

NetSquid



A tool for Quantum Network design

IETF Quantum Internet Research Group interim

Rob Knegjens, on behalf of the NetSquid dev team

April 8th 2020

The NetSquid Project

Network Simulator for Quantum Information using Discrete events

NetSquid



- Developed at QuTech (TNO and TUDelft) since 2017
- In active use by
 - Groups of Stephanie Wehner and David Elkouss (QuTech QINC roadmap)
 - Partners in the Quantum Internet Alliance (EU Quantum flagship)

Public beta release: very soon!

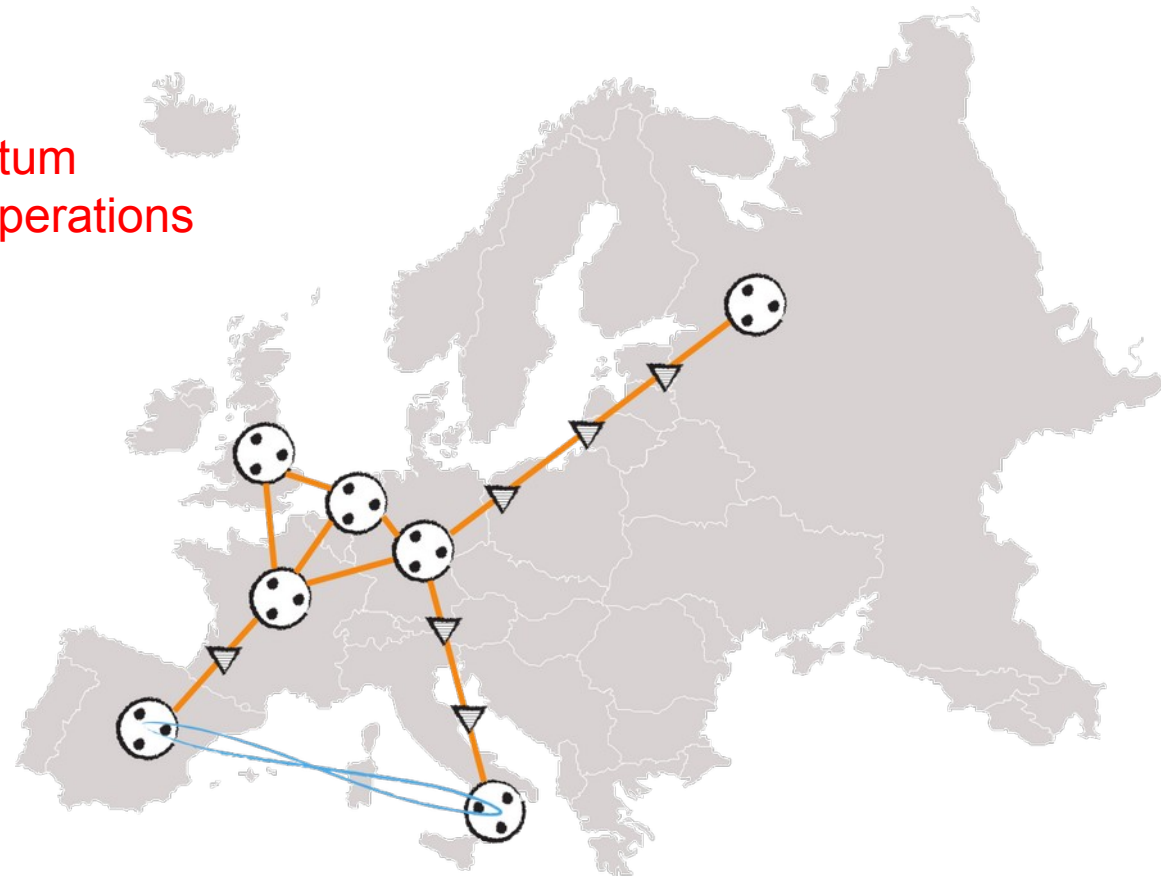
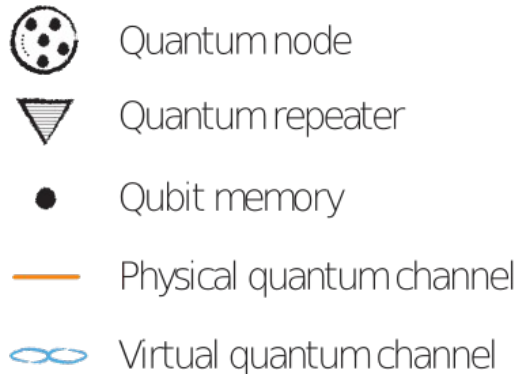
<https://netsquid.org>



A Quantum Internet

Key resource: quantum entanglement

- Rate
 - \downarrow **loss**
 - Fidelity
 - \downarrow **noise**
- } Non-ideal quantum channels and operations

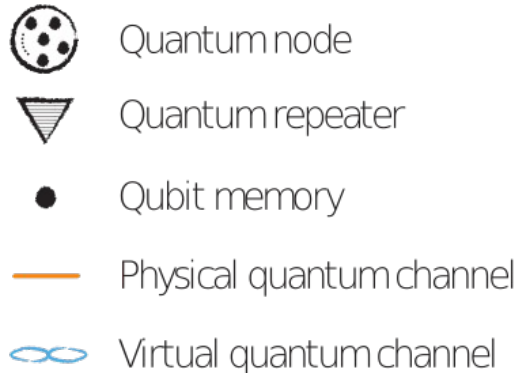


A Quantum Internet

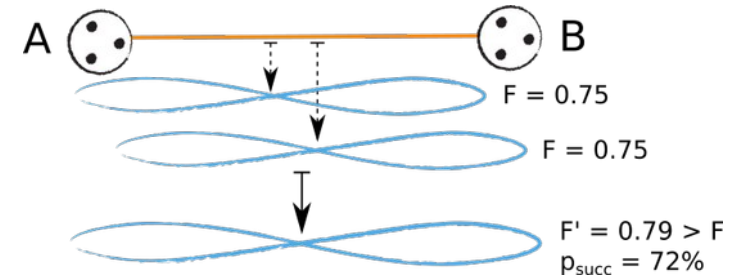
Key resource: quantum entanglement

- Rate
 - Fidelity
- ↓ **loss**
• ↓ **noise**
- } Non-ideal quantum channels and operations

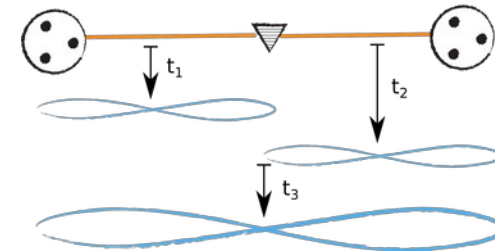
Mitigate:



Noise: entanglement purification

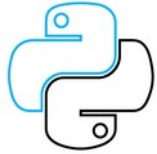


Loss: quantum repeaters



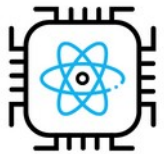
Designing a quantum internet involves solving complex timing dependencies

The NetSquid Simulator



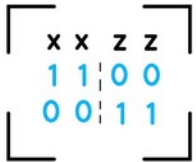
Python 3 package

Optimised C/Cython code under the hood.



Specialised quantum computation library

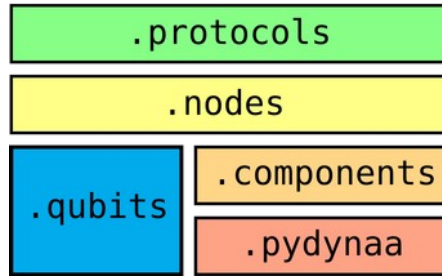
“Qubit-centric” and optimized for repeated sampling.



Seamless choice of quantum state formalisms

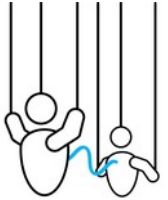
Trade-offs: performance, scalability and versatility

netsquid package



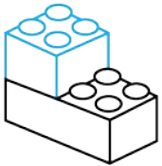
Asynchronous framework

For programming quantum network protocols.



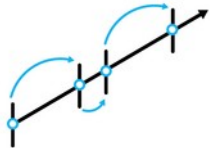
Modular component library

Physically model network hardware with composable base classes.



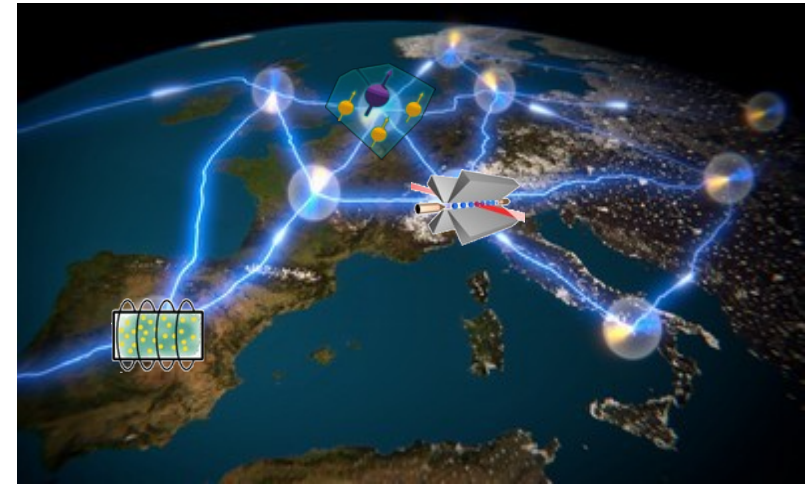
Discrete event simulation engine

Accurately track quantum decoherence across a network in time.



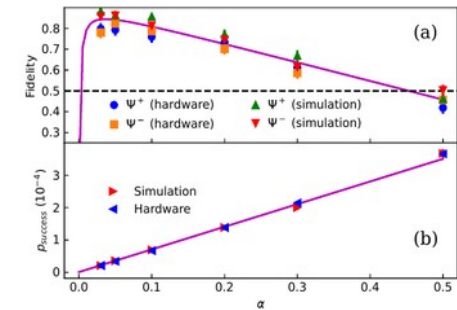
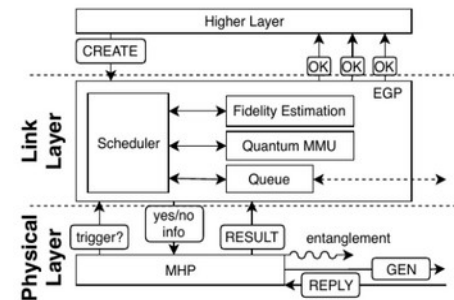
Use cases

- Accurately model the effects of time on the performance of scalable quantum networks
- Investigate the requirements and feasibility for the layers of a quantum internet stack:
 - *physical* → *control plane* → *user applications*
- Emulate future hardware for demonstrator setups



Examples

- Performance of a *quantum link layer protocol*
- Parameter optimization and benchmarking for a pan European quantum internet (QIA)
- Parameter sensitivity for repeater chains*



A Link Layer Protocol for Quantum Networks

Axel Dahlberg, Matthew Skrzypczyk, Tim Coopmans, Leon Wubben, Filip Rozpędek, Matteo Pompili, Arian Stolk, Przemysław Pawełczak, Robert Knegjens, Julio de Oliveira Filho, Ronald Hanson, Stephanie Wehner

*T. Coopmans et al, APS 2019

Getting started with NetSquid

First register at the NetSquid forum: <https://forum.netsquid.org>

uses forum credentials

Online documentation

NetSquid Forum
Online community for discussing the NetSquid quantum network simulator

Register Login

Unanswered topics Active topics

FAQ Search

NetSquid / Board index It is currently 06 Apr 2020, 05:59

FORUM	STATISTICS	LAST POST
NetSquid Documentation Online documentation pages for NetSquid. Sign in using your forum credentials.		
Official Announcements and Information Official news, updates and reference information from the NetSquid development team.	Topics: 1 Posts: 1	Version 0.8.0 by Leon 09 Mar 2020, 08:37

NetSquid

0.8.6

Search docs

ABOUT

- Introduction
- Installation
- Release notes
- Authors
- License

LEARN NETSQUID

- NetSquid in 10 minutes
- Tutorial
- Examples

API REFERENCE

- NetSquid package
- pyDynAA package

Docs » NetSquid in 10 minutes View page source

Previous Next

NetSquid in 10 minutes

Welcome to this quick start guide to NetSquid. In this guide we will give a brief outline of how to use NetSquid. For a more in depth introduction you can follow the [tutorial](#) or experiment with the provided [examples](#). To run the code examples presented in this guide you will need to have NetSquid installed, which you can do by following [these instructions](#). If you have any questions along the way please feel free to ask them on the [NetSquid forum](#).

We will give a brief walk-through of how to setup and run a quantum network simulation that plays a game of ping-pong between two nodes using a qubit. If the concept of a qubit is completely new to you it may help to first refer to the [start of the tutorial](#) for a short introduction. The game is played by the two remote nodes, *Ping* and *Pong*, who take turns measuring the qubit in their preferred basis – the standard (Z) or Hadamard (X) bases respectively – and then sending the qubit on to the other via a shared quantum channel. A cartoon of this setup is illustrated below:

**Install via
PyPI server:**

```
pip3 install --extra-index-url https://<username>:<password>@pypi.netsquid.org netsquid
```

Your forum credentials

User license (pending): free for non-commercial use

Snippets: user extensions to NetSquid

- *NetSquid Snippets* are Python packages that extend NetSquid
- Created, maintained and **shared** by users
- Some snippets already hosted on NetSquid PyPI server

Examples:

Package name	Description	Maintainer*
netsquid-physlayer	Physical layer modeling	Axel Dahlberg
netsquid-nv	NV centre modeling	Tim Coopmans
netsquid-ae	Atomic ensembles	David Maier
netsquid-netconf	Network configuration	Guus Avis
netsquid-magic	Magic EGPs	Tim Coopmans
netsquid-qmm	Quantum memory manager	Axel Dahlberg
netsquid-qpm	Quantum program manager	Wojciech Kozlowski
netsquid-simulationtools	Simulation tools for NetSquid	Guus Avis

New snippets can be generated using the template repository:

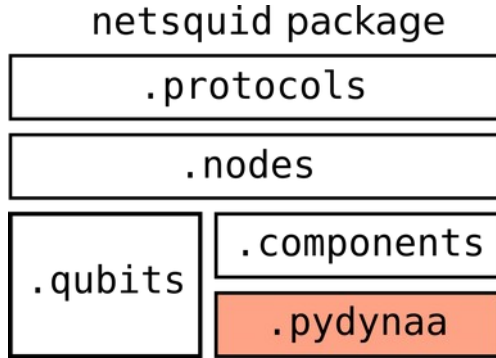
<https://github.com/SoftwareQuTech/NetSquid-SnippetTemplate>

* *maintainers in some cases not the creator(s)*

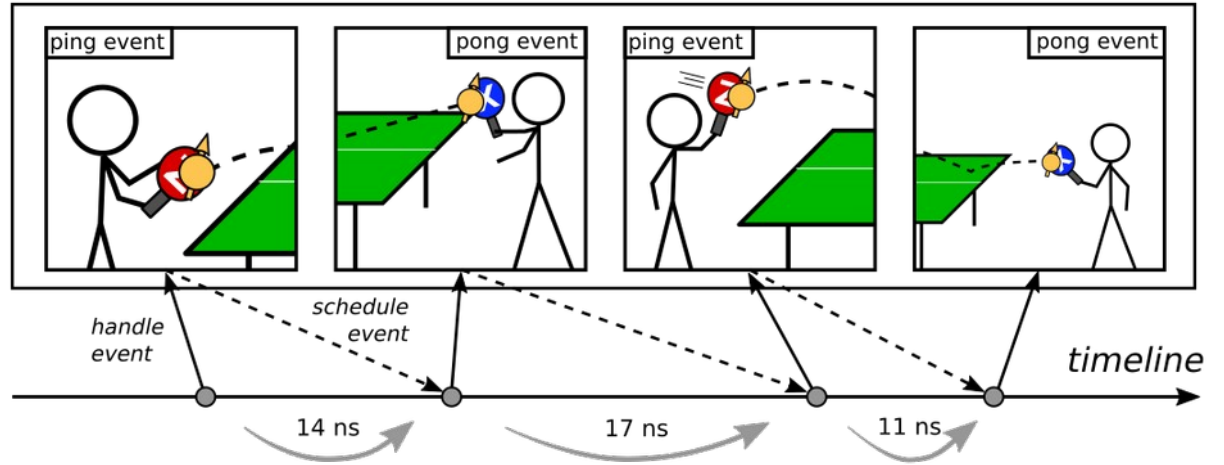
More info at <https://netsquid.org>

Backup slides

The NetSquid Package (I)

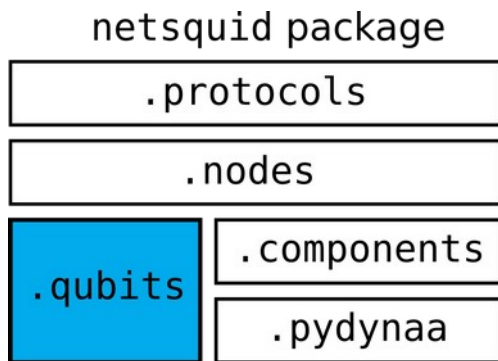


PyDyna subpackage provides the **discrete event simulation engine**
*Based on the core of the DynAA simulator**

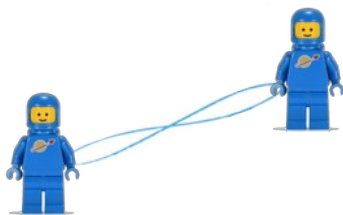
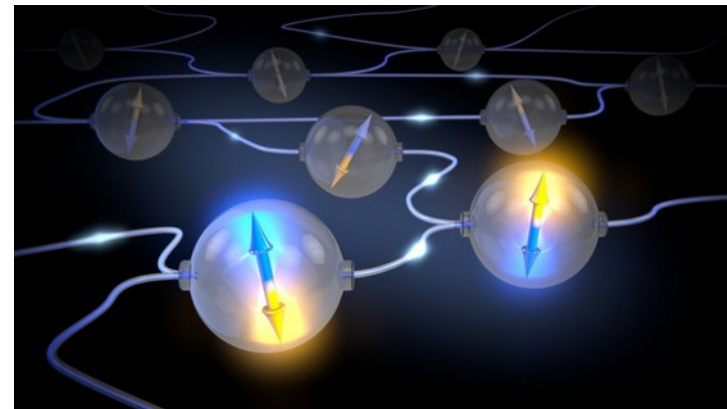


Time progresses by stepping from event to event.

The NetSquid Package (II)



- **Qubit** objects *dynamically* share quantum states
- Computation optimised for repeated sampling
- Seamless choice of quantum state formalisms for **small and large networks**

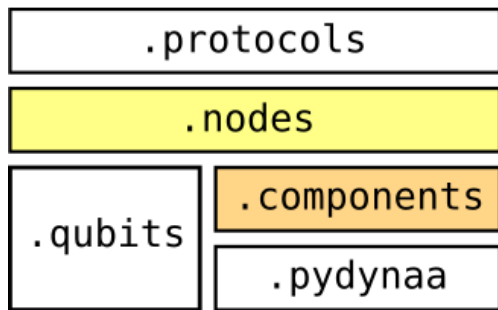


Formalism trade-offs: universality, memory efficiency, speed

Ket vector	Density matrix	Stabilizer state
$ \Phi^+\rangle$	$\rho = \Phi^+\rangle\langle\Phi^+ $	$\{X \otimes X, Z \otimes Z\}$
$ \Phi^+\rangle = \frac{1}{\sqrt{2}} (00\rangle + 11\rangle)$	$\begin{pmatrix} \frac{1}{2} & 0 & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} \end{pmatrix}$	$\begin{array}{cc cc} X & X & Z & Z \\ \hline 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{array}$
$\sim 2^n$	$\sim 2^{2n}$	$\sim 2n^2$

The NetSquid Package (III)

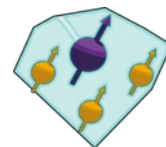
netsquid package



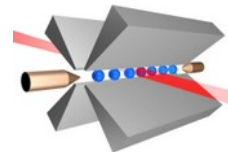
Components are the physical entities of a network:

- **Base components:** channels, quantum processing devices, photon sources, ...
- **Composite components:** nodes, connections, ...
- **Attachable physical models:** delay, loss, memory noise, gate noise

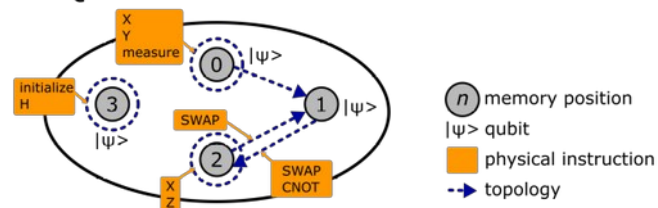
NV centre



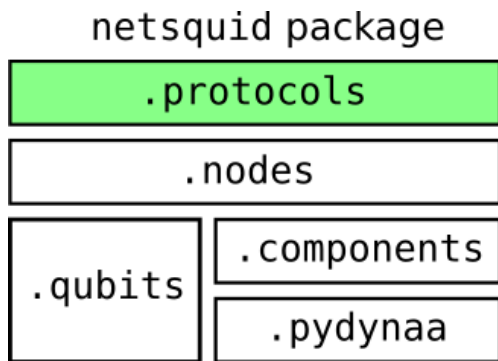
Trapped ions



QuantumProcessor



The NetSquid Package (IV)



Protocols dictate node behaviour (virtual entities)

- Numerically simulate via random sampling
- Massively parallelizable



Example: a distillation protocol

