

UMTS-Simulator

Telekommunikation

Liste V

Vortrag: UMTS-Simulator

05.01.2004

Sascha Sadikni

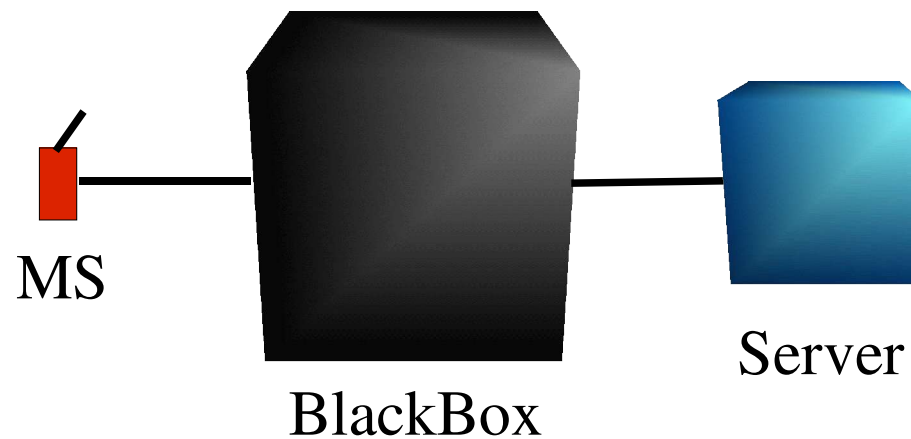
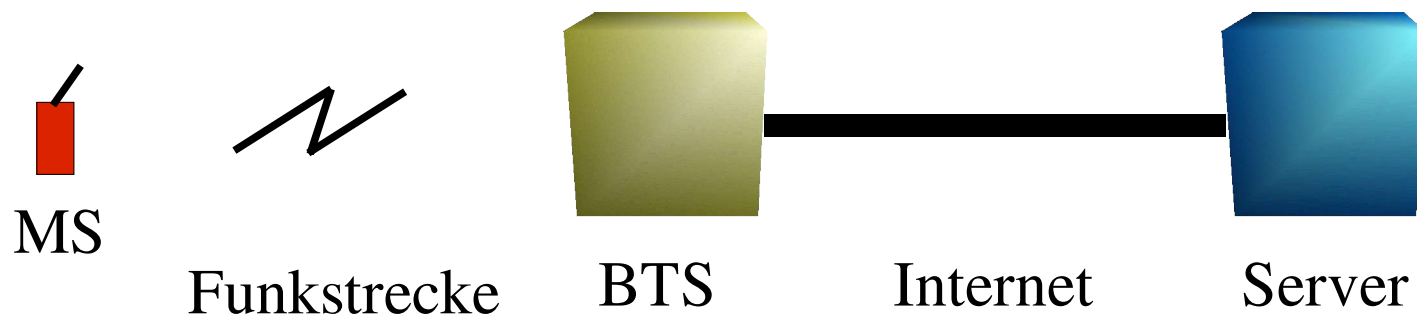
532654

<http://www.informatik.fh-wiesbaden.de/~ssadi001/tk/>

UMTS-Simulator

1. Aufgabenstellung
2. Versuchsaufbau
3. Bridge – Logik
4. 'Quality Of Service' Bewertungsgrößen
5. Wahrscheinlichkeits-Verteilungen
6. Simulator – Logik
7. Native – C (libpcap – read / libnet – write)

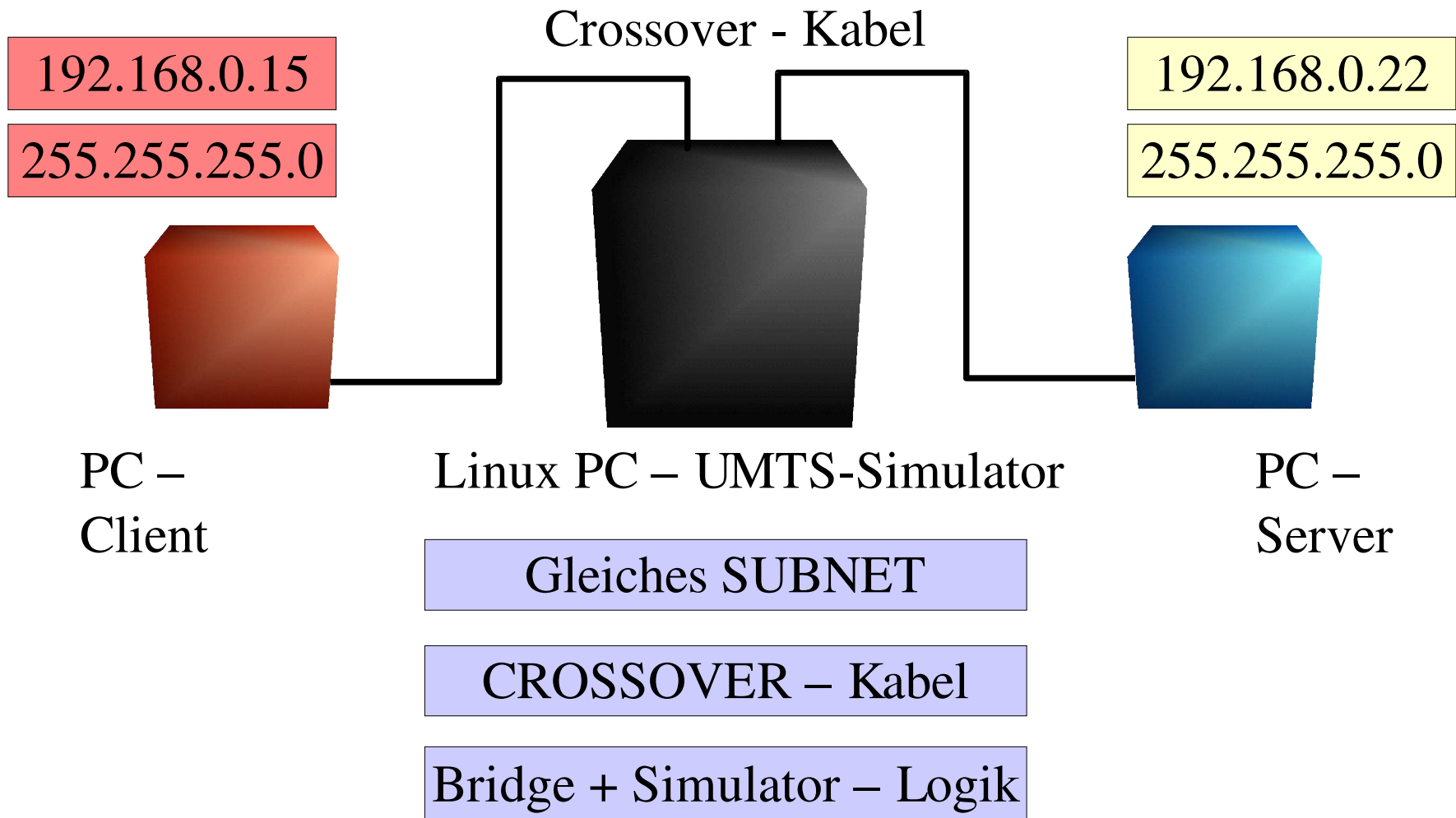
1. Aufgabenstellung



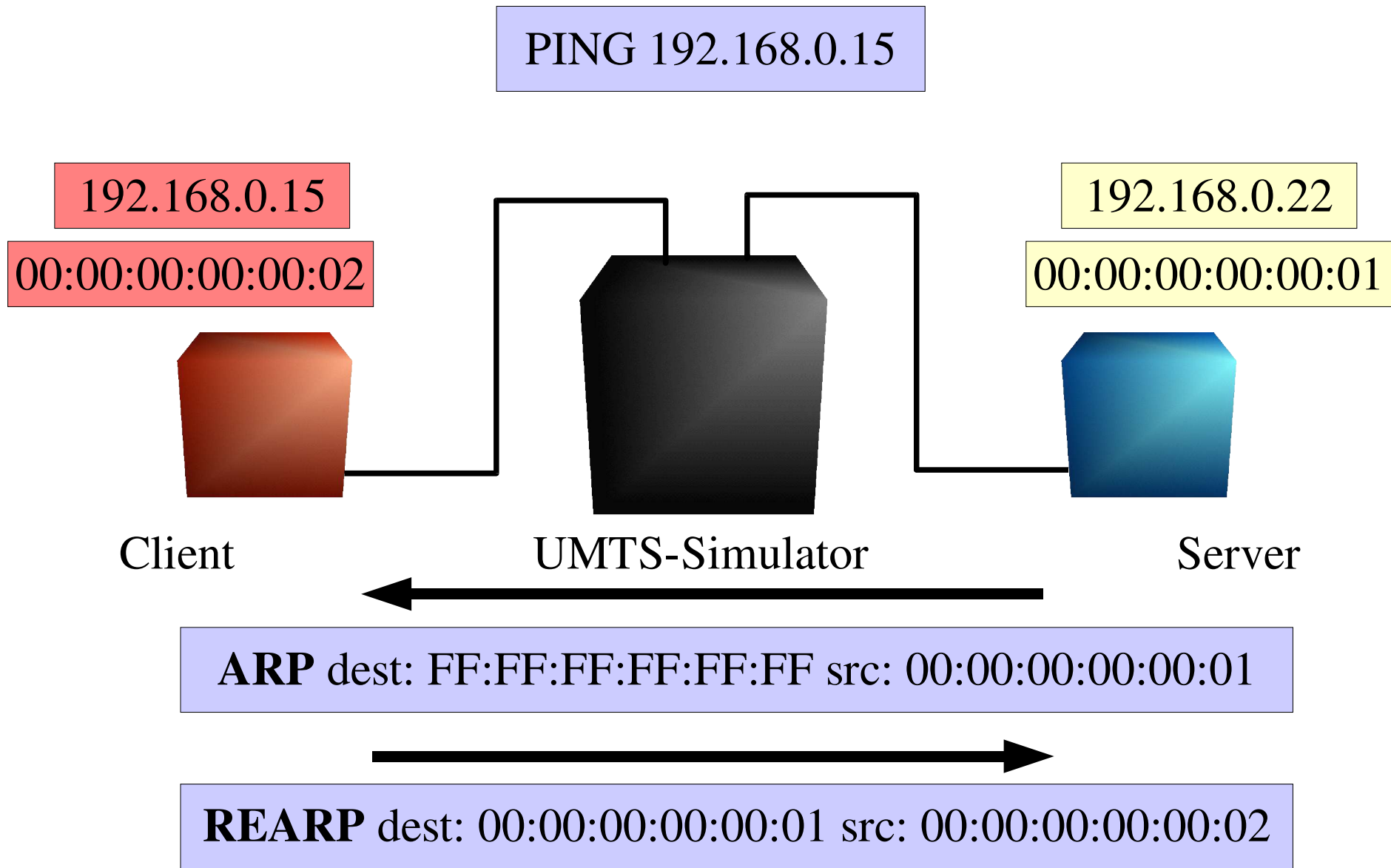
Mobile Station

Base Tranceiver Station

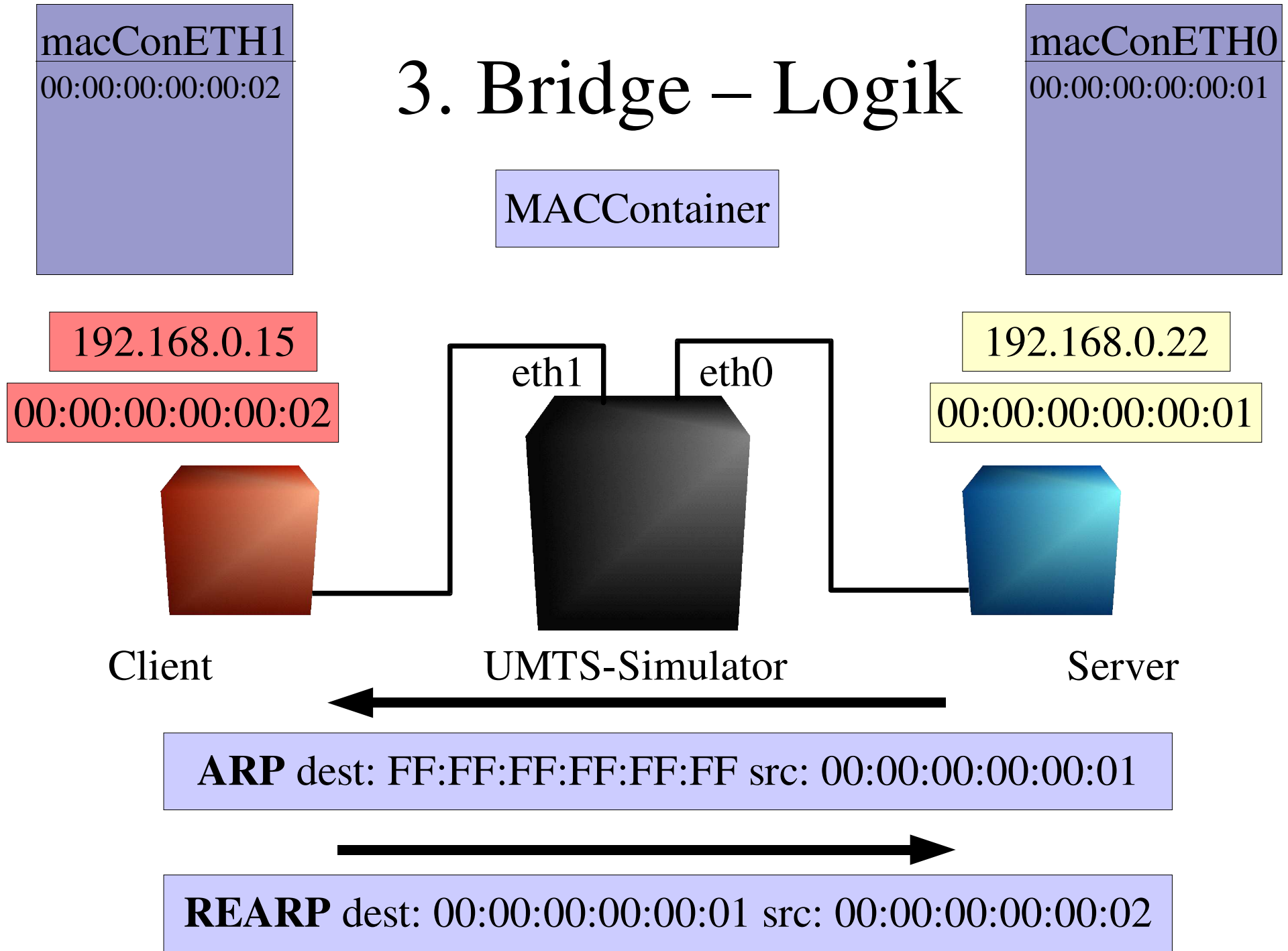
2. Versuchsaufbau



3. Bridge – Logik



3. Bridge – Logik

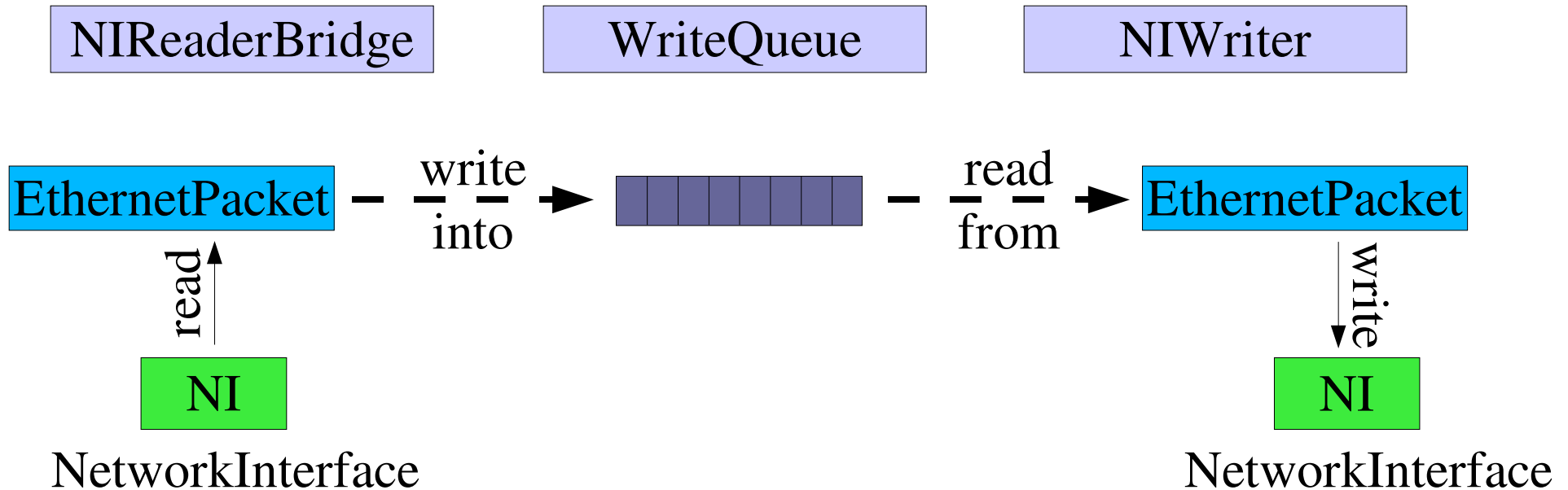


3. Bridge – Logik

NIRedReaderBridge – Thread:

```
// macConETH1 and macConETH0
// should be initialized
while(true) {
    packet = readPacket();
    if(!macConWrite.containsMAC(packet.srcMAC))
    {
        macConRead.addMAC(packet.srcMAC);
        queueWrite.addPacket(packet);
    }
}
```

3. Bridge – Logik



liest von einer
Netzwerk-
Schnittstelle
in eine Queue

schreibt auf einer
Netzwerk-
Schnittstelle
von einer Queue

4. *QoS* – Bewertungsgrößen

1. Block Loss Rate – BLR

Prozentzahl der verlorengegangenen Pakete

2. Block Delay Spread – BDS

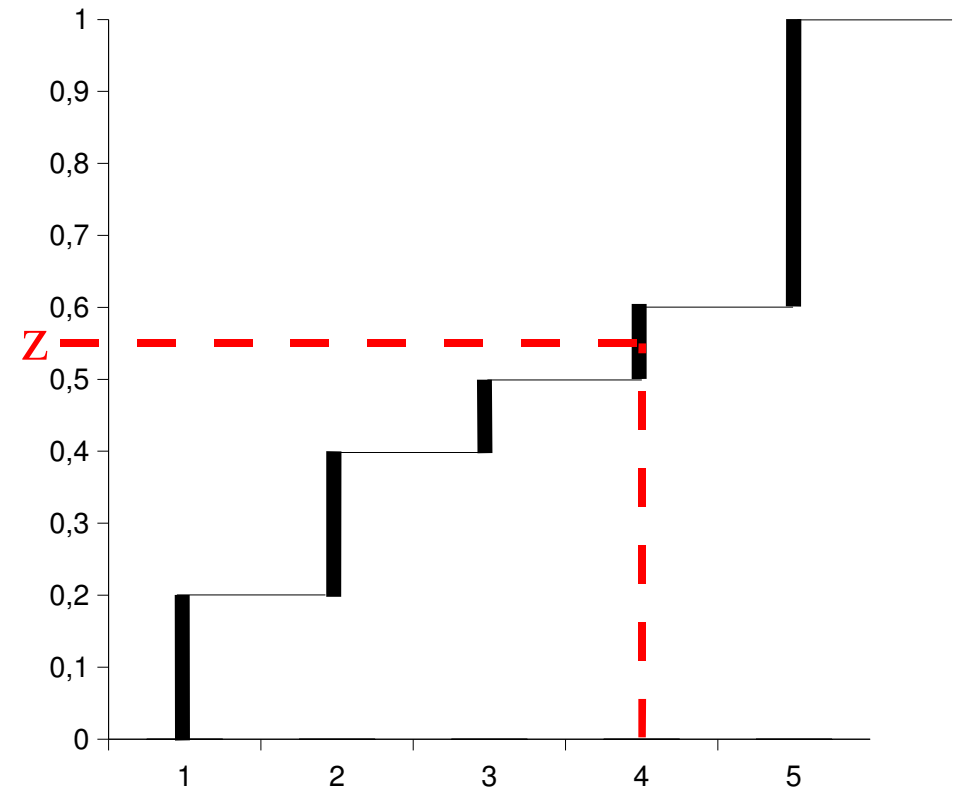
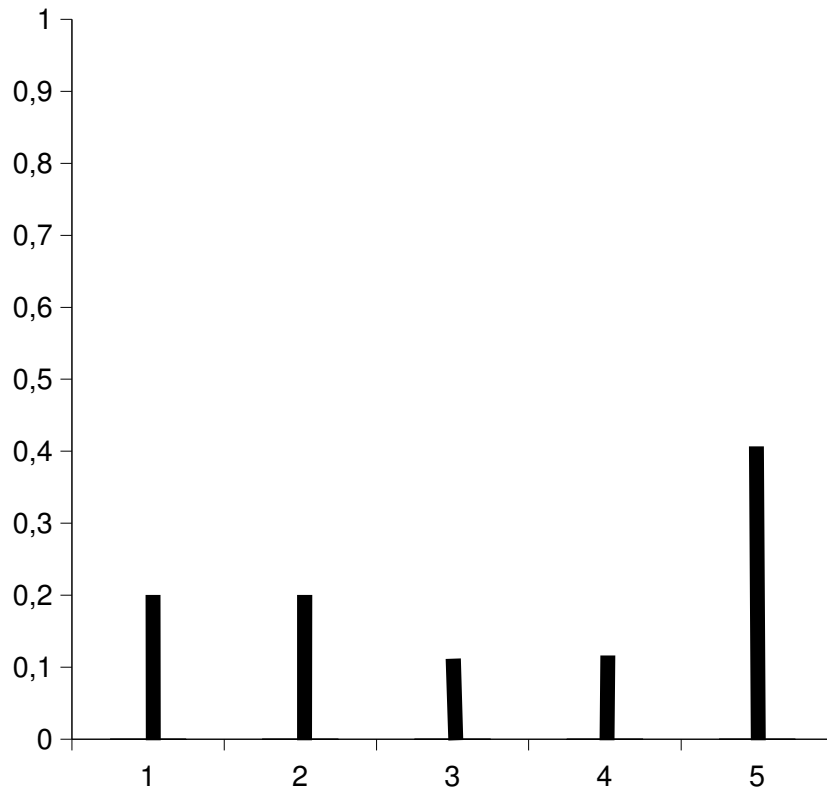
Abstand von zwei aufeinanderfolgenden
Paketten (beim Empfänger) in Millisekunden

3. Block Out of Sequence-Weite – BOSW

Pakete kommen in falscher Reihenfolge an

5. Wahrscheinlichkeits-Verteilungen

Diskrete Verteilung



5. Wahrscheinlichkeits-Verteilungen

Gauss Verteilung

Standard – Normal – Verteilung:

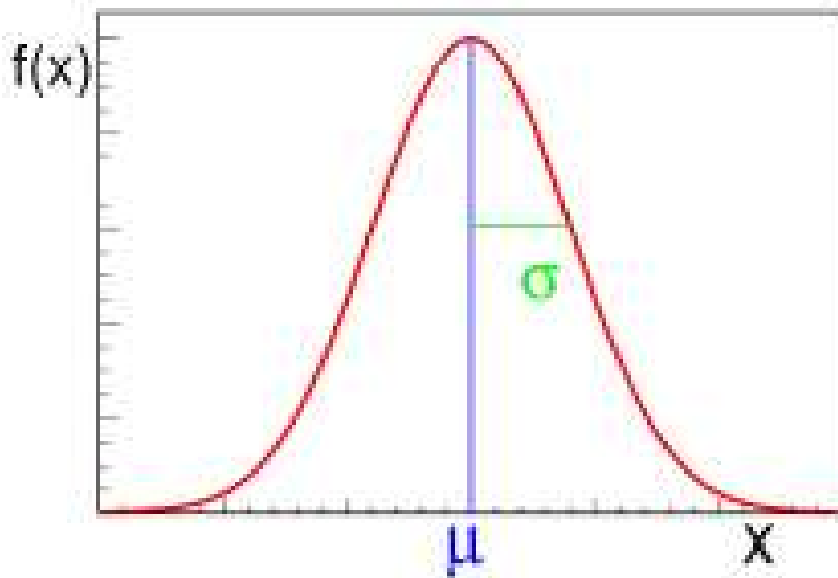
$\mu = 0$ - Mittelwert

$\sigma = 1$ - Standardabweichung

$$\sigma^2 = \Sigma [(x_i - \mu)^2] / n$$

$z = \text{Random.nextGaussian}();$

$$x = z * \sigma + \mu$$



6. Simulations – Logik

Filter.blrFilter(key):

```
ConfigValues cv=ConfigValues.getInstance(key);
double randBLR =randomBLR.nextDouble();

int pBLR = cv.getActBLR();

// *** if pBLR==-1(error, cv not init)
// *** => return true (no filter)
if(randBLR>(((double)pBLR)/100.0))return true;
return false;
```

6. Simulations – Logik

Filter.bdsFilter(key):

```
ConfigValues cv=ConfigValues.getInstance(key);
```

```
long bdsAvg      = cv.getActBDSAvg();
```

```
long bdsSigma   = cv.getActBDSSigma();
```

```
// *** normal-distribution
```

```
z = randomBDS.nextGaussian();
```

```
x = (z * bdsSigma) + bdsAvg;
```

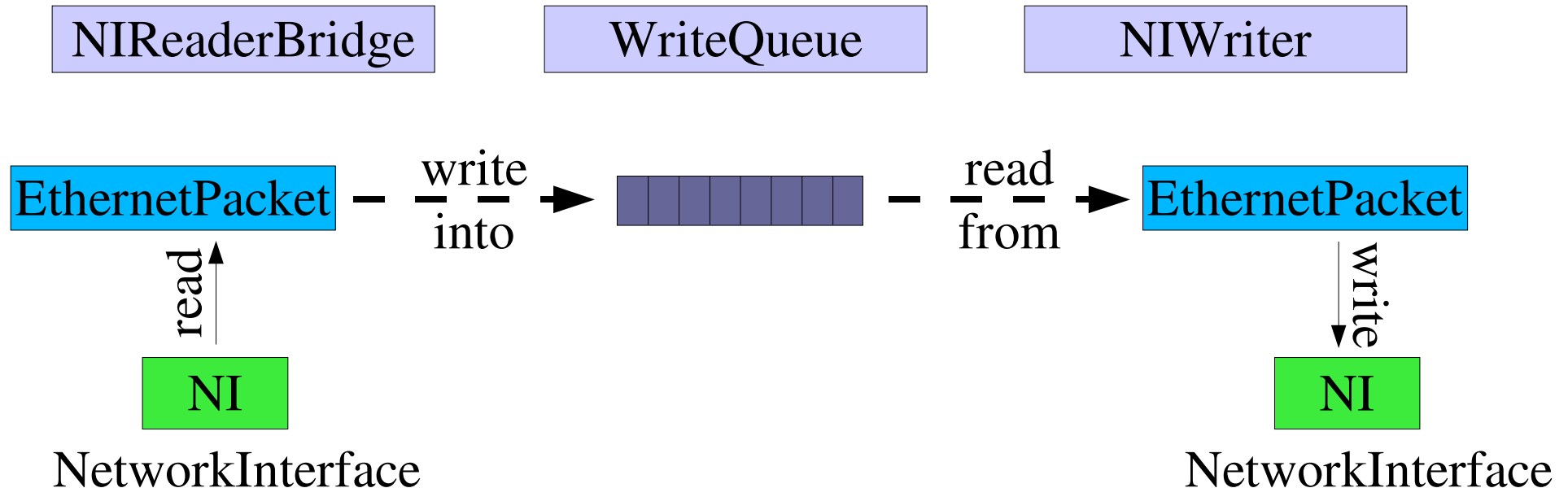
```
return x;
```

6. Simulations – Logik

Filter.osFilter(key):

```
ConfigValues cv=ConfigValues.getInstance(key);
double randOS = randomOS.nextDouble();
int pOS      = cv.getActOS();
int osAvg    = cv.getActOSAvg();
int osSigma  = cv.getActOSSigma();
// *** if pOS=-1(error,cv not init)=>no filter
if( randOS < (((double)pOS)/100.0) ) {
    // *** normal-distribution
    double z = randomOS.nextGaussian();
    double x = (z * osSigma) + osAvg;
    return x;
}
// *** else: no out of sequence
return 0;
```

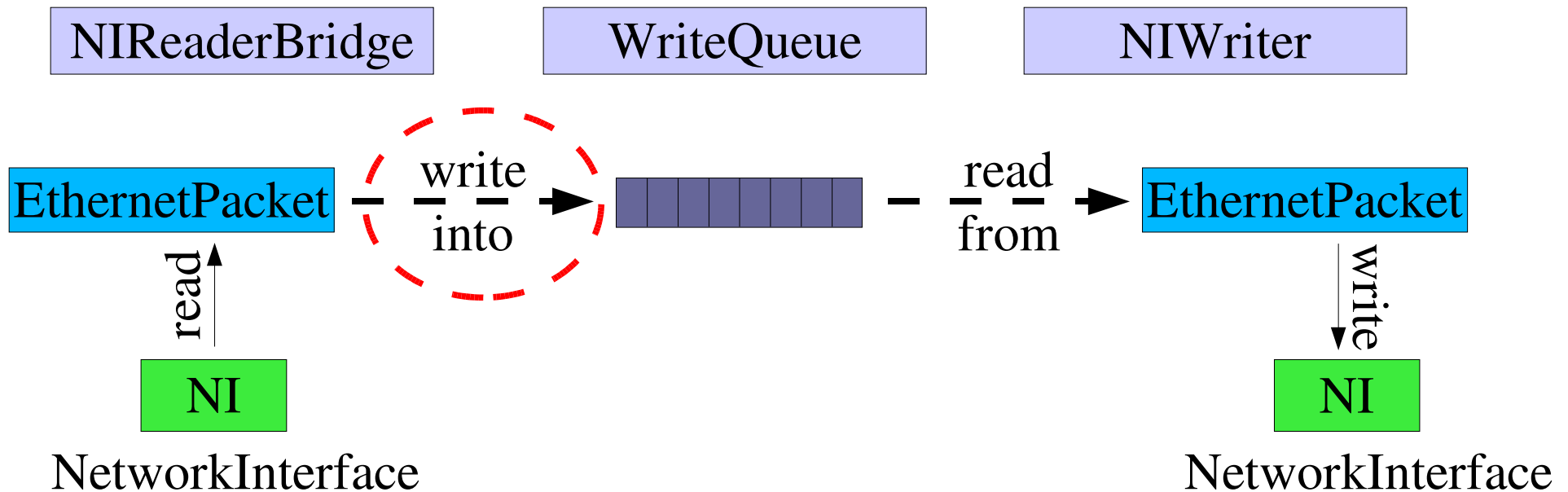
6. Simulations – Logik



liest von einer
Netzwerk-
Schnittstelle
in eine Queue

schreibt auf einer
Netzwerk-
Schnittstelle
von einer Queue

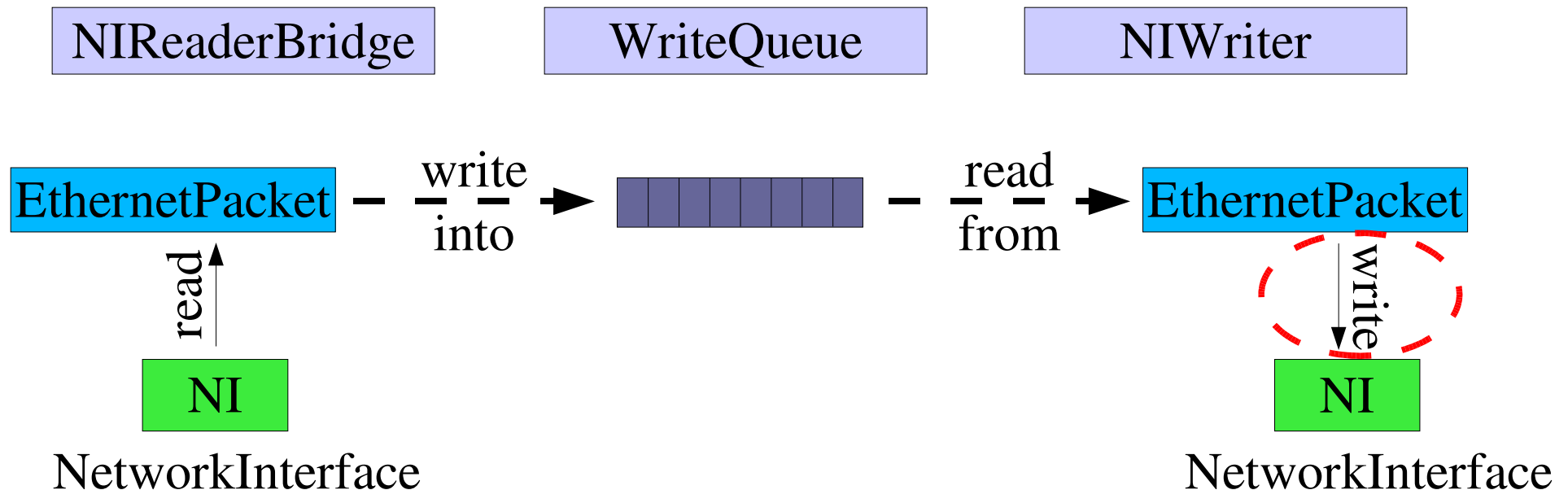
6. Simulations – Logik



BLR Filter:

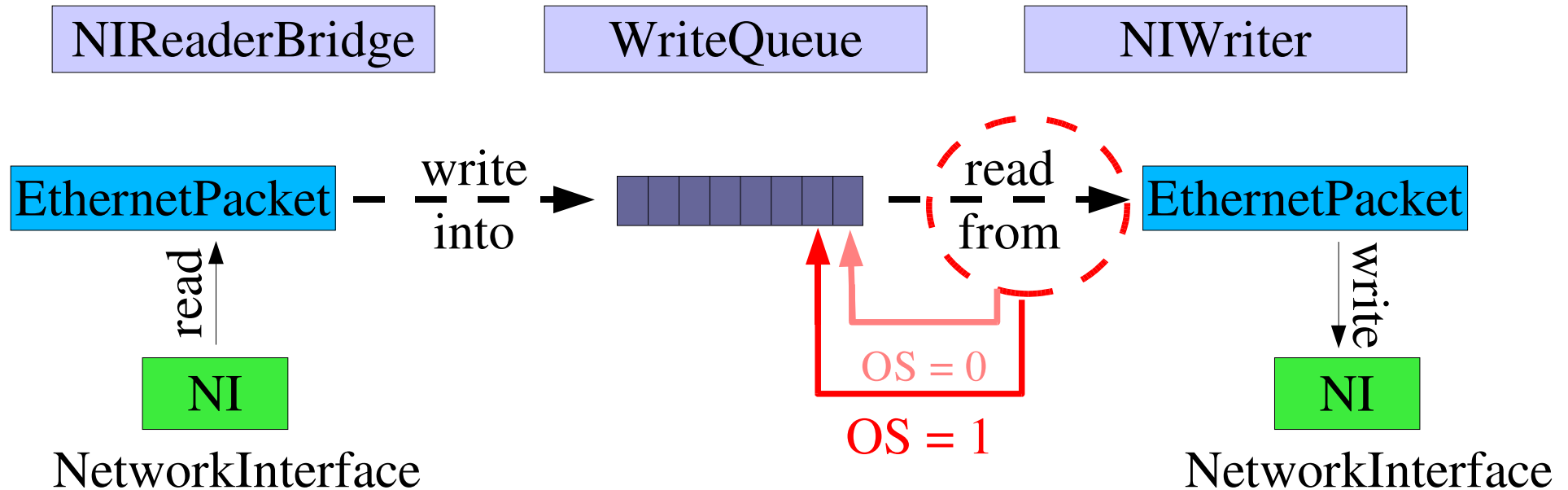
```
if(Filter.blrFilter() == true) {  
    writeIntoQueue(EthernetPacket);  
}
```

6. Simulations – Logik



```
BDS Filter:  
readFromQueue();  
sleep( Filter.bdsFilter() );  
writeOnNI(EthernetPacket);
```

6. Simulations – Logik



BOSW Filter:
`Queue.get(Filter.osFilter());`

6. Simulations – Logik

NIReaderBridge – Thread:

```
// macConETH1 and macConETH0
// should be initialized
while(true) {
    packet = ni.readPacket();
    if(!macConWrite.containsMAC(packet.srcMAC) {
        macConRead.addMAC(packet.srcMAC);
        if(Filter.blrFilter(devRead)) {
            queueWrite.addPacket(packet,
                Filter.osFilter(devRead),
                Filter.bdsFilter(devRead));
        }
    }
}
```

6. Simulations – Logik

NIWriter – Thread:

```
while(true) {  
    packetInfo = queue.getPacket();  
    delay      = packetInfo.getDelay();  
    packet     = packetInfo.getPacket();  
  
    this.sleep(delay);  
  
    ni.writePacket(packet);  
}
```

6. Simulations – Logik

WriteQueue – getPacket():

```
// *** try to find the first packet with os==0
for(i=0;i<packetInfos.size();i++) {
    pi = packetInfos.get(i);
    if(pi.getOSValue()<=0) {
        packetInfos.removeElementAt(0); break;
    }
    else {
        pi.decrOSValue();
        pi=null;
    }
}
return pi;
```

7. Native – C

```
// *** init the LIBPCAP:  
pcap_t *pcap_descr = pcap_open_live(dev_read,  
                                     BUFSIZ,  
                                     promisc,  
                                     timeout (-1),  
                                     pcap_errbuf);  
  
// *** close LIBPCAP:  
pcap_close(pcap_descr);
```

7. Native – C

```
// *** read from device with LIBPCAP:

struct pcap_pkthdr pcap_packetHeader;
const u_char *packet=pcap_next(pcap_descr,
                               &pcap_packetHeader);

// *** error-handling
pcap_geterr(pcap_descr);

// *** return captured length
return pcap_packetHeader.caplen;
```

7. Native – C

```
// *** init the LIBNET:  
libnet_t *libnet_descr=libnet_init(LIBNET_LINK_ADV,  
                                   dev_write,  
                                   libnet_errbuf);  
  
// *** close LIBNET:  
libnet_destroy(libnet_descr);
```

7. Native – C

```
// *** write to device with LIBNET:

write_status = libnet_adv_write_link(libnet_descr,
                                     packet,
                                     len);

// *** error-handling
libnet_geterror(libnet_descr);

// *** return the count of really written bytes
return write_status;
```

UMTS-Simulator

- Von einem Netzwerk-Interface lesen (LIBPCAP)
- Auf einem Netzwerk-Interface schreiben (LIBNET)
- Bridge – Logik
- Simulation von 'QoS'-Größen
 - BLR – Block Loss Rate
 - BDS – Block Delay Spread
 - BOSW – Block Out of Sequence Weite

Quellen

LIBPCAP

<http://www.tcpdump.org/>

LIBNET

<http://www.packetfactory.net/projects/libnet/>

JAVA

<http://java.sun.com/docs/books/tutorial/index.html>

Verteilungen

<http://www.uni-konstanz.de/FuF/wiwi/heiler/os/vt-index.html>