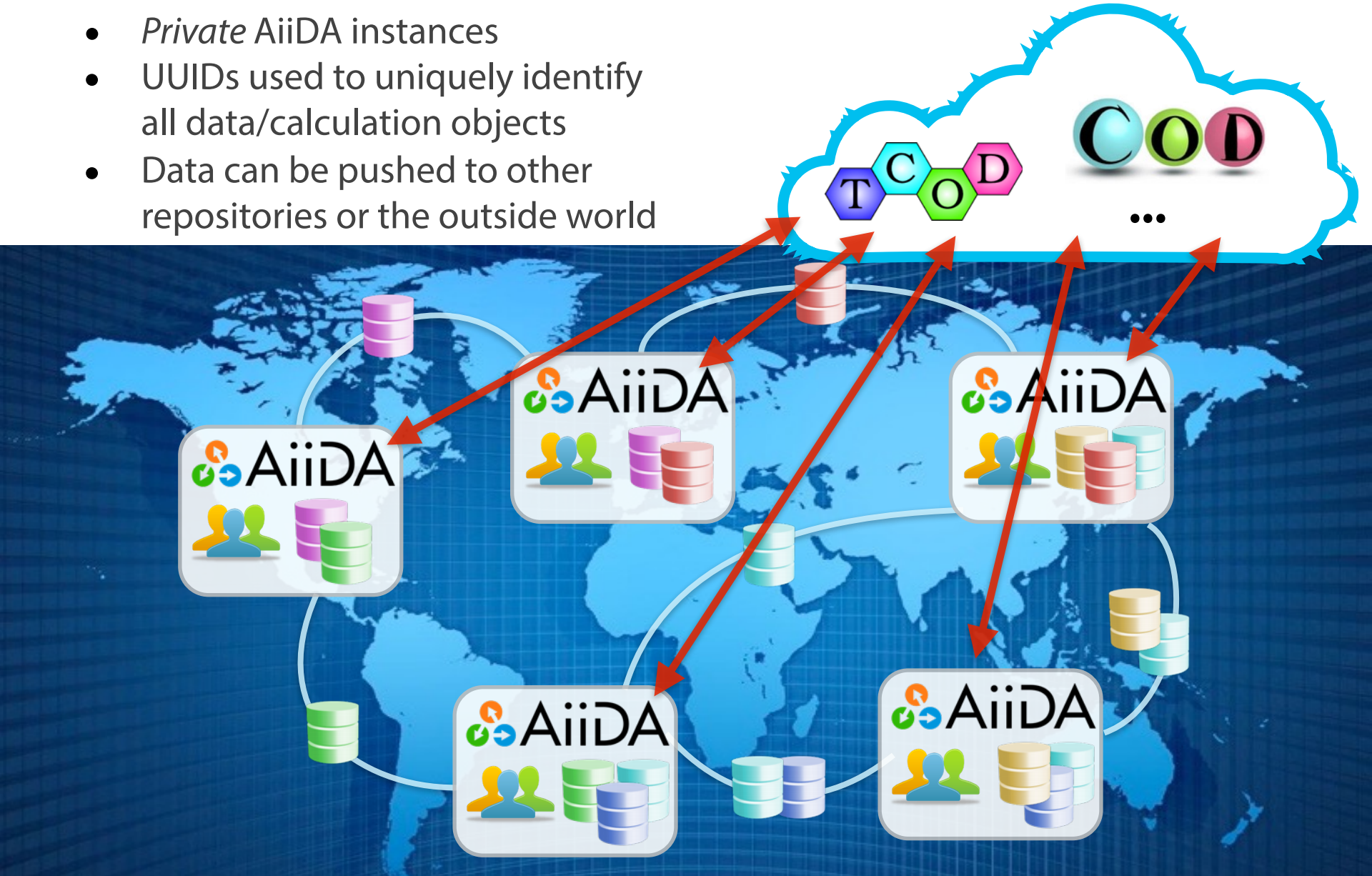
Sharing in AiiDA



G. Pizzi et al., Comp. Mat. Sci 111, 218-230 (2016)

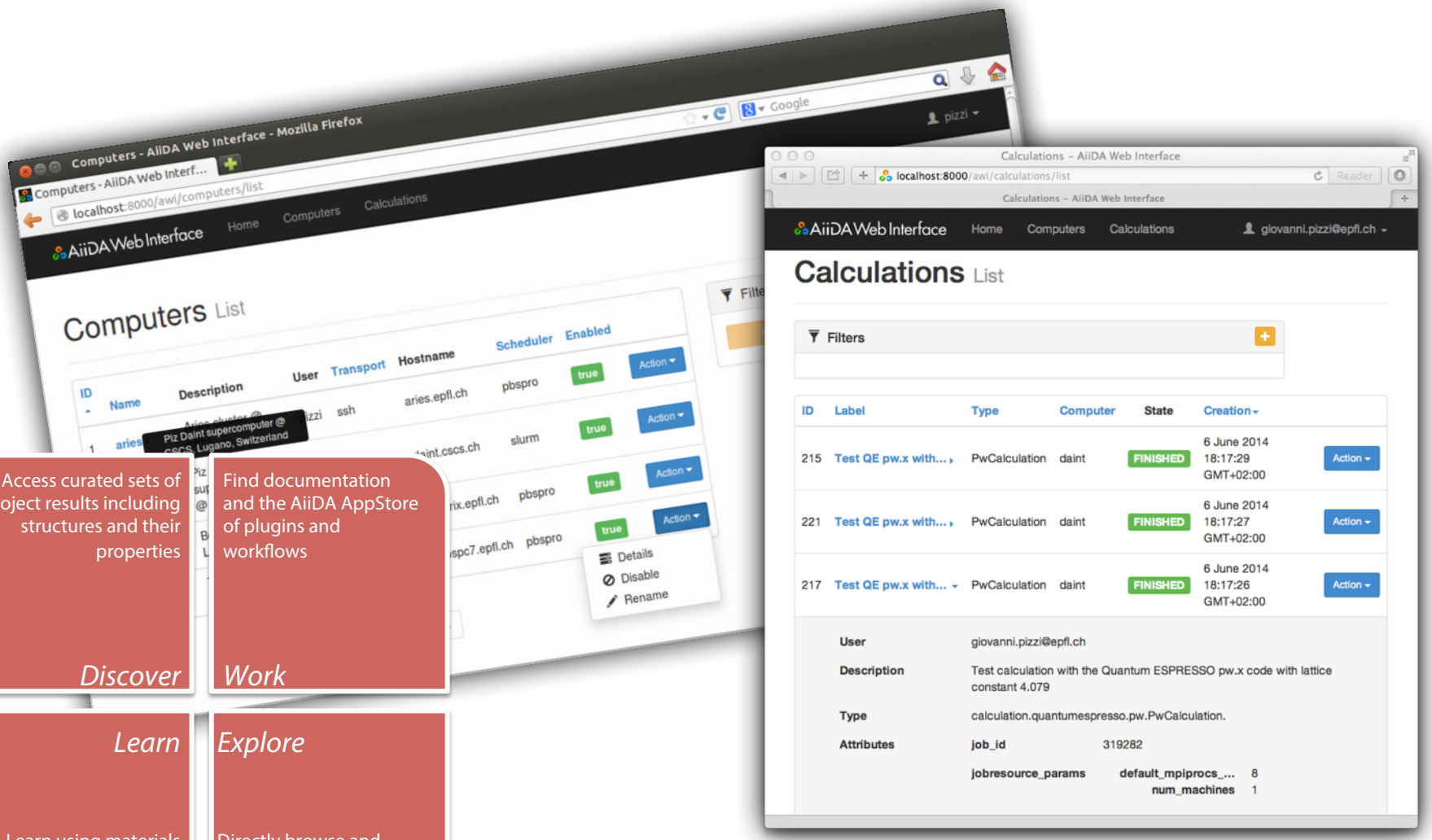
Sharing in AiiDA

- *Private* AiiDA instances
- UUIDs used to uniquely identify all data/calculation objects
- Data can be pushed to other repositories or the outside world



Materials Cloud

www.materialscloud.org



Access curated sets of project results including structures and their properties

Discover

Find documentation and the AiiDA AppStore of plugins and workflows

Work

Learn

Learn using materials including lectures and video tutorials related to materials science

Explore

Directly browse and visualise the AiiDA database

App-store model

"App-store"-like model for plugins & workflows, e.g.

- **Computers**: automatically setup a new cluster or supercomputer
- **DB importers**: load structures and data from COD, ICSD, …
- **Calculations**: plugin for your simulation software
- **Turn-key solutions**: workflows to compute a desired property
- …

The MaterialsCloud AiiDA app-store

Simulation codes

Quantum ESPRESSO

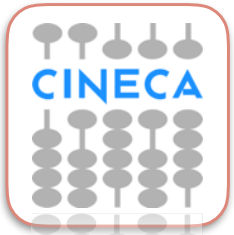
Yambo

CP2K

Fleur

Computing clusters

CSCS (CH)

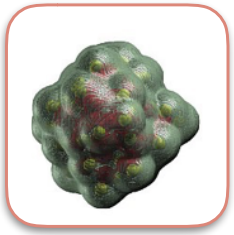


CINECA (IT)

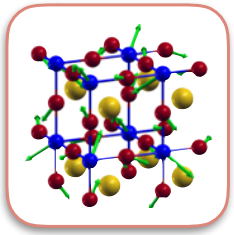
BSC (ES)

Jülich (DE)

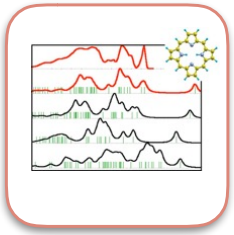
Turn-key workflows



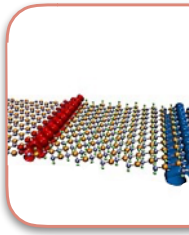
Electronic density



Phonons



Optical properties



Transport

Acknowledgements

The AiiDA team

Giovanni Pizzi (EPFL)

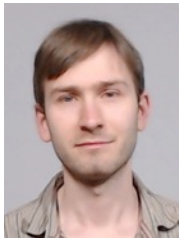
Andrea Cepellotti (EPFL)

Nicolas Mounet (EPFL)

Fernando Gargiulo (EPFL)

Boris Kozinsky (BOSCH)

Nicola Marzari (EPFL)



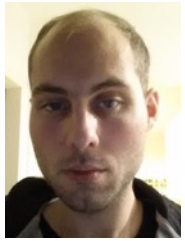
Martin Uhrin (EPFL)

Leonid Kahle (EPFL)

Spyros Zoupanos (EPFL)

Snehal Waychal (EPFL)

Sebastiaan Huber (EPFL)



Rico Häuselmann (EPFL)

Andrius Merkys (Vilnius)

Former AiiDA team members — Riccardo Sabatini, EPFL — **Plugin contributors** — **Quantum ESPRESSO NEB**: Marco Gibertini (EPFL); **Quantum ESPRESSO DOS, PDOS; Wannier90**: Daniel Marchand (EPFL); **CP2K**: Aliaksandr Yakutovich (EMPA), Uli Schauer (ETHZ), Tiziano Müller, Andreas Gloess, Patrick Seewald (UZH); **FLEUR**: Jens Broeder, Gregor Michalicek, Daniel Wortmann (Jülich); **Exciting**: Anton Kozhevnikov (CSCS); **YAMBO**: Andrea Ferretti, Mike Atambo, Daniele Varsano (CNR-NANO), Gianluca Prandini, Antimo Marrazzo (EPFL); **SIESTA**: Victor Garcia Suarez (Uni Oviedo); **VASP**: Mario Zic (Trinity College Dublin);, … **Contributors** — Prof. Christoph Koch (EPFL); Valentin Bersier, Philippe Schwaller, Ivano Castelli (THEOS EPFL); Marco Dorigo (ICAMS - Bochum); Eric Hontz (MIT & Bosch RTC)

Infrastructure Projects

- **MARVEL NCCR (http://marvel-nccr.ch): Swiss National Centre of Competence on Computational Design and Discovery of Novel Materials; started May 2014, funded 2014-2026. 33 PIs from 11 Institutions.**

Home Project Research People News Events

MARVEL

Management Team

Executive Committee

Equal Opportunities

Welcome on board

The goal of the NCCR-MARVEL is to radically transform and accelerate invention and discovery in science and

Who's behind MARVEL?

Lewin Boehnke
Postdoc in group of Philipp Werner Read more

Recents News

Software development engineers
Up to three software development engineer positions will be available as of late summer / early fall 2015 at SISSA, Trieste.

Visualization contest winner
Andrius Merkys, a MARVEL PhD student at EPFL, has won the first prize of the ACCES (Application-Centered Computational Engineering Science) visualisation contest, in the image section.

Alessandro Curioni named director of IBM Research, Zurich

Infrastructure Projects

- **MARVEL NCCR (http://marvel-nccr.ch): Swiss National Centre of Competence on Computational Design and Discovery of Novel Materials; started May 2014, funded 2014-2026. 33 PIs from 11 Institutions.**

- **MaX: MAterials design at the eXascale (http://max-center.eu): EU H2020 e-infrastructure project, from Sep 2015 (Modena/Trieste/EPFL/Barcelona/ Julich) + 5 supercomputing centers (CINECA, CSCS, BCS, Julich, KTH)**

Home Project Research People News Events

MARVEL

Who's behind MARVEL?

Lewin Boehnke
Postdoc in group of Philipp Werner Read more

Recents News

Software development engineers
Up to three software development engineer positions will be available as of late summer / early fall 2015 at SISSA, Trieste.

Visualization contest winner
Andrius Merkys, a MARVEL PhD student at EPFL, has won

Conclusions

A computational science platform adopting the ADES model:

Automation

automate repetitive tasks via daemon, **abstracting** into APIs

Data

reproducibility&provenance, directed acyclic graphs, queries

Environment

flexible platform; **workflows** to encode scientists' knowledge

Sharing

social ecosystem to encourage interactions

Contacts and info:

Website: http://www.aiida.net

Docs: http://aiida-core.readthedocs.io

Git repo: https://github.com/aiidateam/aiida_core/


https://www.facebook.com/aiidateam


@aiidateam