### The problem

- **Simulation**
  - P2P Simulations
    - For early prototyping and feasibility checks
    - Fast development & test cycle
  - Live deployment
    - Precise system evaluation, but
    - Large development effort, challenging to debug
- **Live deployment**
  - Transitioning from simulation to live deployment typically involves a complete application rewrite

### What is ProtoPeer?

- Message passing framework in Java
  - Message serialization
  - Message queuing – full control over the queues
- Extensible event-driven architecture
- Primitives for building peer-to-peer systems
  - Overlays, overlay routing, bootstrapping
- Tools for system evaluation
  - Event injection & scenarios
  - Measurement API
- Performance:
  - Scalable: simulation scales to tens of thousands of peers, on a regular PC
  - High throughput: TCP/UDP messaging pipeline at 2-10k messages per second

### System evaluation support

- Reliable, repeatable measurements are important
  - Measurements typically instrumented in ad-hoc way
- **In ProtoPeer**: measurement API, automated network-wide measurement aggregation
  - Basic statistics: avg, min, max, percentiles
- Event injection & scenarios
  - Delays from the King dataset
  - PlanetLab loss & delay snapshots
  - Users can plug in their own network models
- Variety of network models for simulation:
  - Delays from the King dataset
  - PlanetLab loss & delay snapshots
  - Users can plug in their own network models

### The goal

Write the application once

- Single JVM, many peers
- Different network models
- LAN
- ModelNet
- PlanetLab

### The solution

**Time and networking abstraction**

- **Application**
- **Networking API**
- **Time API**

Simulated time and networking

- **Network model**
  - TCP
  - UDP
  - Real-time clock
- **Other time and networking implementations**

### Peerlets

- Peerlets are well-defined pieces of message passing functionality plus state
  - Reusability
- The peer is the execution context for the peerlets
  - Clock, network interface
- **init()-start()-stop() lifecycle**
- Peerlets can discover one another via the peer
  - Composability
- Advantage: developers sharing their peerlets with others, peerlet libraries

---

[http://protopeer.epfl.ch](http://protopeer.epfl.ch)