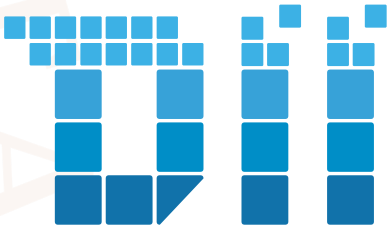


Quantum programming Lab

Leonardo Bacciottini

leonardo.bacciottini@phd.unipi.it

Part of the course in

Quantum Computing and Quantum Internet

held by Prof. Luciano Lenzini

The **Net**work **S**imulator for **Qu**antum **I**nformation using **D**iscrete events (**NetSquid**)

What is NetSquid

Installation guide

Installation

1. Register on the NetSquid official website here
2. Install NetSquid using pip3:

```
pip3 install --extra-index-url https://pypi.netsquid.org netsquid
```

3. Dive into NetSquid documentation

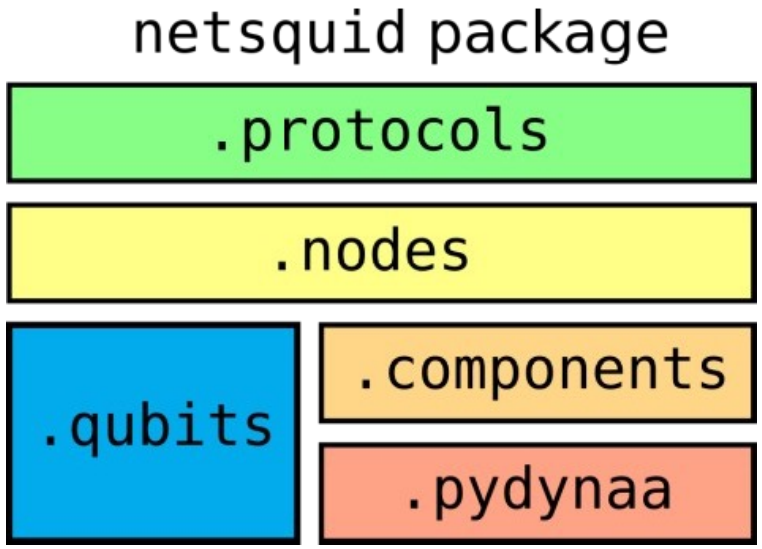
- Suggestion: install NetSquid in a conda environment

NetSquid

- NetSquid is a **discrete event simulator**
- Available for Linux and MacOS systems
- Deployed as a Python package (C++ under the hood)

- Why NetSquid?

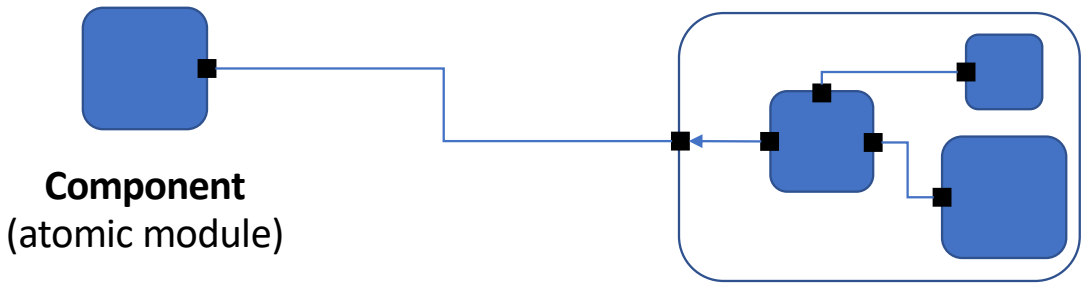
  - ✓ Allows any abstraction level
  - ✓ Extendable



DEPARTMENT OF INFORMATION ENGINEERING

Simulation elements

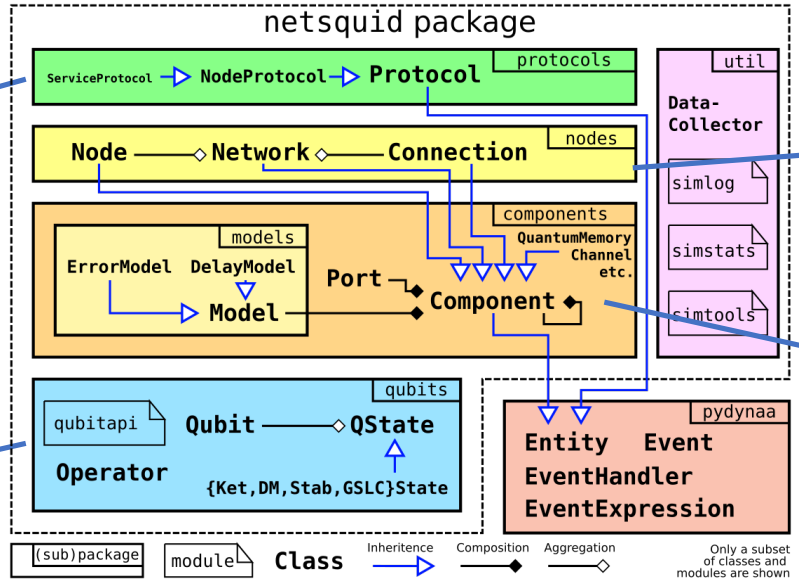
- To model a system we have to define two things:
  1. **Static elements** (atomic and compound modules, ports and connections)



  2. **Dynamic elements** (Behavior of components and nodes)
     - Components come with a pre-defined, well-known behavior
     - *Protocols* can be added to nodes to realize user-defined behaviors

NetSquid package

Nodes behavior



Quantum algebra:

- Qubits definition
- State representations
- Quantum operations

Compound modules:

- Nodes
- Connections
- Networks

Atomic modules:

- Quantum memories
- Channels
- [...]

Simulation engine:

- Entities
- Events
- Event Queue

DEPARTMENT OF INFORMATION ENGINEERING

Pydynaa

- A few words about the simulation engine

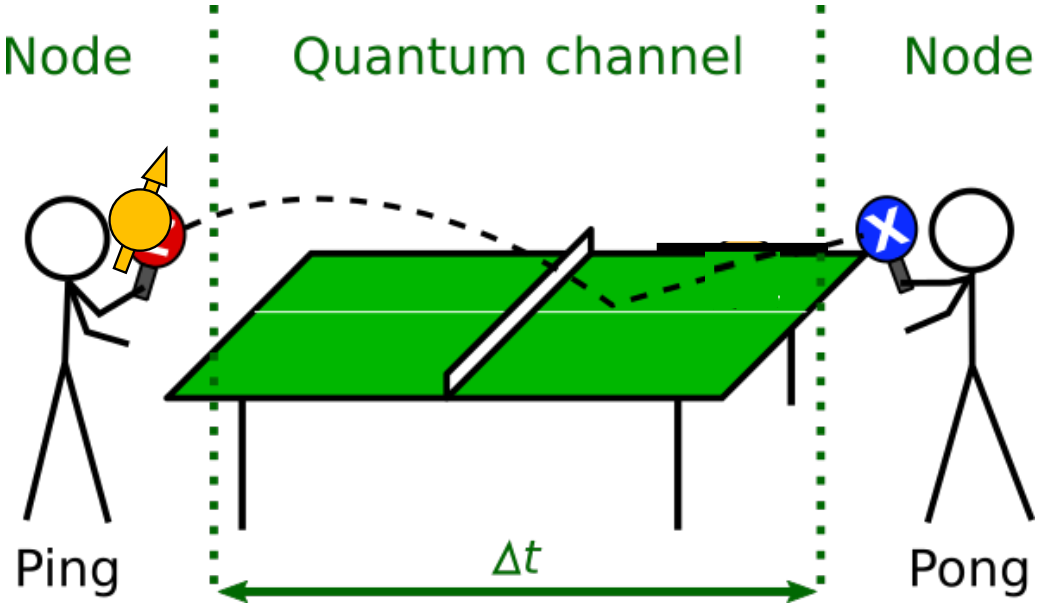
- Simulation advances according to a **discrete event scheme**.
- Simulation entities can wait on a specific type of **Event**, described by an **EventExpression**:

```
my_evexpr = self.await_port_input(node.ports["q0"]) or self.await_timer(duration=500)
yield my_evexpr
```

Quantum Ping Pong

Ping measures in Z basis (CBS):

$$\left\{|0\rangle = \begin{pmatrix}1\\0\end{pmatrix}, |1\rangle = \begin{pmatrix}0\\1\end{pmatrix}\right\}$$



Pong measures in X basis:

$$\left\{|+\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\1\end{pmatrix}, \quad |-\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\-1\end{pmatrix}\right\}$$

Figure taken from [NetSquid website]

Model

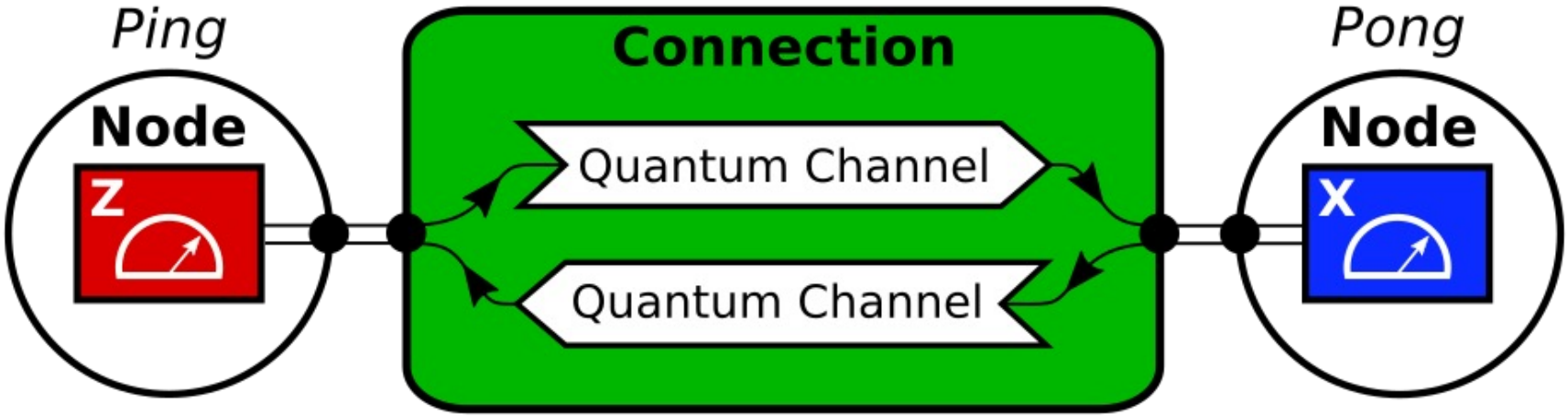
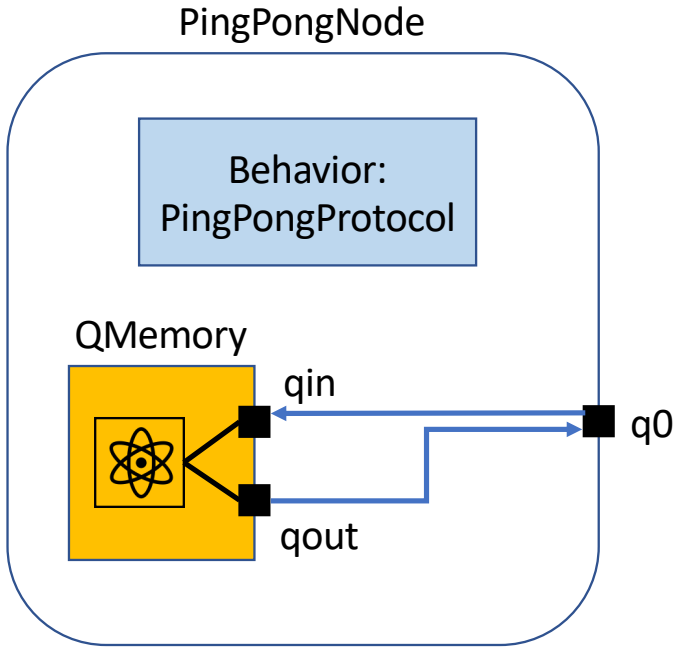


Figure taken from NetSquid website

PingPong Node

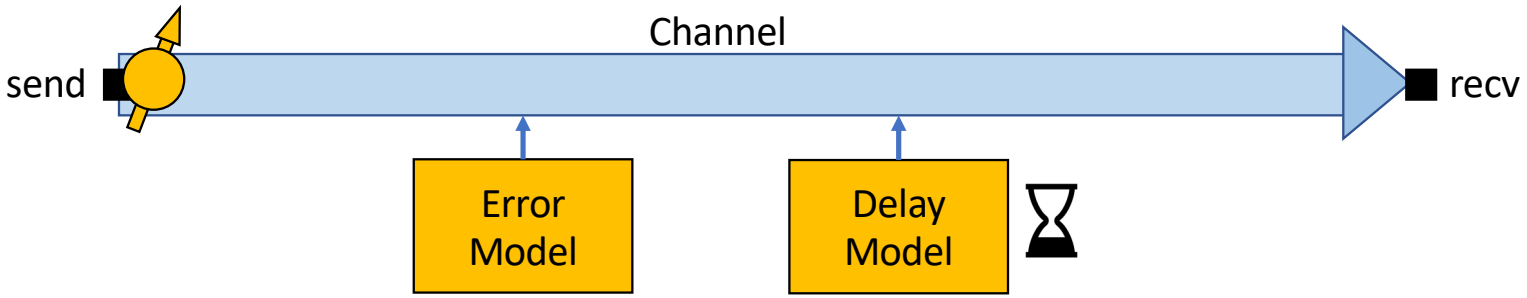


Exercise 1

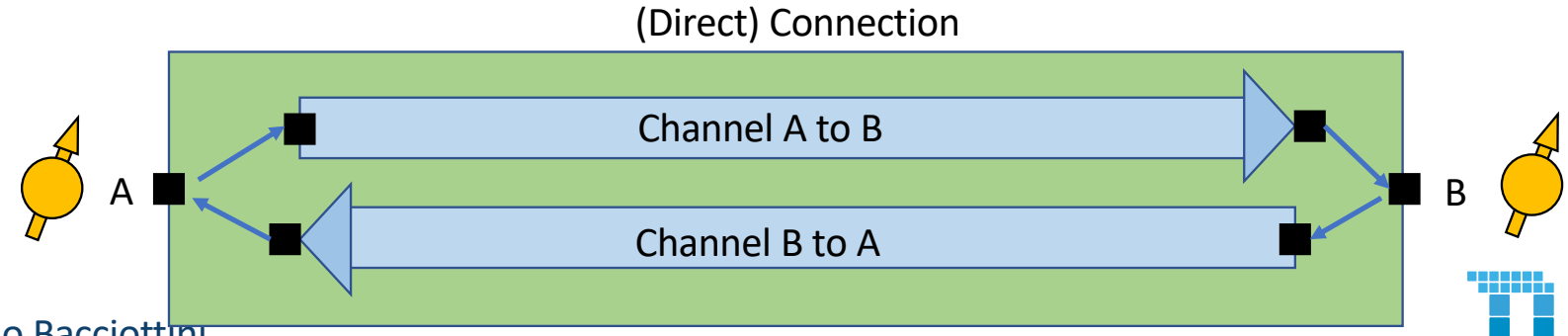
- Open a Python IDE, import **netsquid**
- Implement the PingPongNode class as an extension of the Node class:
  1. Add the quantum memory as a subcomponent
  2. Connect qin/qout ports of the qmemory to the q0 port of the node
  3. Add the PingPongProtocol to define the node behavior.

Channels and Connections

- Channels are **components** that model a one-way link:



- Connections are **nodes** that implement arbitrary complex links:



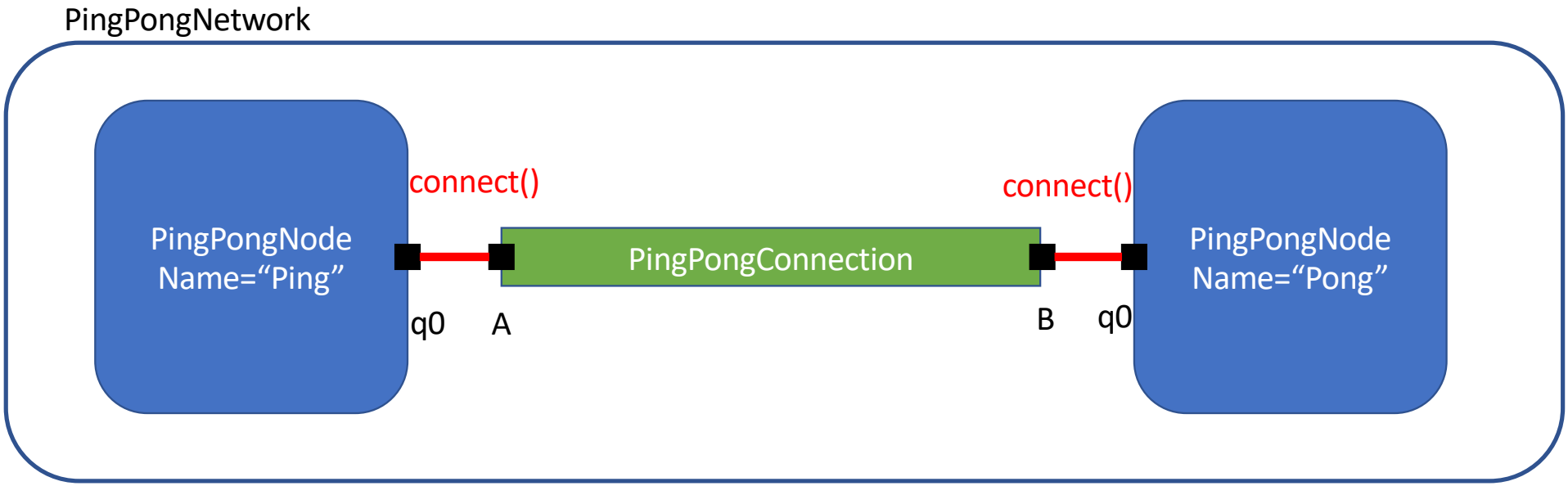
Exercise 2

- Implement the connection from the previous slide on netsquid:
  1. Create a PingPongConnection class extending the class Connection
  2. Add the two channels as sub-components
  3. Forward input/output of channel ports to the connection ports

- Shortcut: Use the class DirectConnection

Avengers, Assemble!

- Assemble the Ping Pong network:



Exercise 3

- Implement the network on netsquid.
- Run the simulation.

Collecting metrics

- Netsquid supports metrics collection through an emit–catch scheme:

- A class called DataCollector can be defined to catch a specific type of EventExpression when it occurs.
- Whenever the DataCollector catches the EventExpression, a handler method is called that may return a metric sample
- At the end of the simulation we can access the metrics set as a Python dataframe.