Centro Singular de Investigación en **Tecnoloxías Intelixentes**

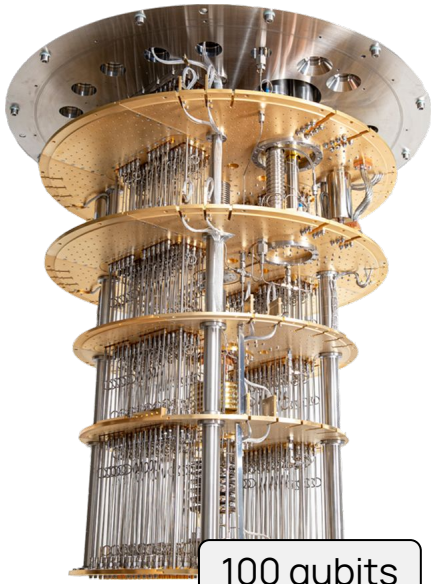
CESGA
GALICIA SUPERCOMPUTING CENTER

Portable Distributed Quantum Computing with NetQMPI

F. Javier Cardama (javier.cardama@usc.es)
Jorge Vázquez Pérez
Tomás F. Pena
Andrés Gómez

Quantum SPAIN
España | digital 2026
Plan de Recuperación, Transformación y Resiliencia
Gobierno de España · Ministerio de Ciencia, Innovación y Universidades
Financiado por la Unión Europea NextGenerationEU
Agencia Estatal de Investigación
RES Red Española de Supercomputación

1. **Introduction:** Why we need distributed quantum computation?

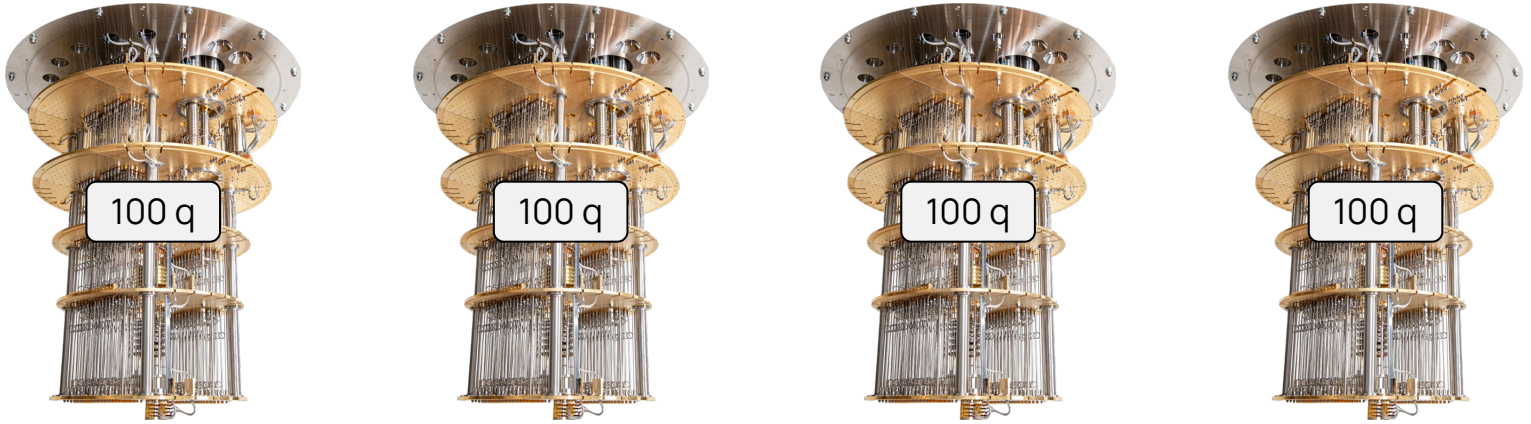


100 qubits

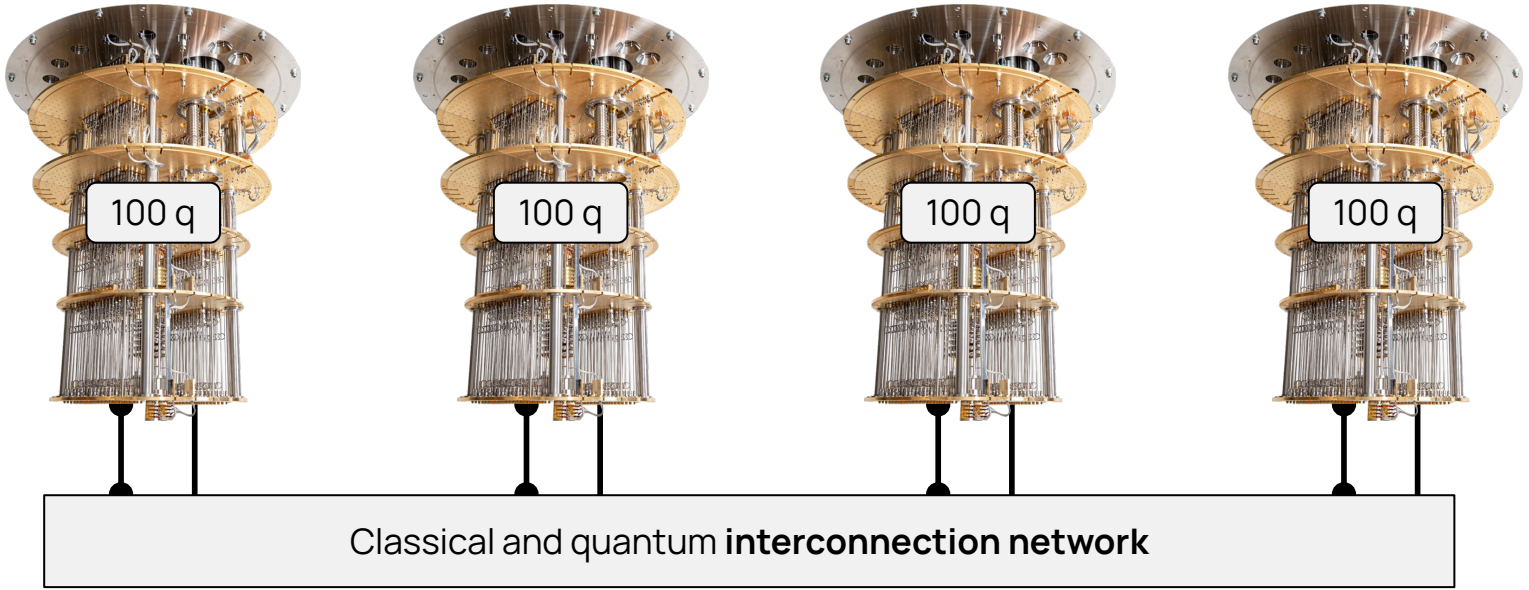
Limitations of use a single quantum computer:

- Noise of the current quantum technologies.
- Limited number of qubits per QPU (*Quantum Process Unit*).

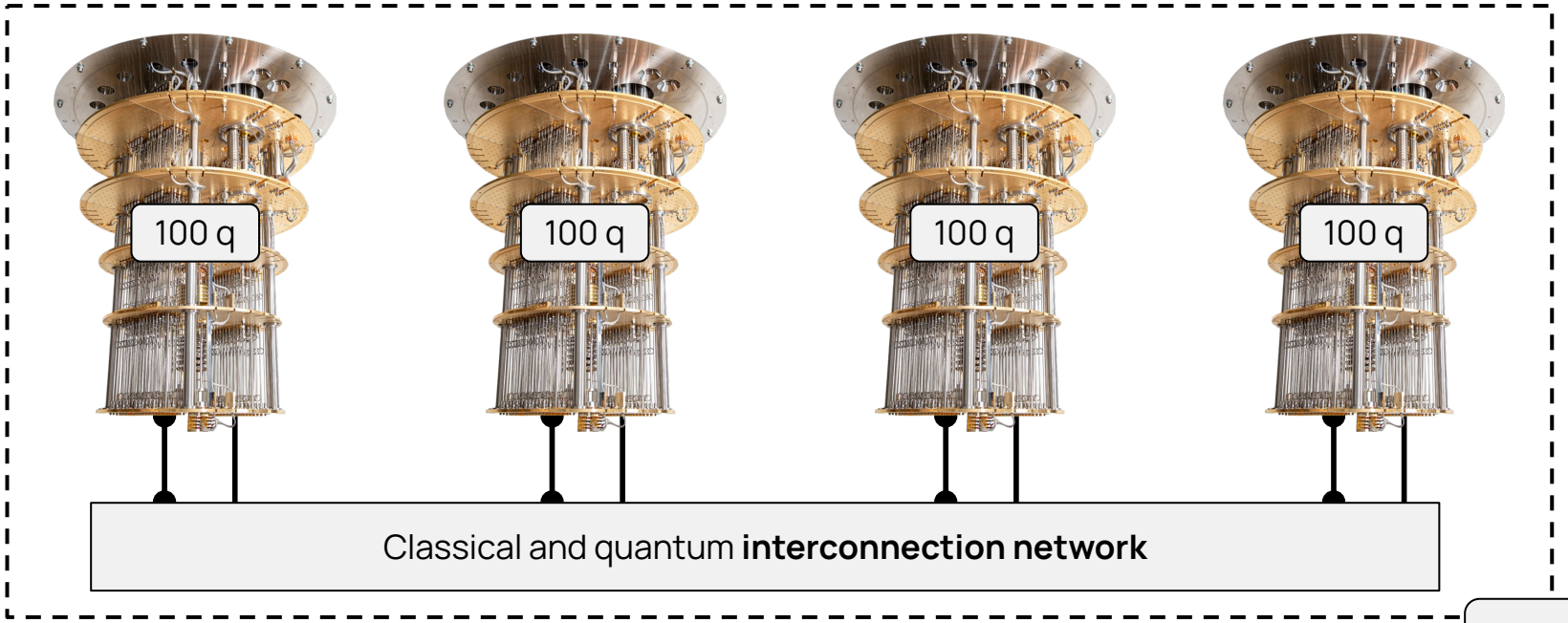
1. **Introduction:** Why we need distributed quantum computation?



1. **Introduction:** Why we need distributed quantum computation?



1. **Introduction:** Why we need distributed quantum computation?



1. **Introduction:** NetQMPI proposal

- There are a **lack of tools oriented to DQC** programming systems.

1. **Introduction:** NetQMPI proposal

- There are a **lack of tools oriented to DQC** programming systems.

- Abstract **low-level network concepts** as:
  - Quantum protocols communication.
  - Quantum sockets management.

```python
with sender:
    q = Qubit(sender)
    q.initialize(...)

    epr = epr_socket.create_keep()
    q.cnot(epr[0])
    q.H()
    m1 = q.measure()
    m2 = epr.measure()
m1, m2 = int(m1), int(m2)
socket.send_structured((m1, m2))
```

sender.py



```python
with receiver:
    epr = epr_socket.recv_keep()

    msg = socket.recv_structured()
    m1, m2 = msg.payload
    m2 and epr[1].X()
    m1 and epr[1].Z()

    receiver.flush()

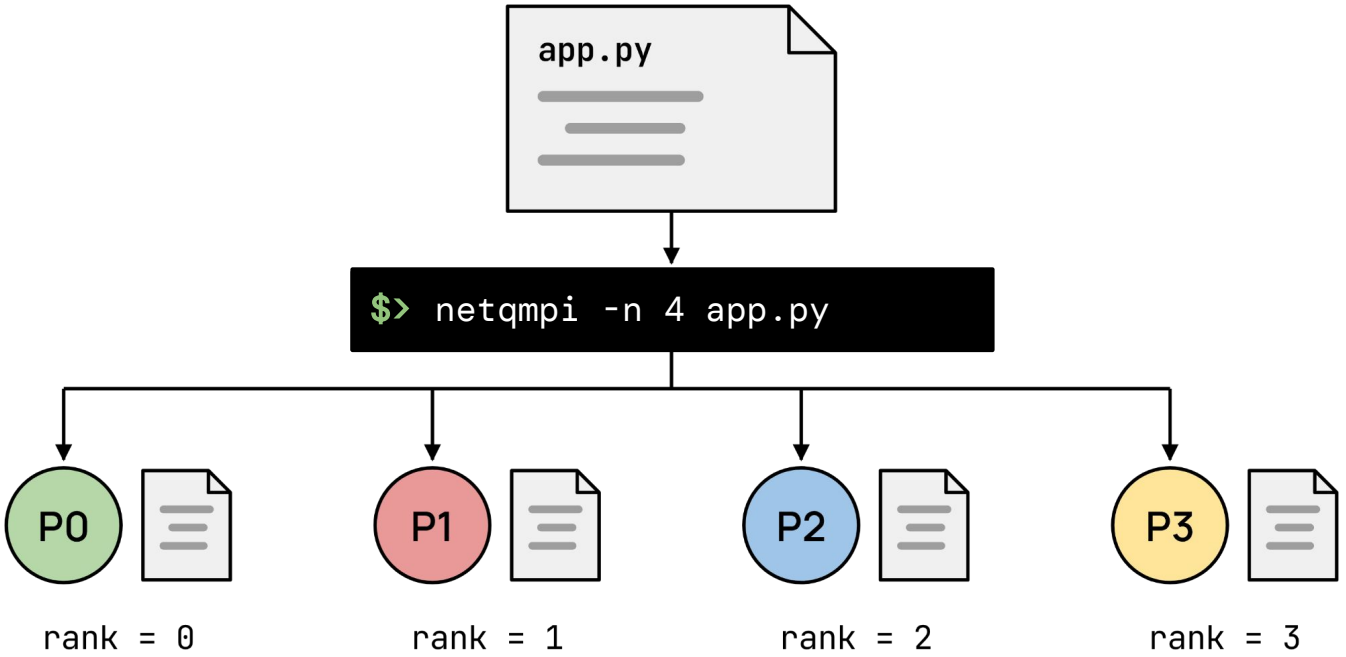
    epr[1].cnot(my_qubit)
```

receiver.py

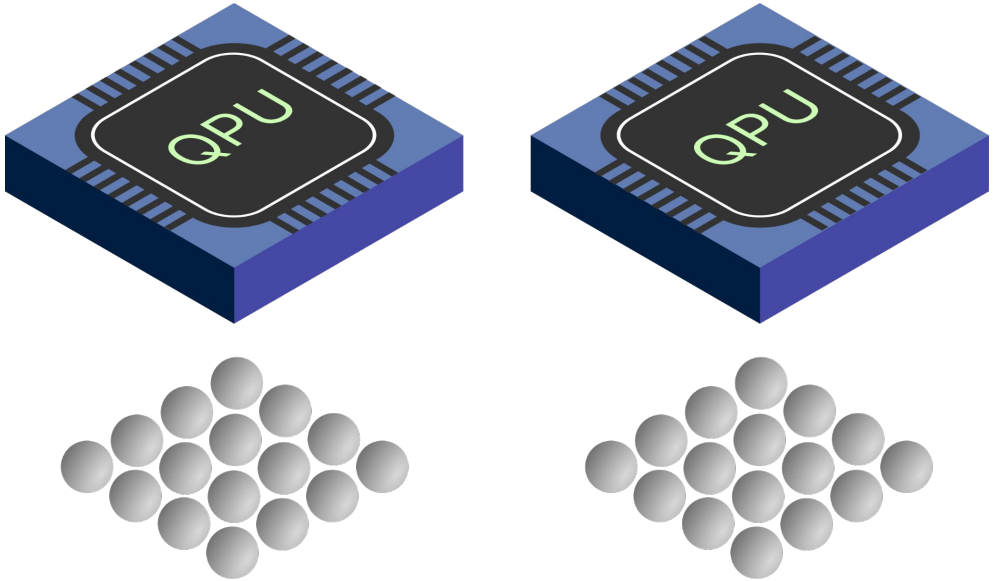


1. **Introduction:** NetQMPI proposal

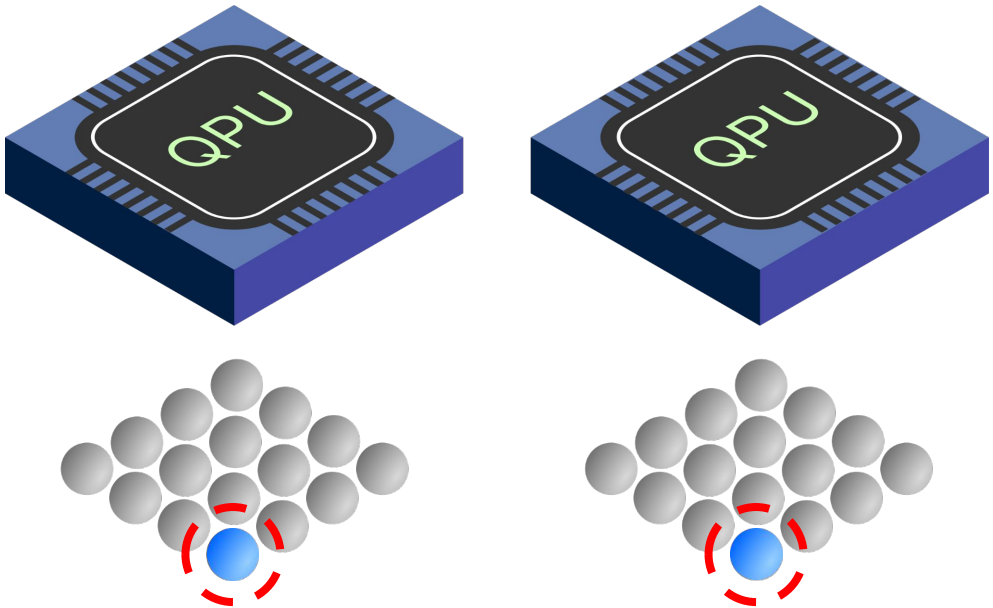
- Software tool for programming DQC systems in a **SPMD** (*Simple Program Multiple Data*) **form**.



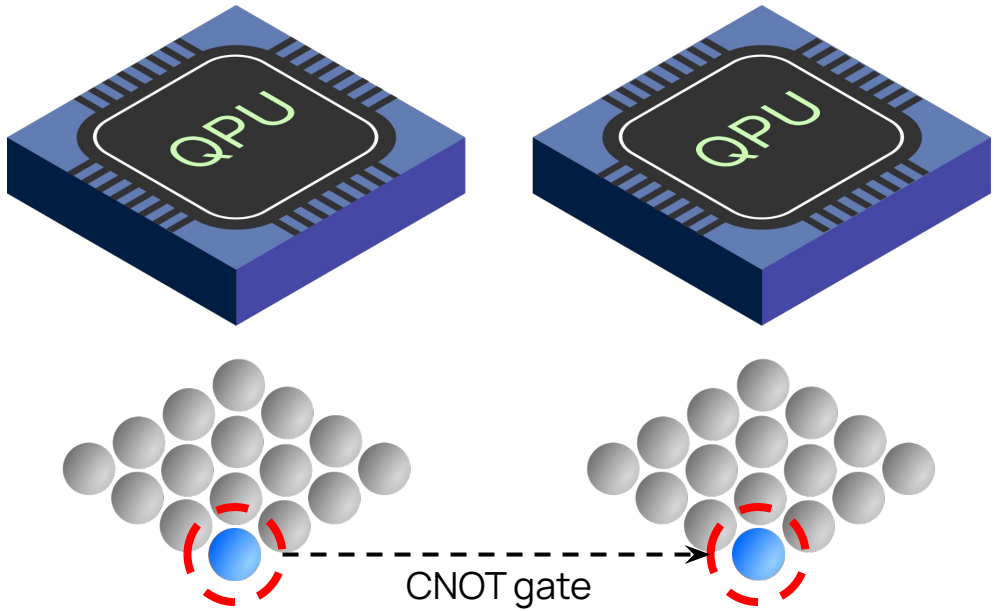
2. **Quantum communication protocols:** how distributed computation?



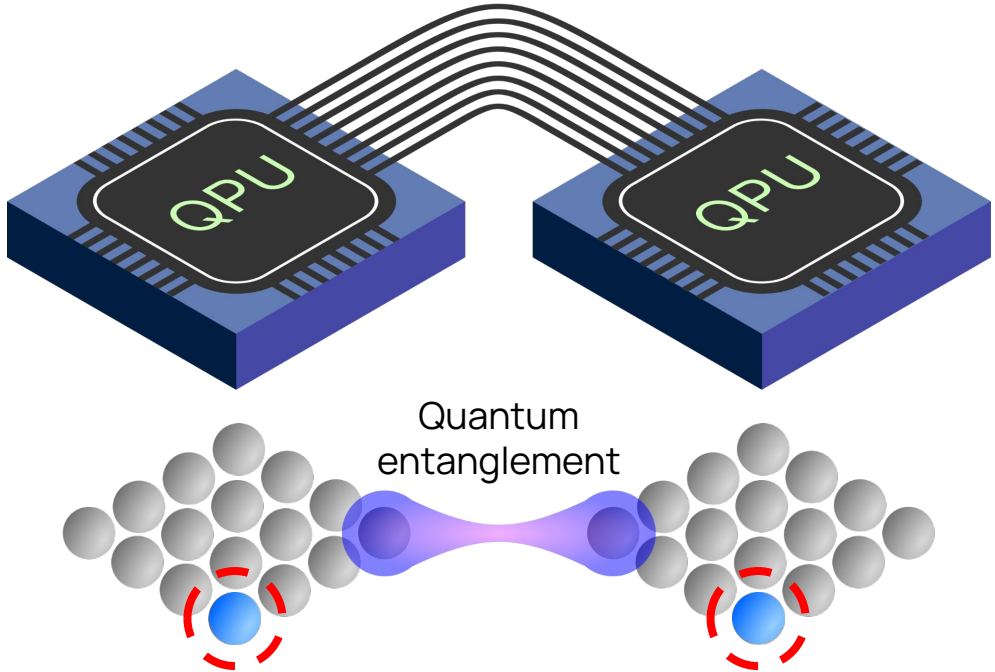
2. **Quantum communication protocols:** how distributed computation?



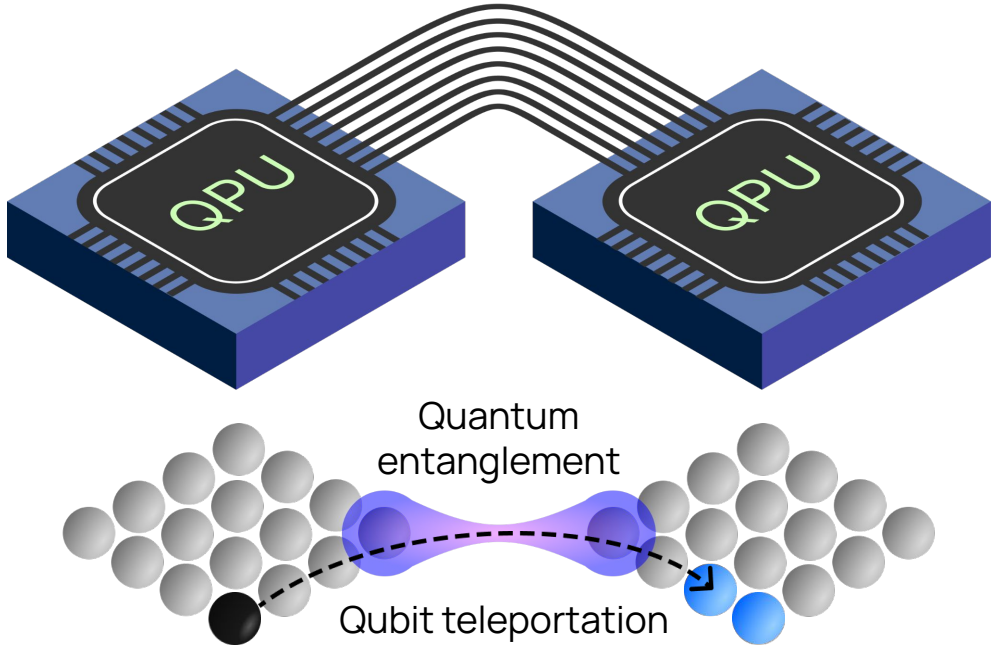
2. **Quantum communication protocols:** how distributed computation?



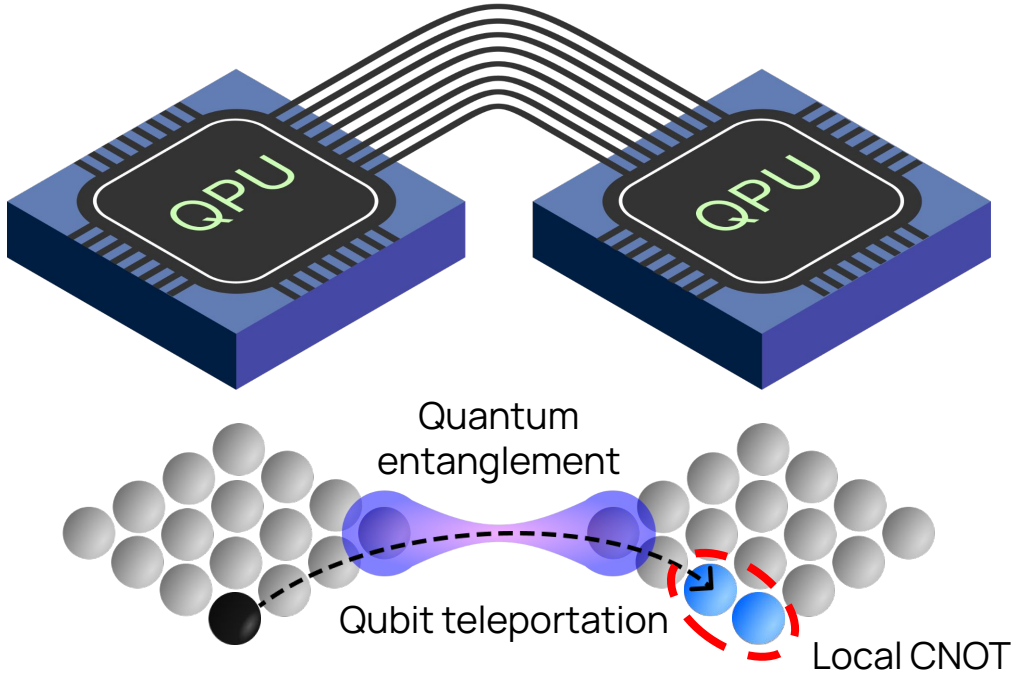
2. Quantum communication protocols: how distributed computation?



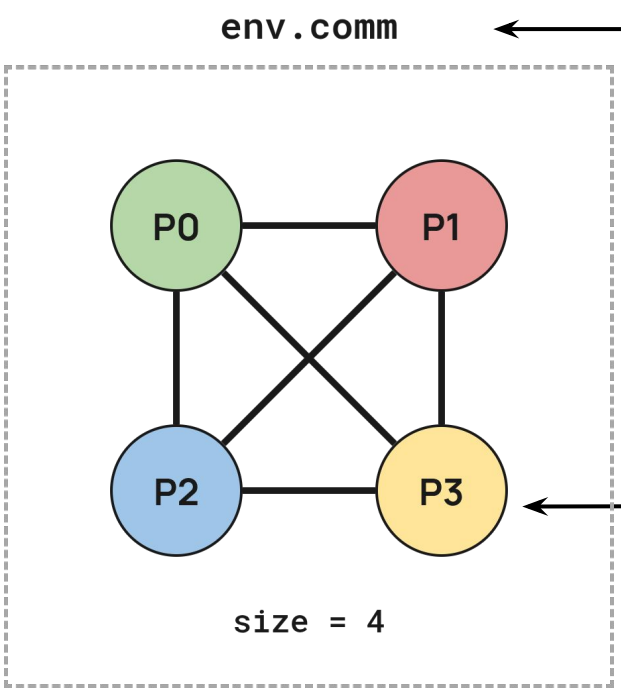
2. Quantum communication protocols: how distributed computation?



2. **Quantum communication protocols:** how distributed computation?



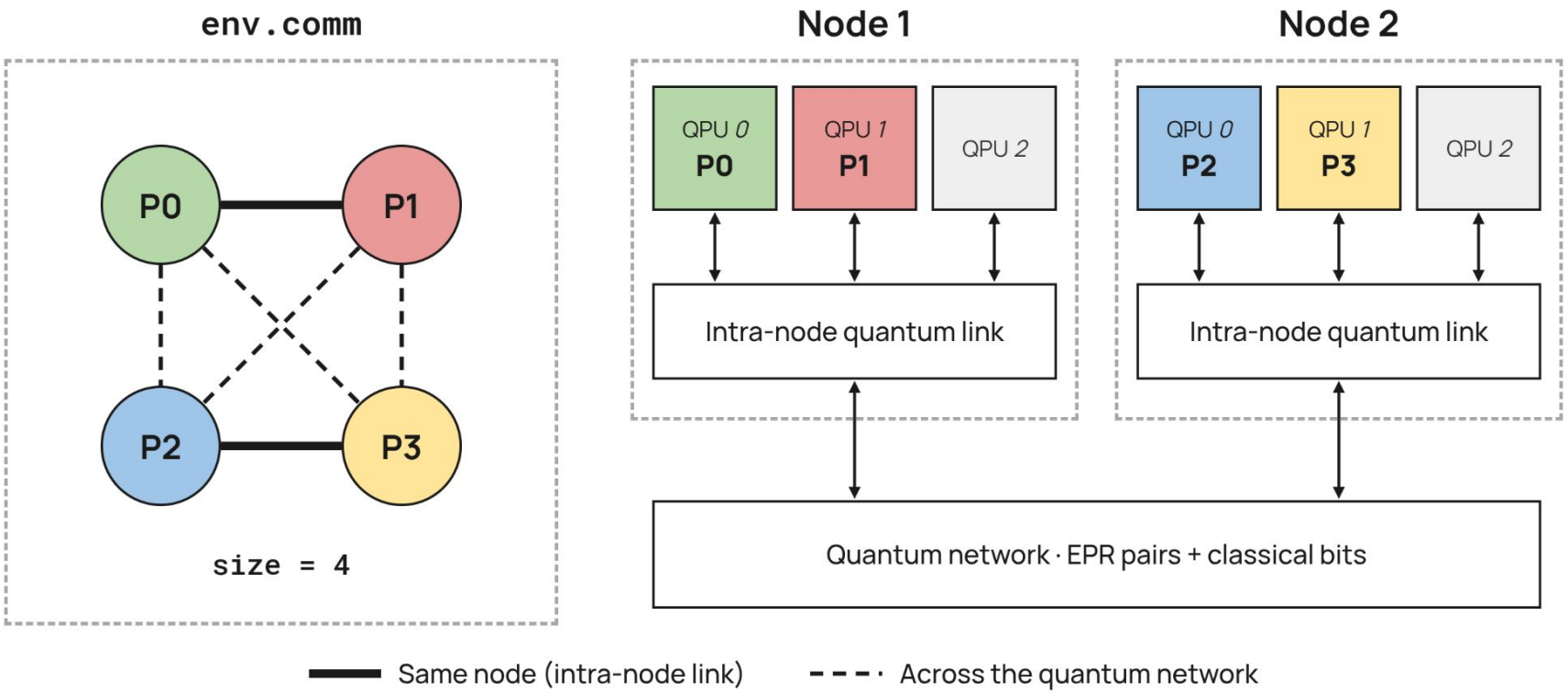
3. NetQMPI programming model: *rank* and *communicator*



Communicator: network abstraction

Rank: computational hardware abstraction

3. NetQMPI programming model: *rank* and *communicator*



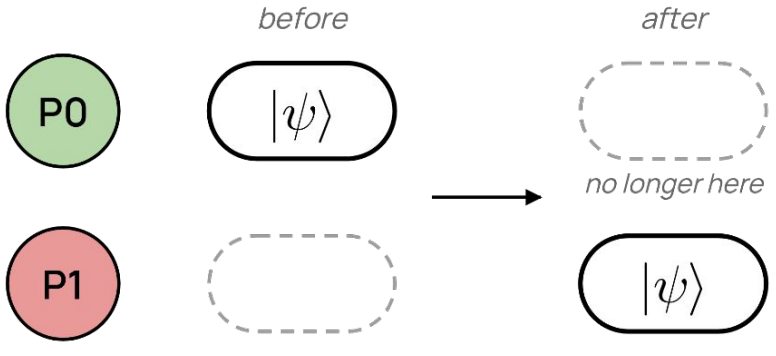
3. NetQMPI programming model: *point-to-point directives*

p2p.py

```
def main(env):
    comm = env.comm
    with comm:
        c = env.create_circuit(1, 1)
        if comm.rank == 0:
            c.h(0)                          P0
            comm.qsend(c, [0], 1)
        else:
            comm.qrecv(c, [0], 0)           P1
            c.measure(0, 0)
```

`qsend / qrecv`

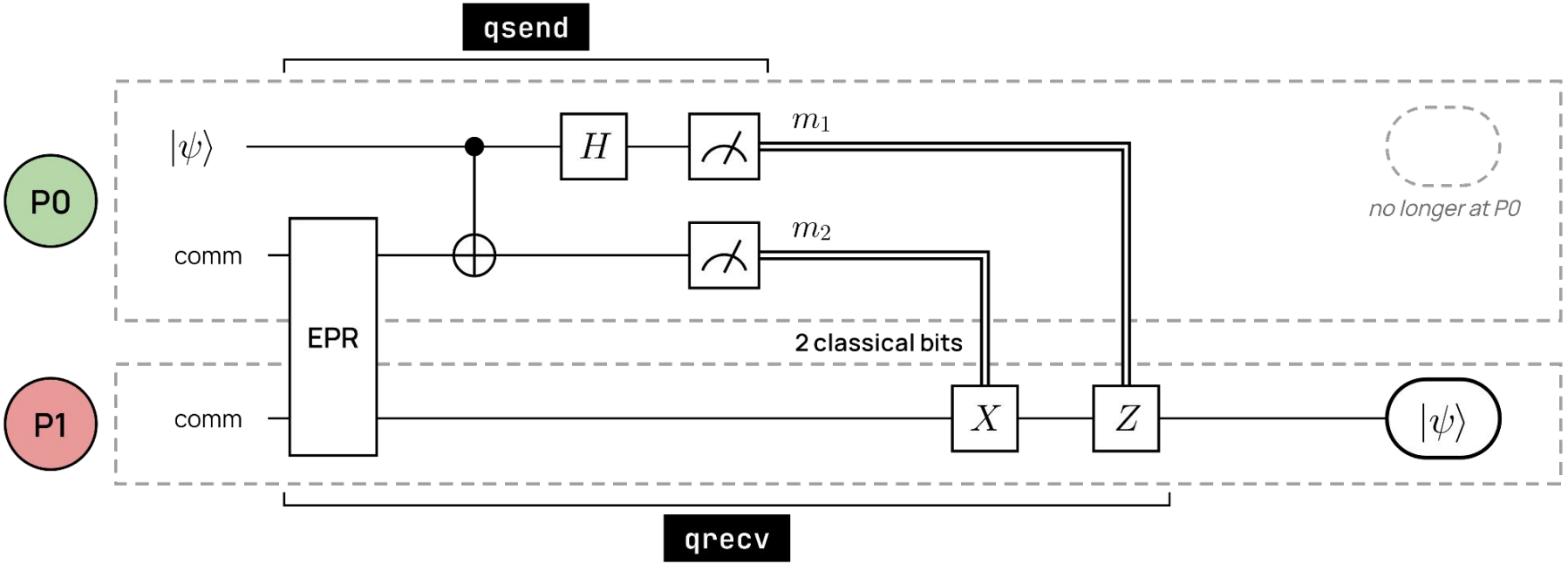
the qubit is moved



$|\psi\rangle|0\rangle \not\rightarrow |\psi\rangle|\psi\rangle$

forbidden by the no-cloning theorem

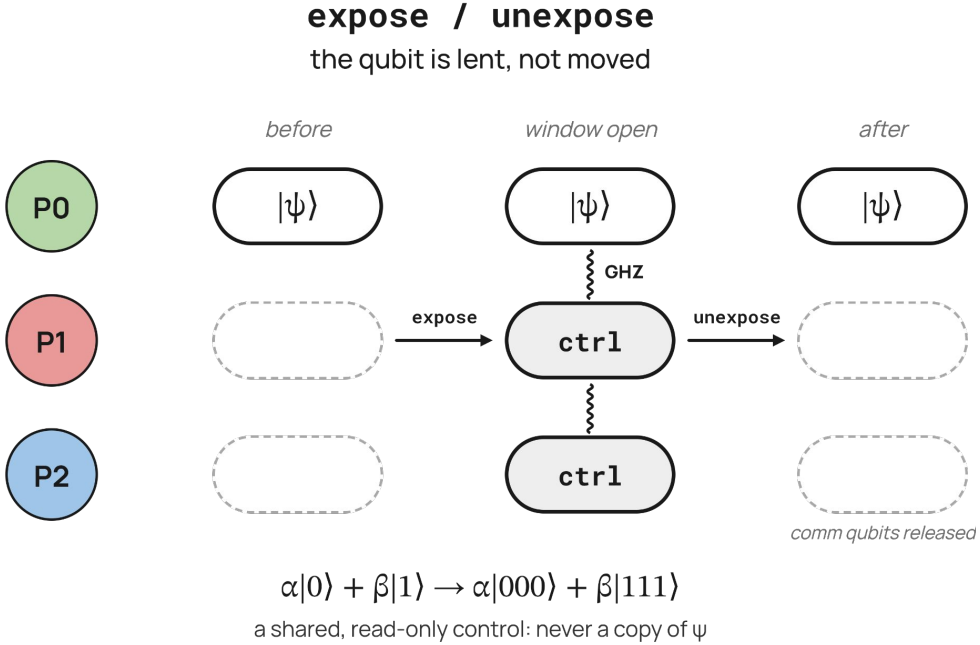
3. NetQMPI programming model: *point-to-point directives*



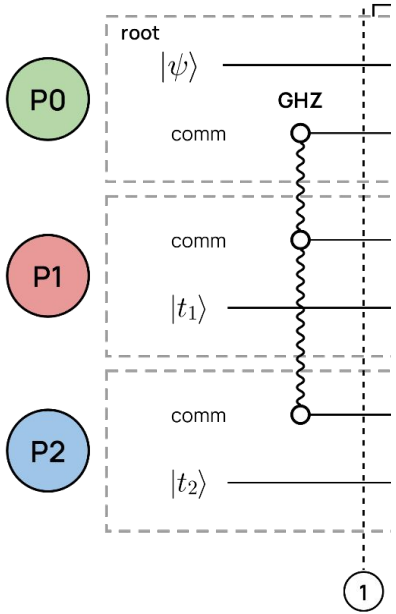
3. NetQMPI programming model: *collective operations*

expose.py

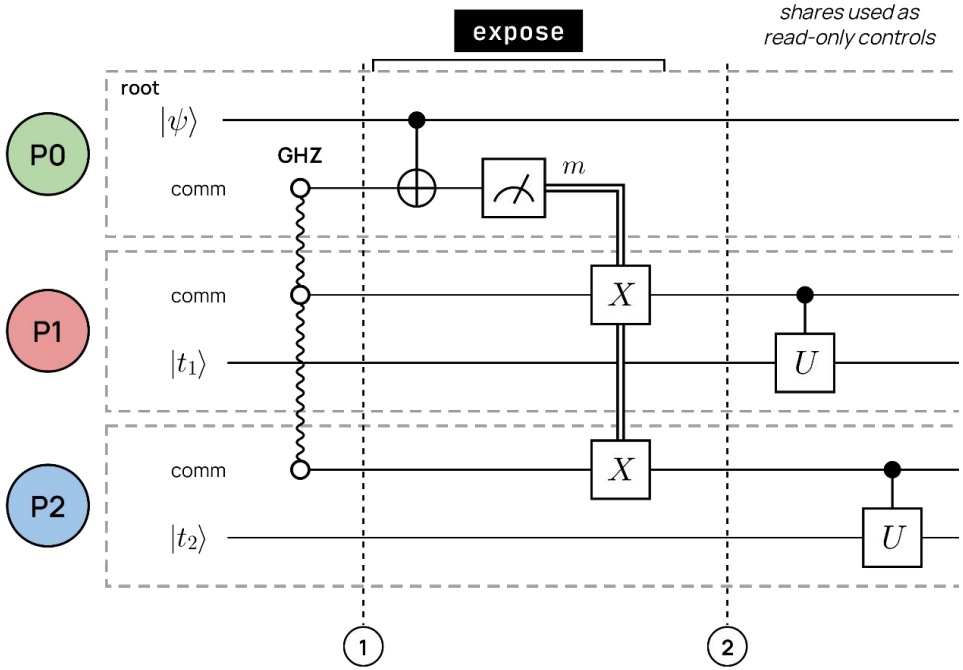
```
def main(env):
    comm = env.comm
    with comm:
        c = env.create_circuit(1, 1)
        if comm.rank == 0:
            c.h(0)
        ctrl = comm.expose(c, 0, [1, 2], root=0)
        if comm.rank != 0:
            c.cx(ctrl, 0)
        comm.unexpose(c, [1, 2], root=0)
        c.measure(0, 0)
```



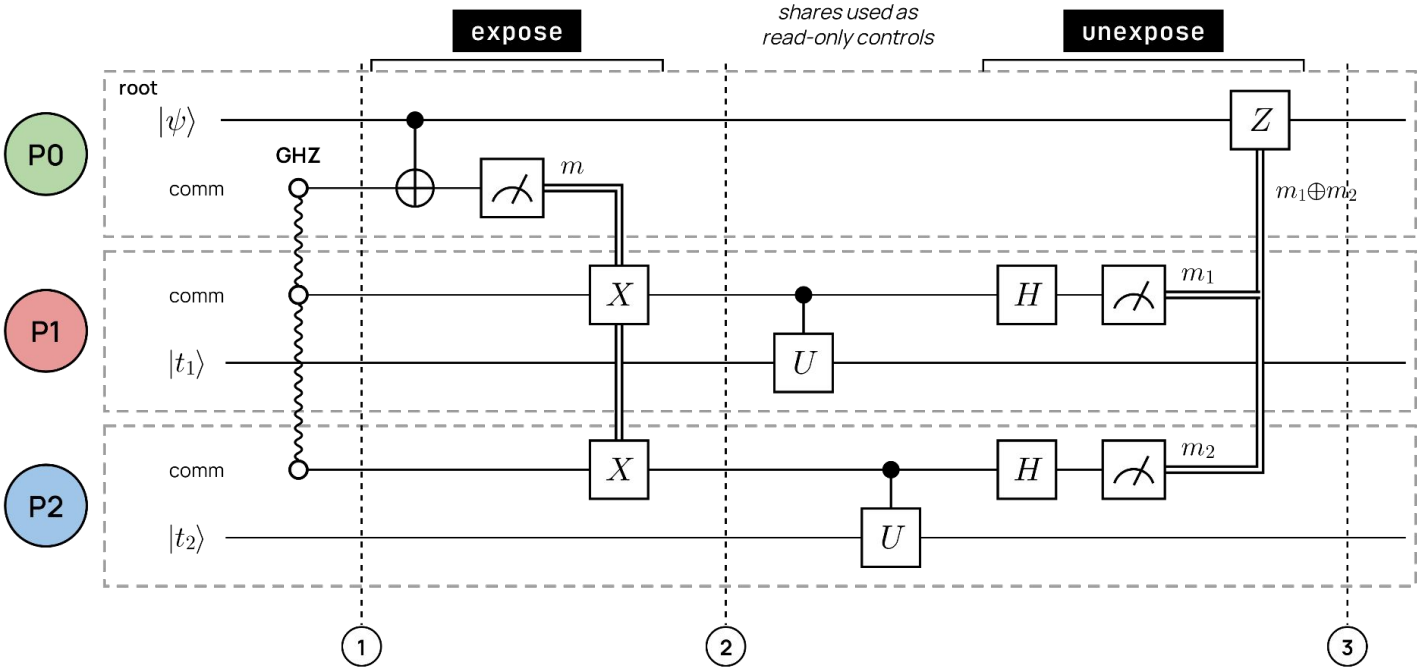
3. NetQMPI programming model: *collective operations*



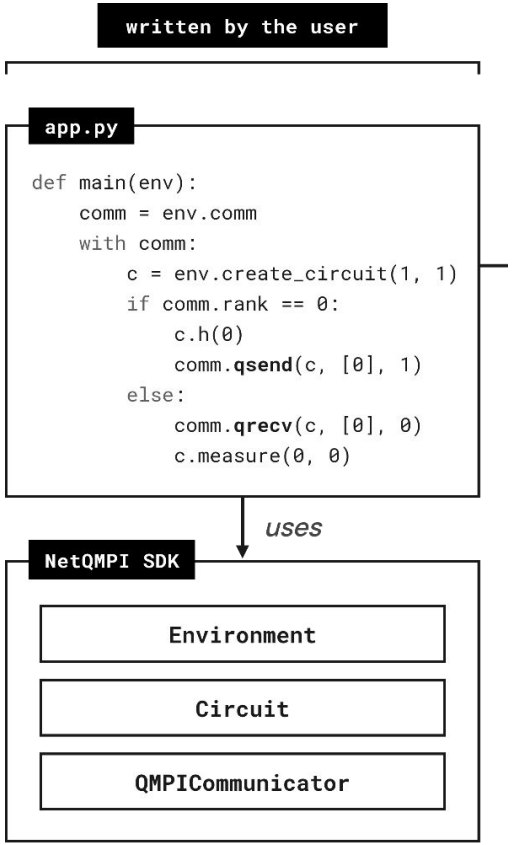
3. NetQMPI programming model: *collective operations*



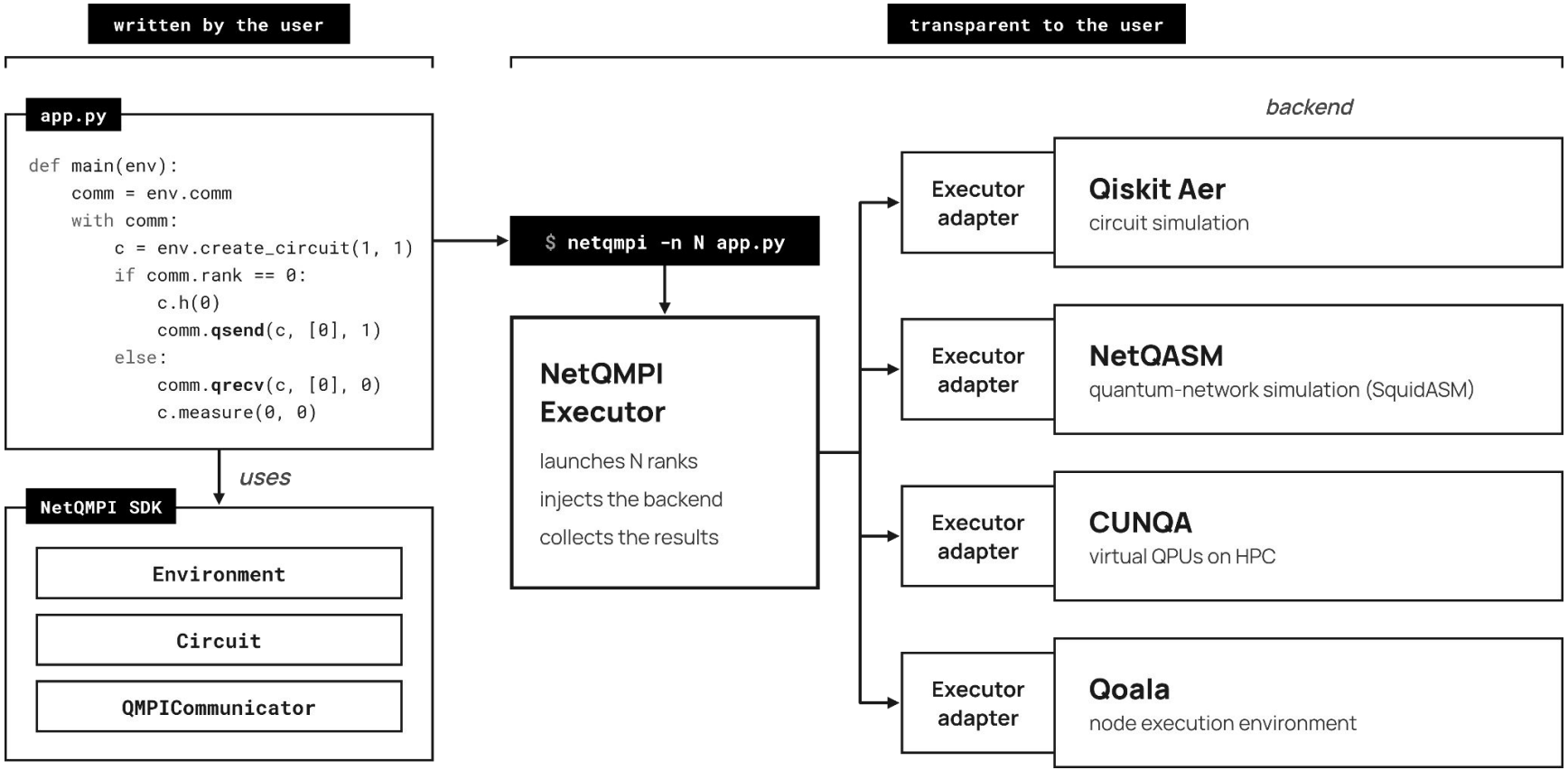
3. NetQMPI programming model: *collective operations*



3. NetQMPI programming model: *multi-backend*



3. NetQMPI programming model: *multi-backend*



4. Conclusions

Conclusions

1. Development of a portable tool for programming DQC systems
2. Abstraction of quantum communications protocols as *teledata* or *telegate.*
3. High-level quantum communications instructions: point-to-point and collective directives

Future work

1. Re-implement the NetQMPI Executor in C++ to reduce runtime overhead.
2. Multi-language SDKs for interoperability with existing HPC codes.
3. Quantify the overhead of using NetQMPI instead of programming each backend directly.

Centro Singular de Investigación en **Tecnoloxías Intelixentes**

CESGA
GALICIA SUPERCOMPUTING CENTER

Portable Distributed Quantum Computing with NetQMPI

F. Javier Cardama (javier.cardama@usc.es)
Jorge Vázquez Pérez
Tomás F. Pena
Andrés Gómez

Quantum SPAIN
España | digital 2026
Plan de Recuperación, Transformación y Resiliencia
Gobierno de España · Ministerio de Ciencia, Innovación y Universidades
Financiado por la Unión Europea NextGenerationEU
Agencia Estatal de Investigación
RES Red Española de Supercomputación