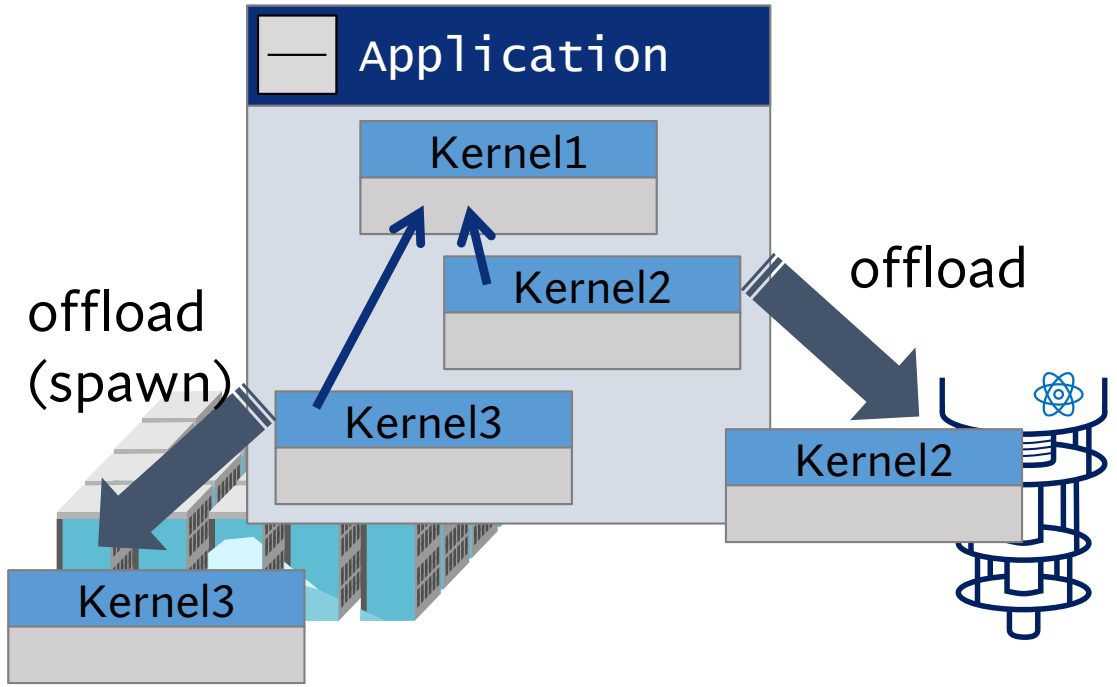
The Role of Supercomputers for Quantum Computing and the Programming Environment Toward the Cooperative Computation of Classical and Quantum Computers

辻 美和子
プログラミング環境研究チーム/量子HPCソフトウェア環境開発ユニット
理研計算科学研究センター

PCCC AI/機械学習技術部会 第 4 回ワークショップ

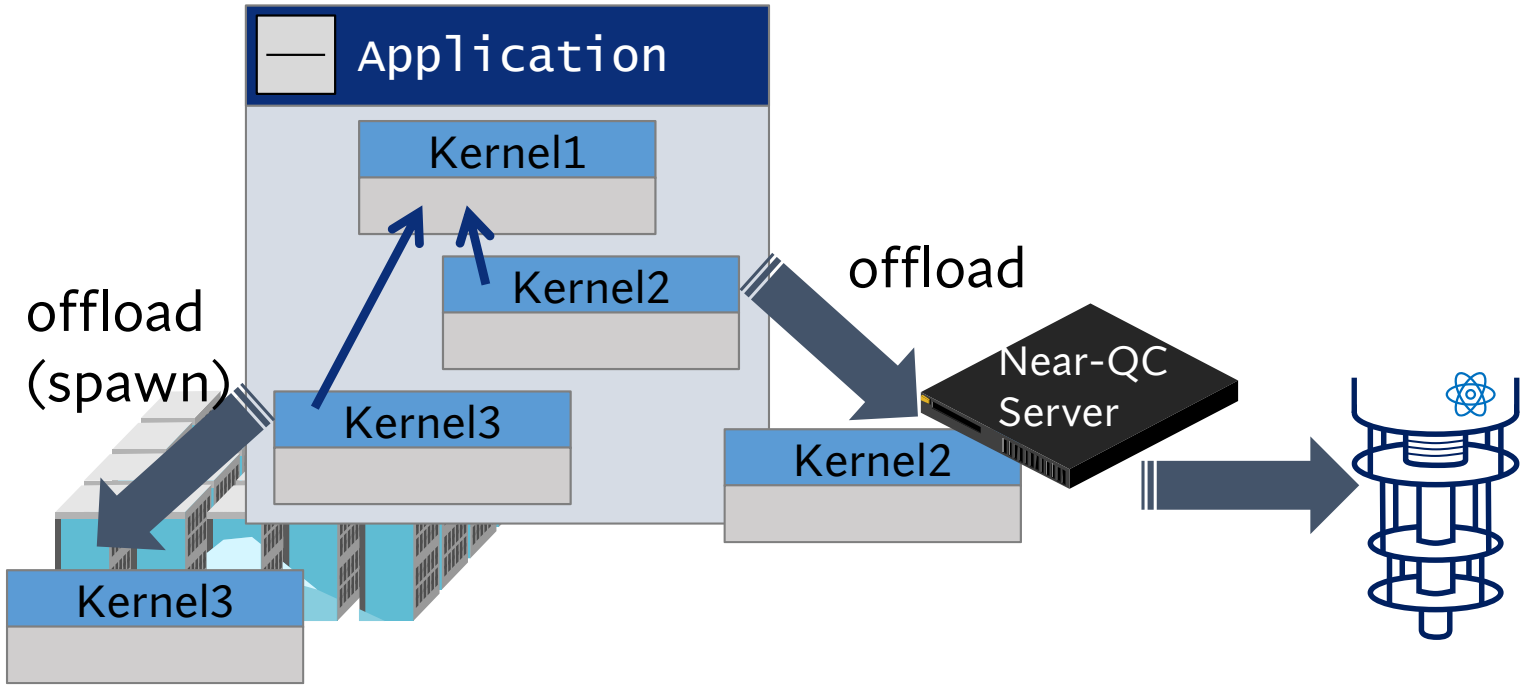
量子HPCアプリケーションに向けたプログラミング環境

- ひとことに「量子HPC」といっても
  - 量子HPCのハイブリッド
    - HPCによって量子計算をサポートあるいは制御する．例）VQE, エラー訂正
  - 量子HPCの協調計算
    - 単一のアプリケーションで，カーネルごとに量子 or HPC を使い分ける
- 次世代 (2030~) の量子計算機は，HPCシステムとは独立なシステムである（はず）
  - ~~a single CPU+QPU chip~~
- いずれにせよ，量子HPCを密に接続し，両者を適切につかいこなすためのフレームワークを提供することが必要
  - スーパーコンピュータからカーネルやアプリケーションを量子計算機にオフロードする
  - オフロードには遠隔手続き呼び出し (RPC) フレームワークを用いる



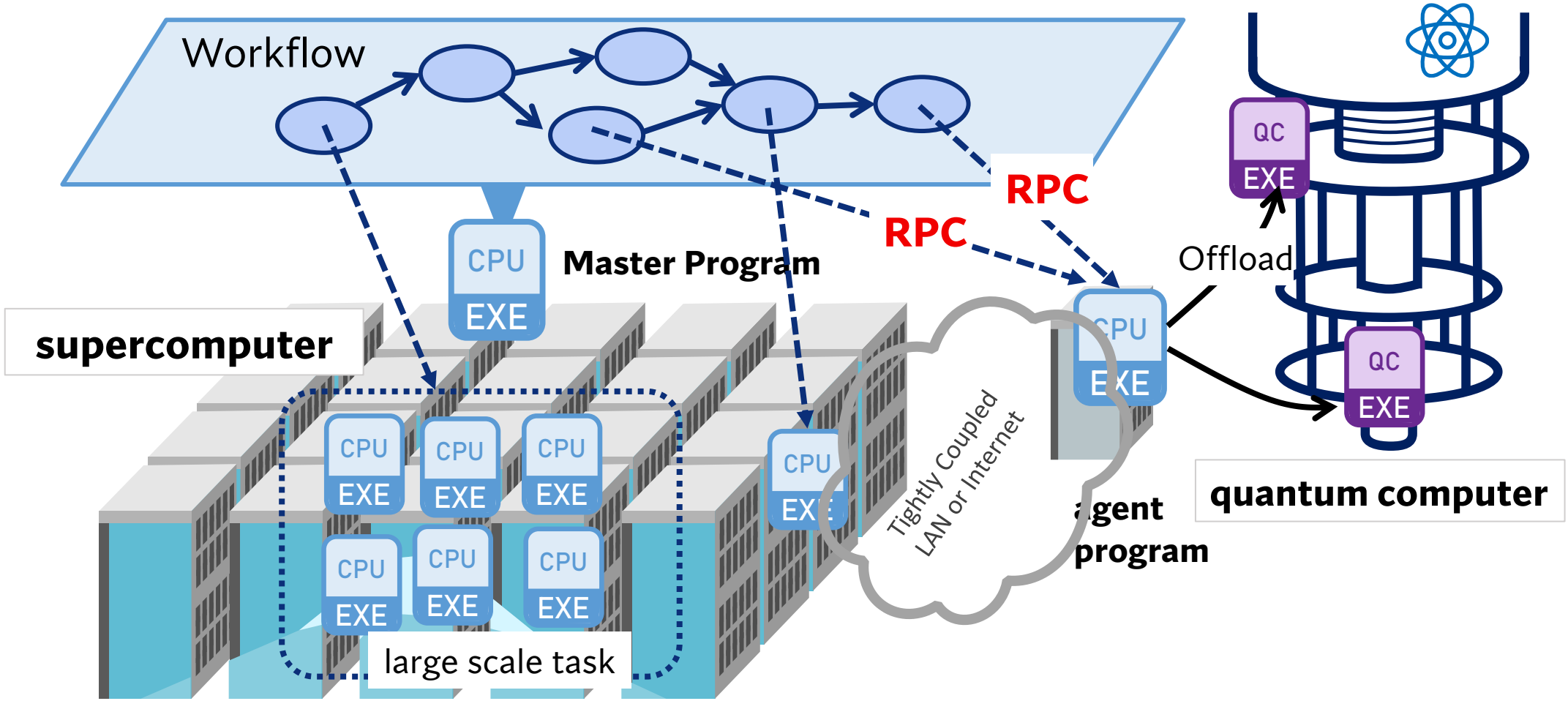
量子HPCアプリケーションに向けたプログラミング環境

- より厳密には，，



量子HPCアプリケーションに向けたプログラミング環境

- 量子HPCにおける協調計算
  - プログラミングモデル ⇒ タスクベース
  - ワークフローをOpenMPやPythonなどで記述する

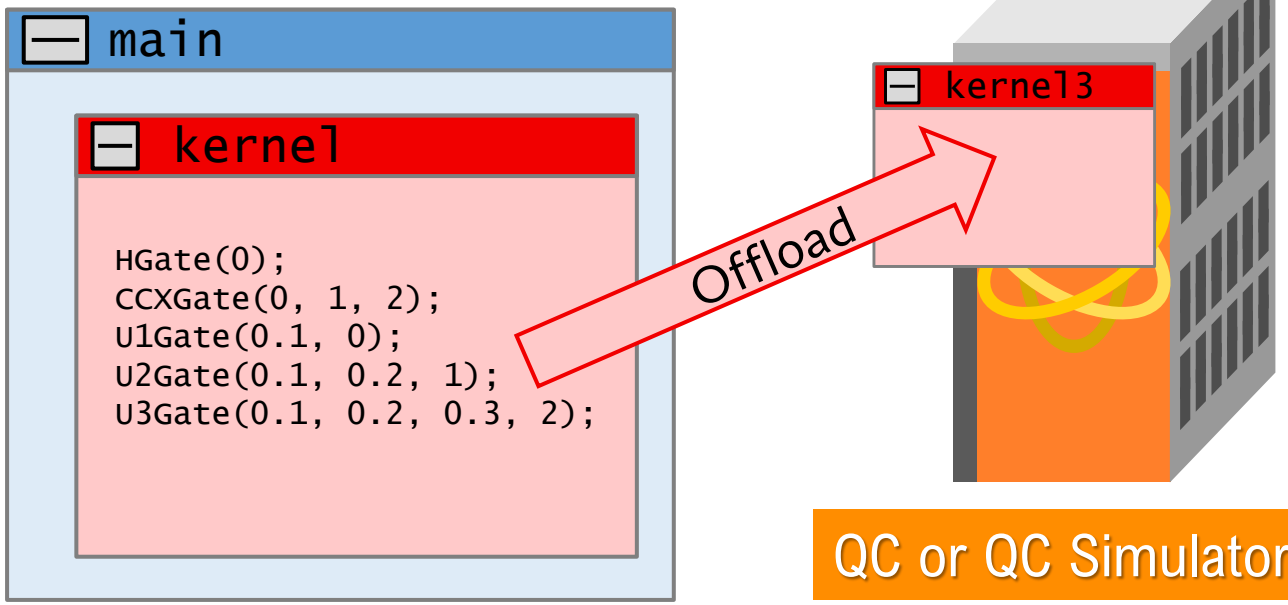


タスクベースの量子HPCアプリケーションの記述例（案）

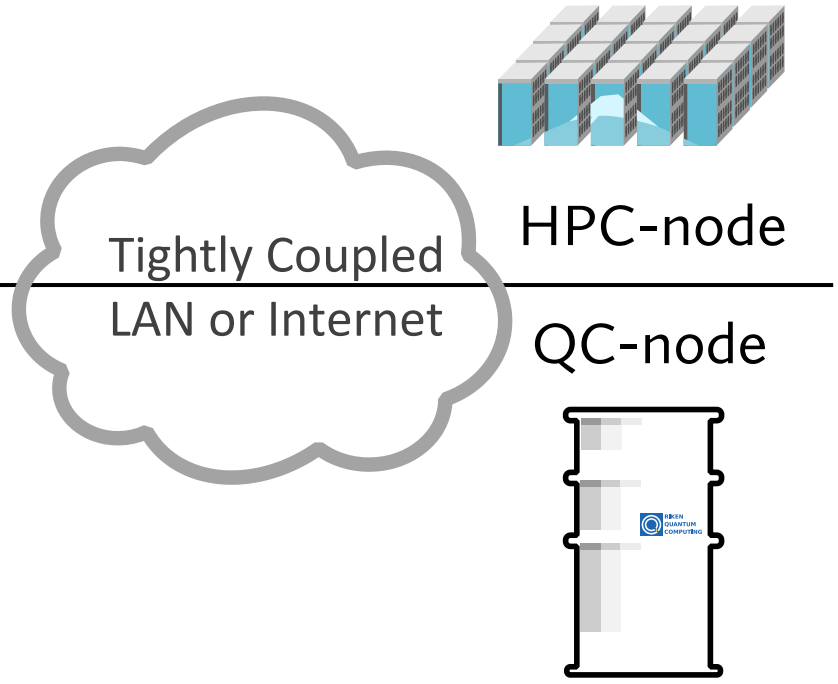
```
#include"qcs_api.hpp“

int main(int argc, char **argv)
{
  int qubits = 5;

  QC_Init(&argc, &argv, qubits);

  HGate(0);
  CCXGate(0, 1, 2);
  U1Gate(0.1, 0);
  U2Gate(0.1, 0.2, 1);
  U3Gate(0.1, 0.2, 0.3, 2

  QC_Measure();

  QC_Finalize();
}
```

※ This is still under development.
The actual product may look different

- 仮に, “**Qiskit-like**” API とする
- 量子計算機（か，シミュレータ）を初期化
- ゲート操作を QC-API library にストアする
- QC-API library は一連のゲート操作をRPCを用いてリモートシステムに転送する，か，同じノードでシミュレータを実行する場合は，単に実行する
- 終了処理



Inside of RPC-based 量子HPCプログラム

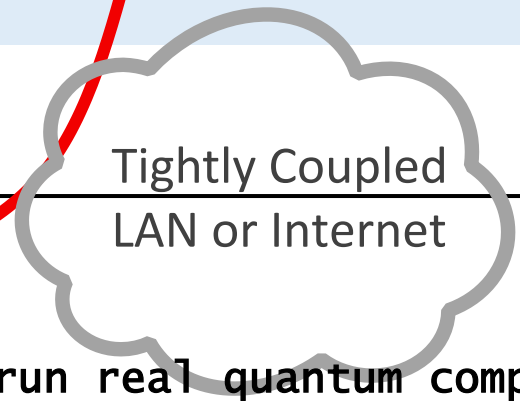


Inside of RPC-based 量子HPCプログラム

```
int qubits = 5;
OmniRpcInit(&argc, &argv);
QC_Init(&argc, &argv, qubits);

HGate(0);
…

QC_Measure();

QC_Finalize();
OmniRpcFinalize();
}
```

main program

```
extern "C" void QC_Measure(){
  for(int i=0; i<ngates; i++){
    // all the intermediate representations
    // (gate, arguments, etc..) are copied to a buffer
  }
  // offload a series of gate operations
  // to a remote node using RPC library
  rq1 = OmniRpcCallAsync("rpc_qc", qcs_info->qubits,
   ngates, buf);
  OmniRpcWait(rq1);
```



HPC-node

QC-node

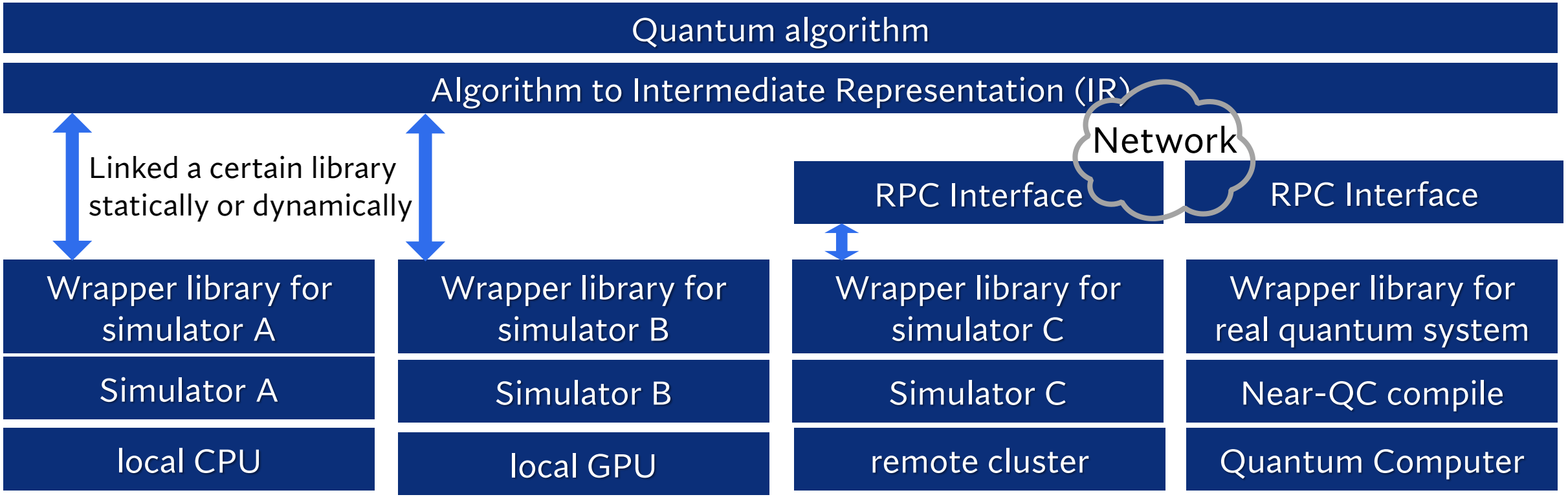
```
Define rpc_qc(IN int nqubits, IN int ngates, IN int nint,
IN void buf[]){
  qcs_info_t *qcs_info;
  for(int i=0; i<ngates; i++){
     // copy the received buffer
  }
   ..
  qcs_measure(qcs_info);
 qcs_finalize_lib();
}
```

```
// perform simulation (or run real quantum computer)
```

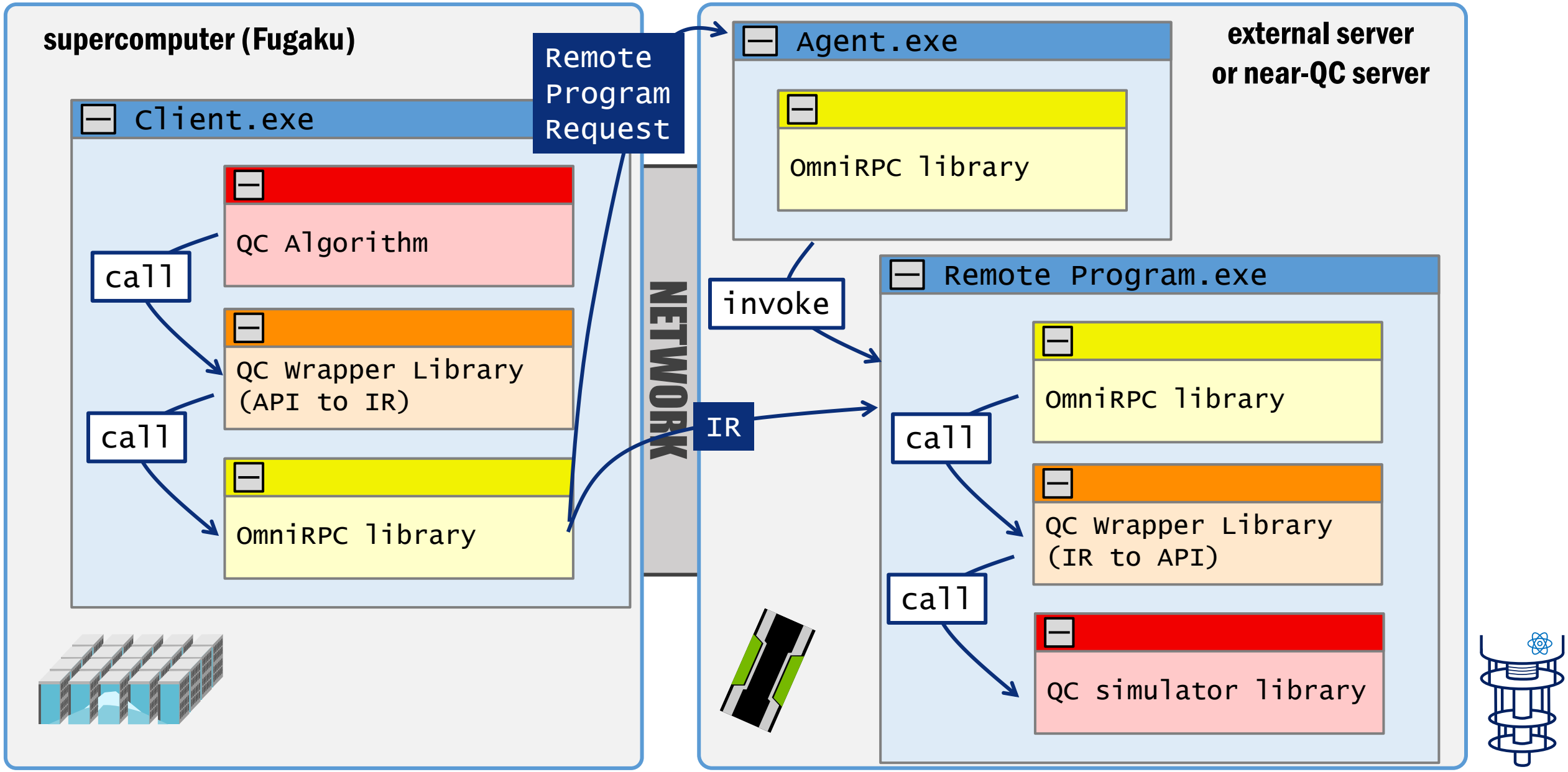
```
extern "C" void qc_measure(){
 int n = qcs_info->ngates;
 for(int i=0; i<n; i++){
    void (*f)(gate_info*) = gate_func[info->gate[i].id];
    f(&(info->gate[i])); //call a specific simulator
                         //function
```

Inside of RPC-based 量子HPCプログラム

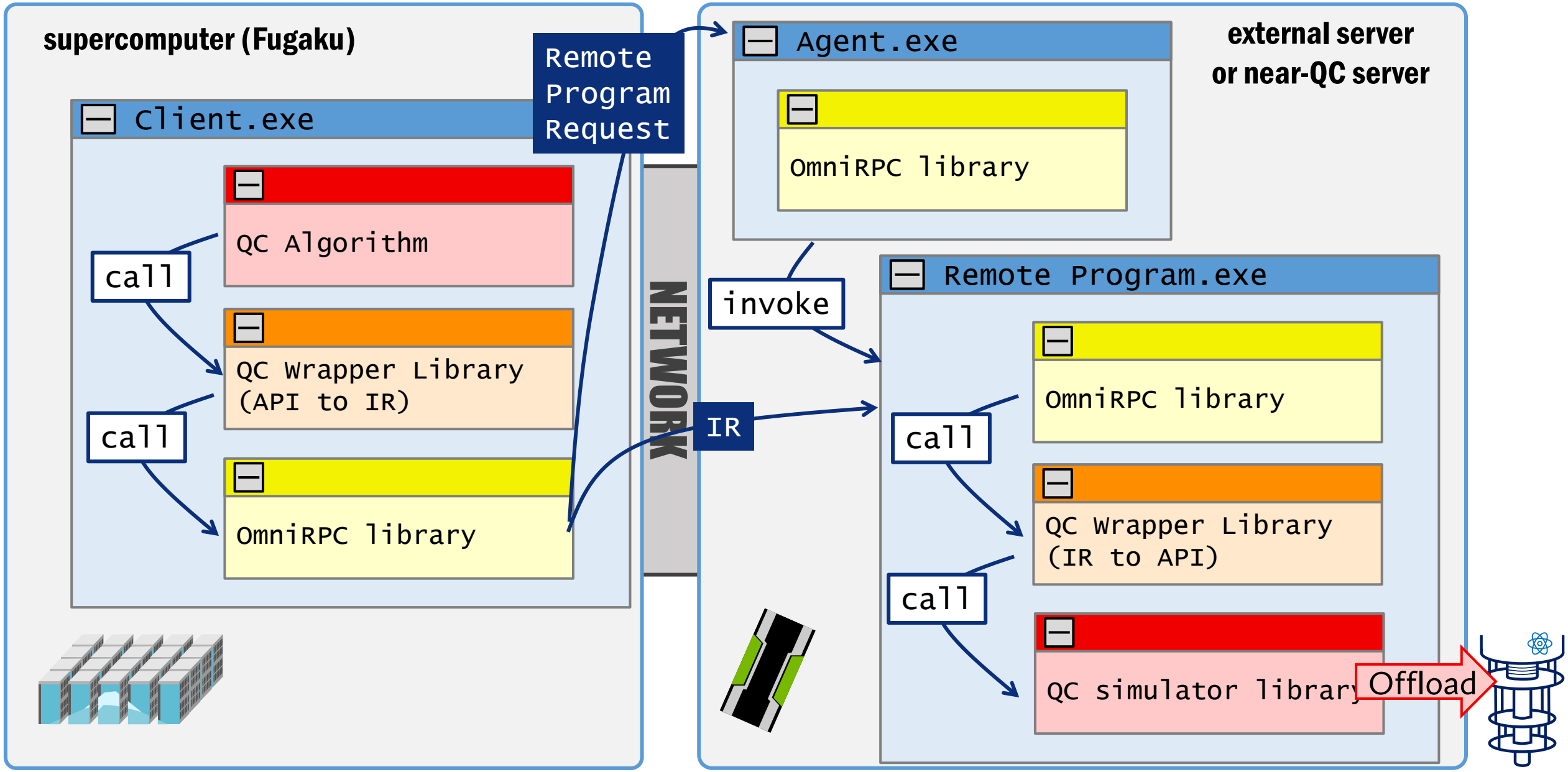
- 量子サイド (NearQC Server) のリモートプログラムは，単一のプログラムがマスタのリクエストに応じて異なるアルゴリズムを実行可能
  ※特定の量子アルゴリズムを実行する特定のリモートプログラムを事前に作成しておくことも可能．
  ゲートリストの受送信のオーバヘッドを軽減できる
- 量子アルゴリズムを量子シミュレータにオフロードする場合，異なるシミュレータが同じコードから利用可能



量子HPCのためのオフローディングプログラミングの概要



量子HPCのためのオフローディングプログラミングの概要

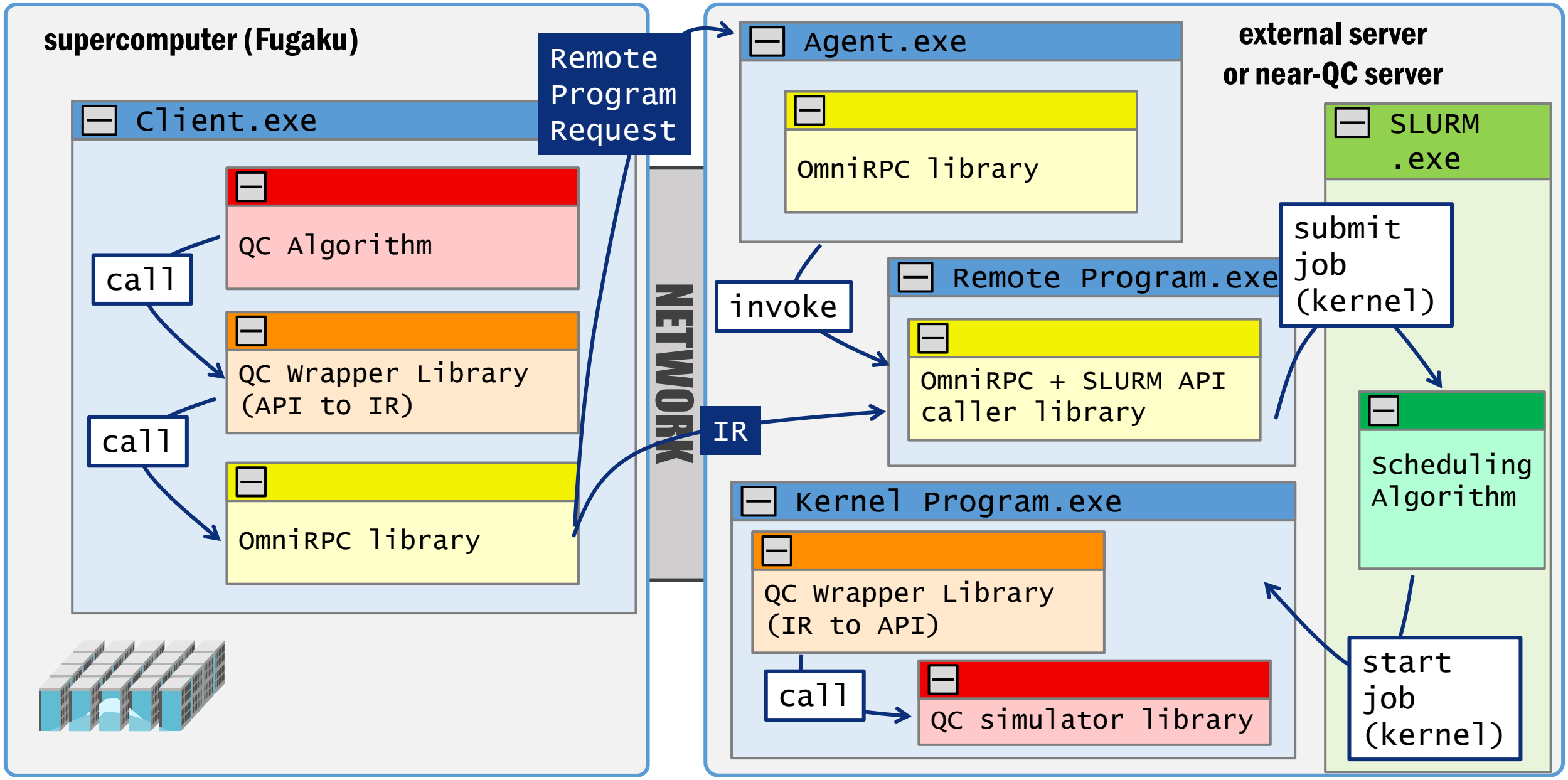


量子HPCのためのスケジューラ+RPC

- 複数のアプリケーション，複数のタスクが，同時に量子計算機にリクエストを出す可能性
- 量子計算機は，いまのところ，希少で貴重な計算資源
  - RIKEN has 150,000+ CPUs and a few QPUs

- 予備評価および調査のためにジョブスケジューラと連携できるようにRPCを拡張し，我々のプログラミングモデルに実装した
  - slurm base

量子HPCのためのスケジューラ+RPC



量子HPCのためのスケジューラ+RPC

- 複数のアプリケーション，複数のタスクが，同時に量子計算機にリクエストを出す可能性
- 量子計算機は，いまのところ，希少で貴重な計算資源
  - RIKEN has 150,000+ CPUs and a few QPUs

- 予備評価および調査のためにジョブスケジューラと連携できるようにRPCを拡張し，我々のプログラミングモデルに実装した
  - slurm base
  - 富岳，あるいは他のスーパーコンピュータ，のジョブスケジューラとのコスケジューリングについては考慮されていない（今後の課題）
  - 2ノードのGPUシステムで試験運用
  - とりあえずやってみることが大切

まとめと今後の課題

- 量子HPCのためのオフローディングに基づくタスクベース・プログラミングモデル
  - 遠隔手続き呼び出しを用いる (RPC)
  - ジョブスケジューラと連携できるようにRPCを拡張
- 今後の課題
  - スーパーコンピュータから外部システムへのRPC
    - security, certification, etc..
  - コスケジューラ