Modelling Communication Software Execution via Tracing

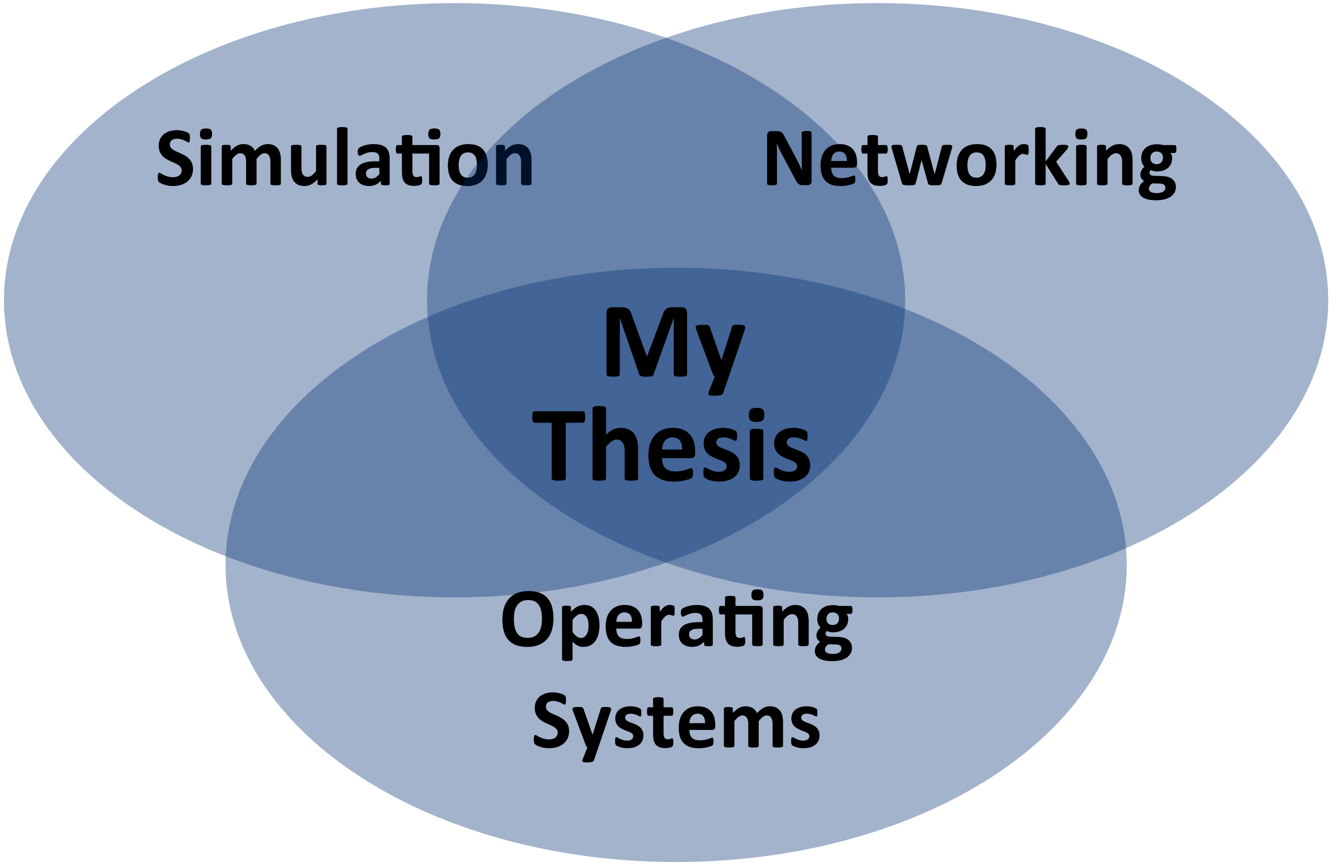
Stein Kristiansen

steikr@ifi.uio.no

Outline

- Research Area
- My PhD-Work
  - Motivation
  - Overview of my work
  - Results
- Possible master thesis areas

Research Area



Simulation

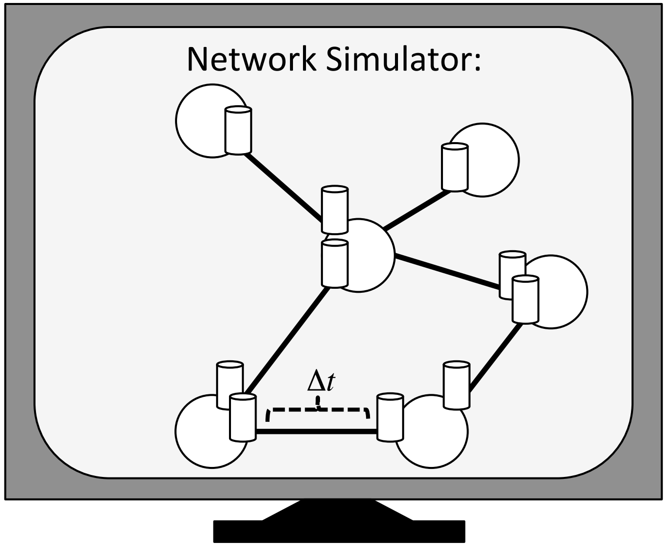
Networking

My Thesis

Operating Systems

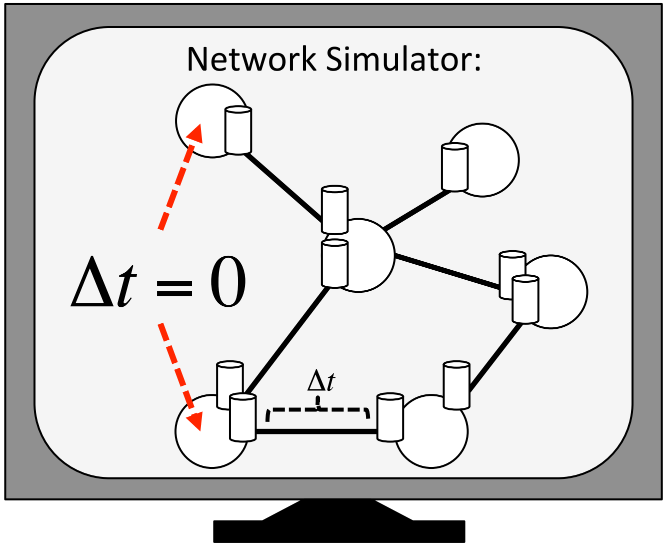
Network Simulation

- Computer systems are often very complex
  - How to test and evaluate?
- Solution: network simulation

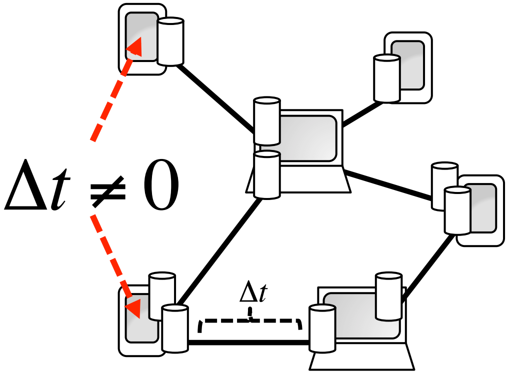


Network Simulation

- Network simulation
  - Impact of communication software execution ignored
  - **Can this always be ignored?**

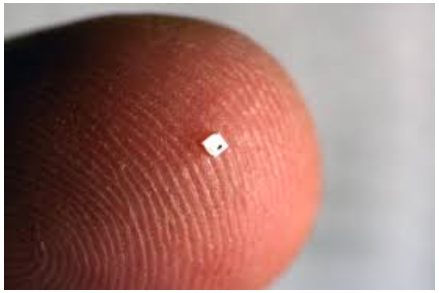


Real World:



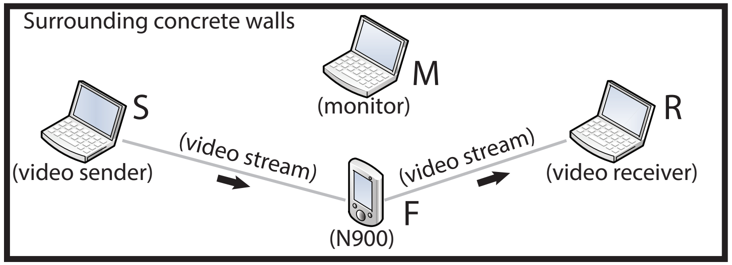
New and emerging networks

- Heterogeneous sets of devices
- In all sizes (many very, very small!)
  - And thus **resource constrained!**

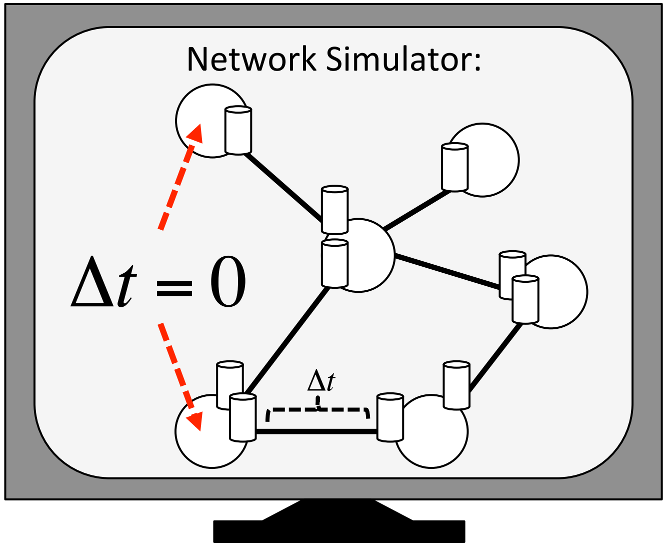


Network Simulation

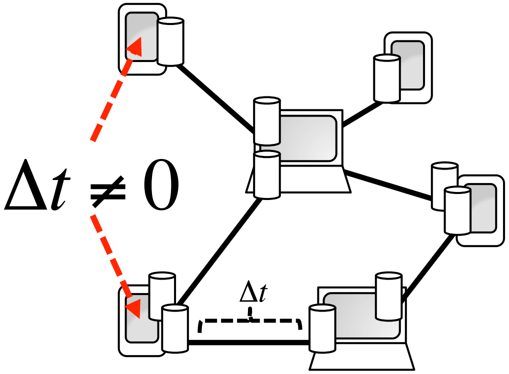
- Network simulation
  - Impact of communication software execution ignored
  - **Can this always be ignored?**
- Test:



- Throughput < 3.7 Mbps
  - WiFi-router: 15Mbps!
- 10-40 ms. already at 2 Mbps
  - Wifi-Router: a few milliseconds!

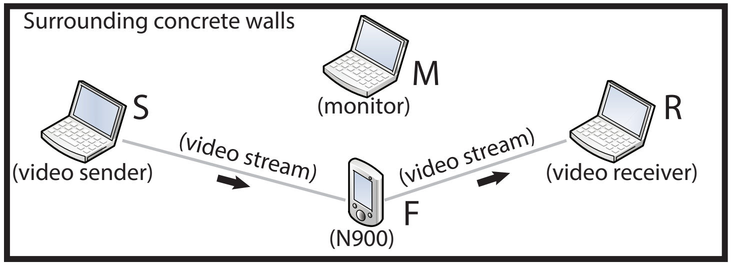


Real World:

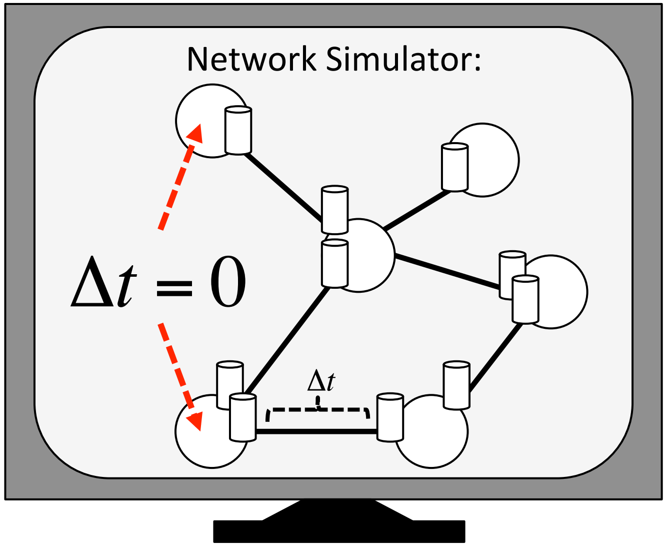


Network Simulation

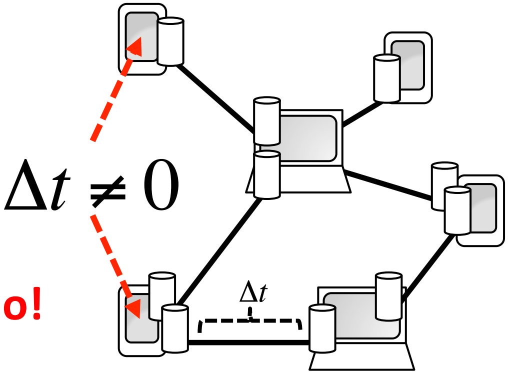
- Network simulation
  - Impact of communication software execution ignored
  - **Can this always be ignored?**
- Test:



- Throughput < 3.7 Mbps
  - WiFi-router: 15Mbps!
- 10-40 ms. already at 2 Mbps
  - Wifi-Router: a few milliseconds!



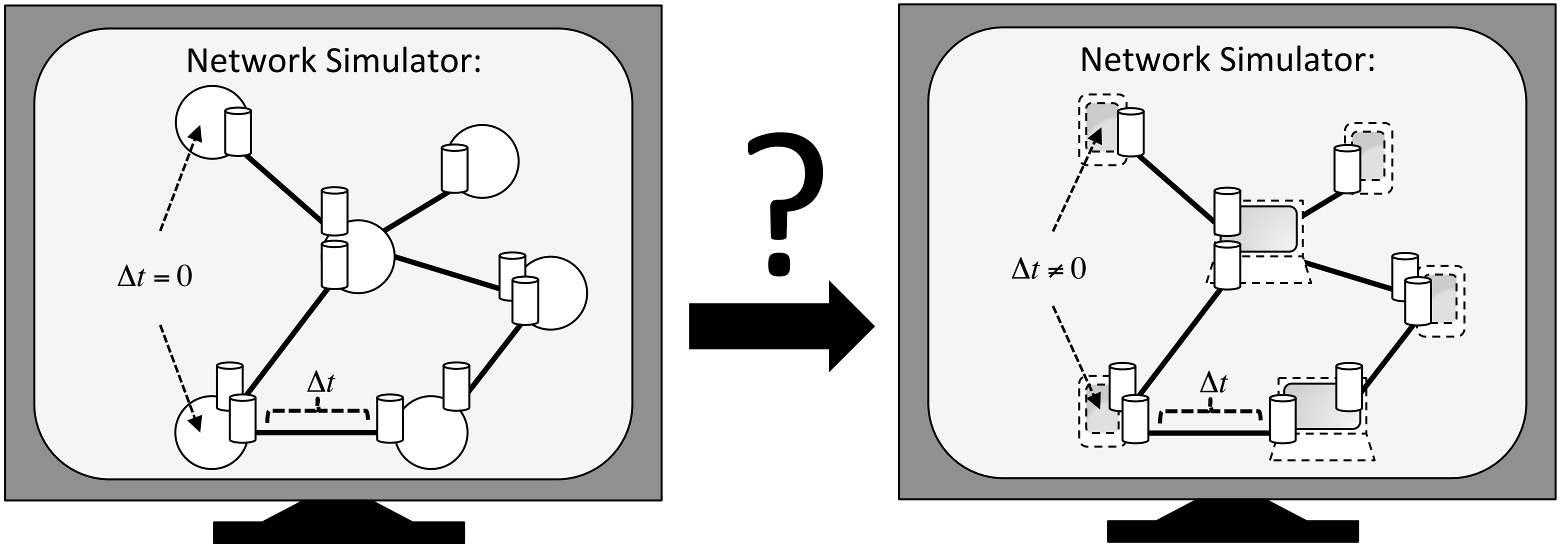
Real World:



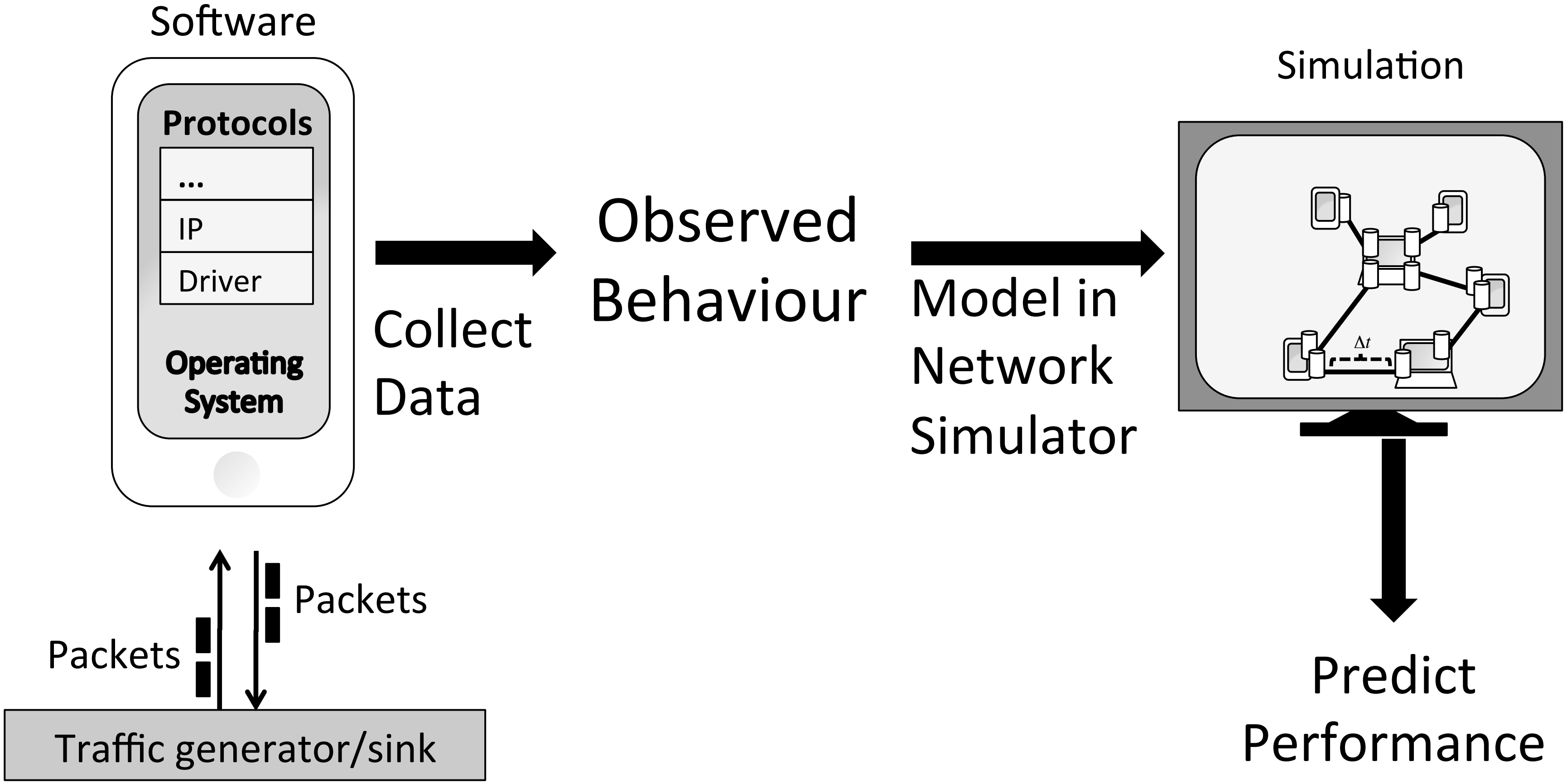
Answer: No!

Problem

How do we make network simulators account for the execution of communication software?



Core Idea



Software

Protocols

...

IP

Driver

Operating System

Packets

Packets

Traffic generator/sink

Collect Data

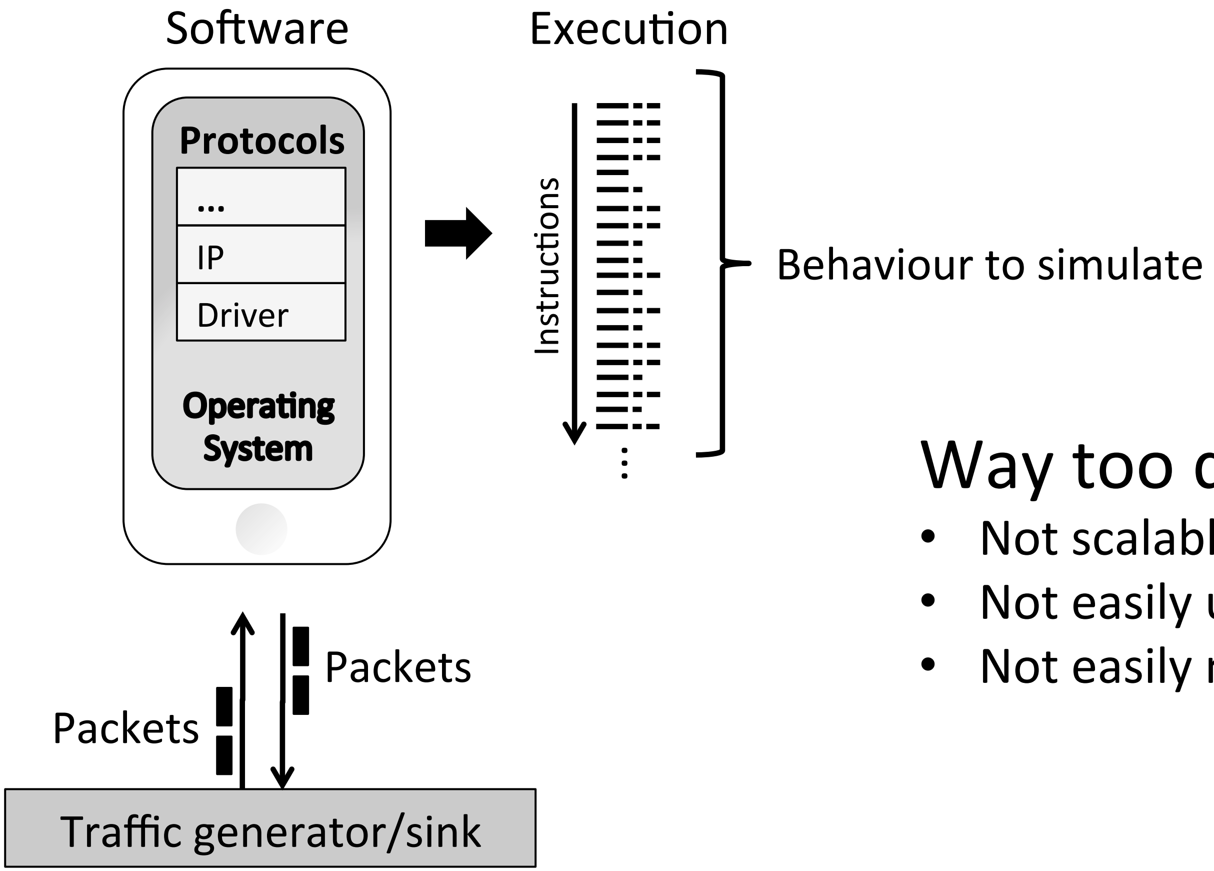
Observed Behaviour

Model in Network Simulator

Simulation

Predict Performance

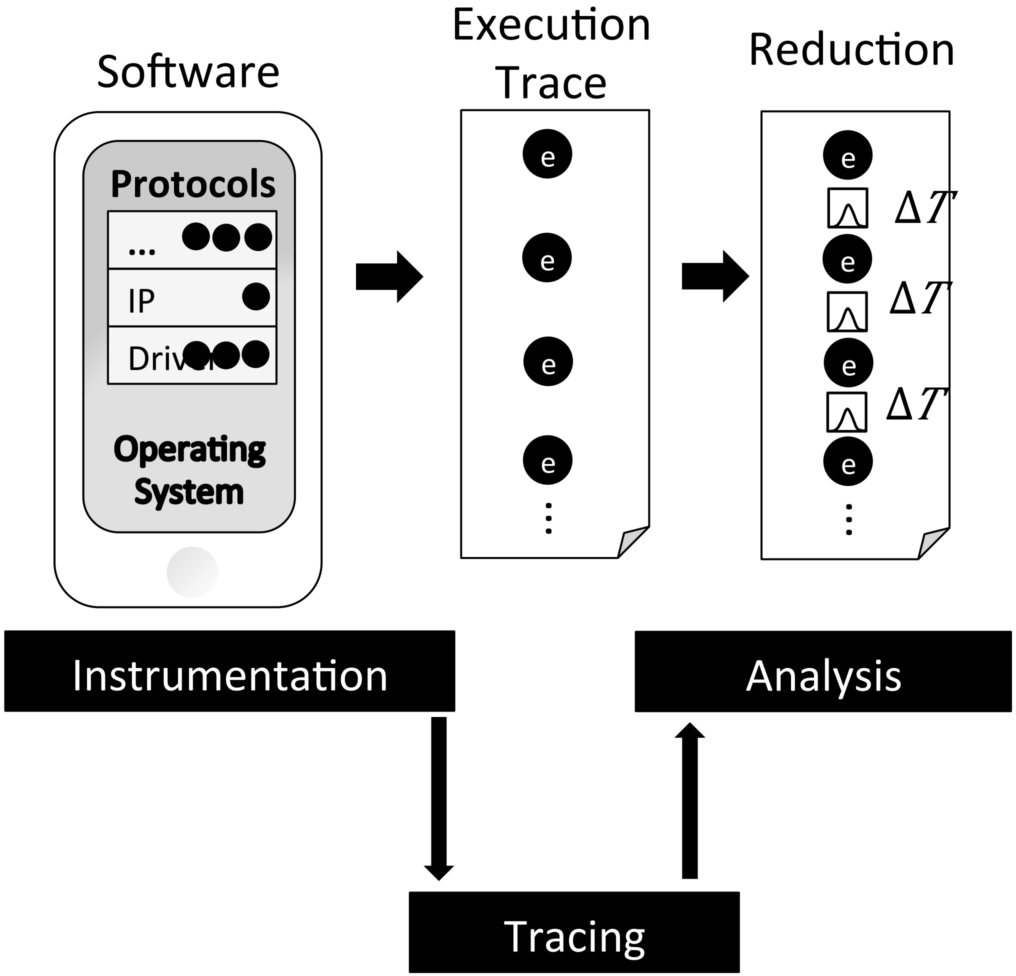
5-Step Approach



Way too detailed!

- Not scalable
- Not easily understandable
- Not easily modifiable

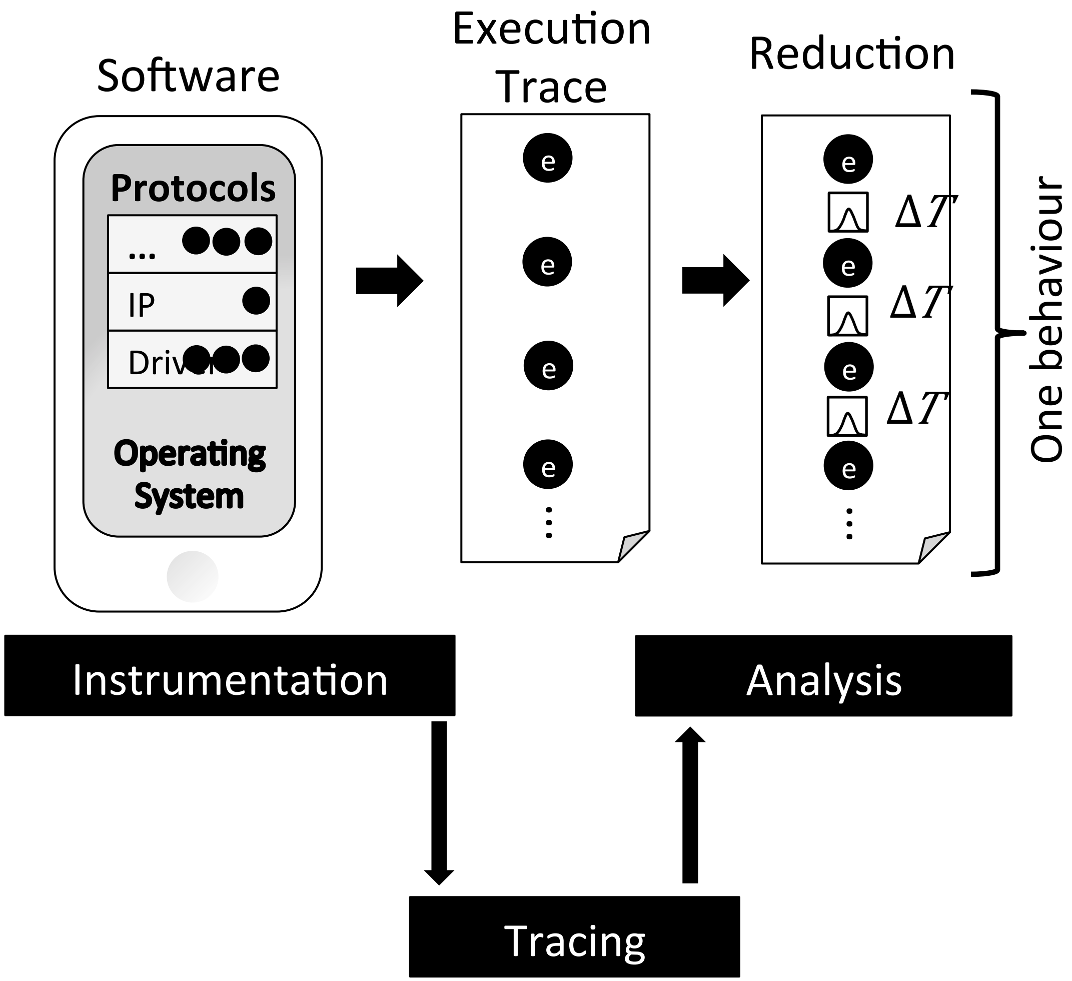
Simplification



Capture important events:

- Queuing
- Synchronization
- Loops
- Interactions with HW
- …

Capturing Change in Behaviour



Behaviour affected by input data,e.g.,:

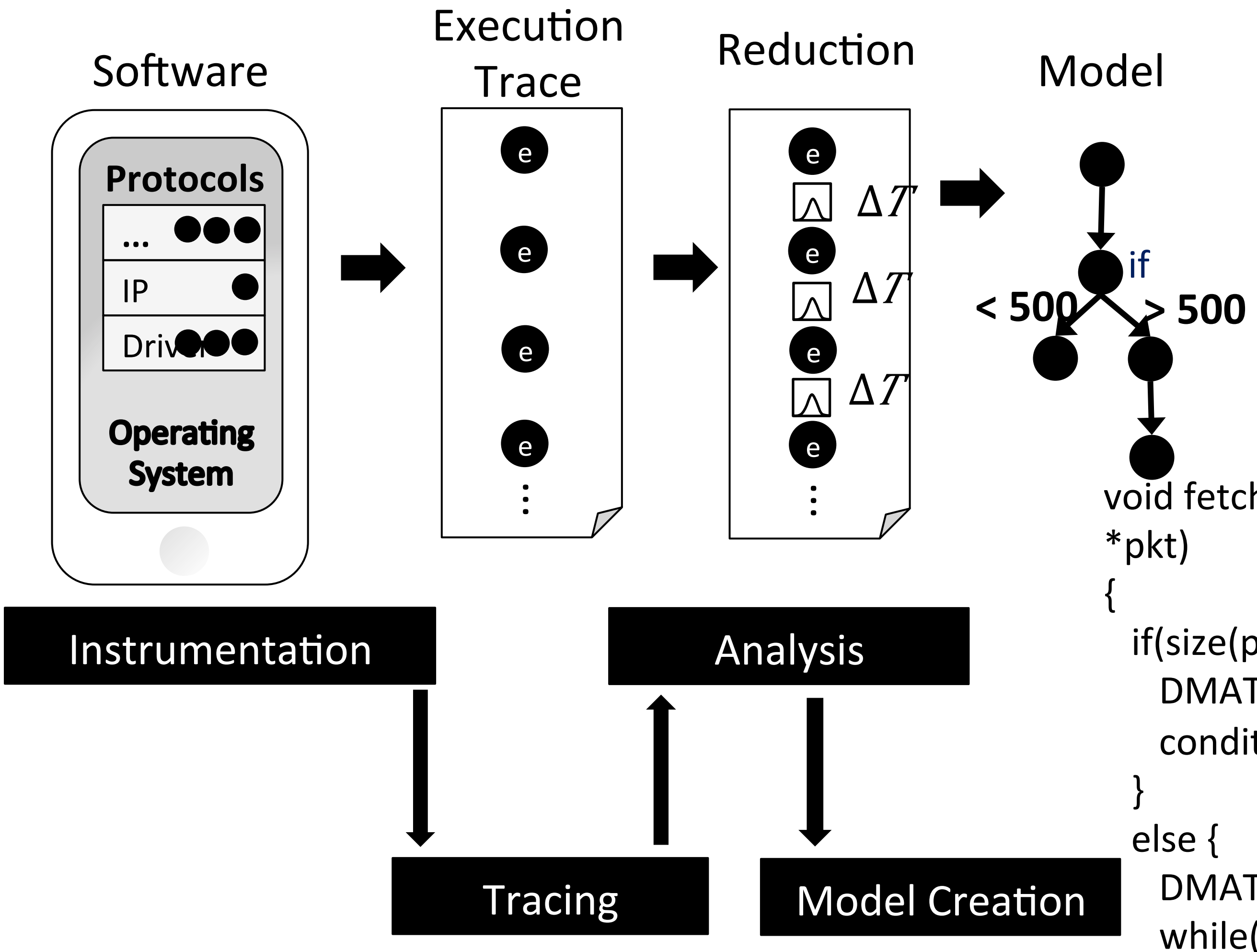
- Packet size, type, …
- Routing table size, …
- Bus state, …

-> Many behaviours!

Example:

```
void fetchFromNIC(Packet
*pkt)
{
  if(size(pkt) > 500) {
    DMATranferNIC(pkt);
    condition_wait(dma);
  }
  else {
    DMATranferNIC (pkt);
    while(!dma_done) {};
  }
}
```

Capturing Change in Behaviour



Software

Execution Trace

Reduction

Model

Instrumentation

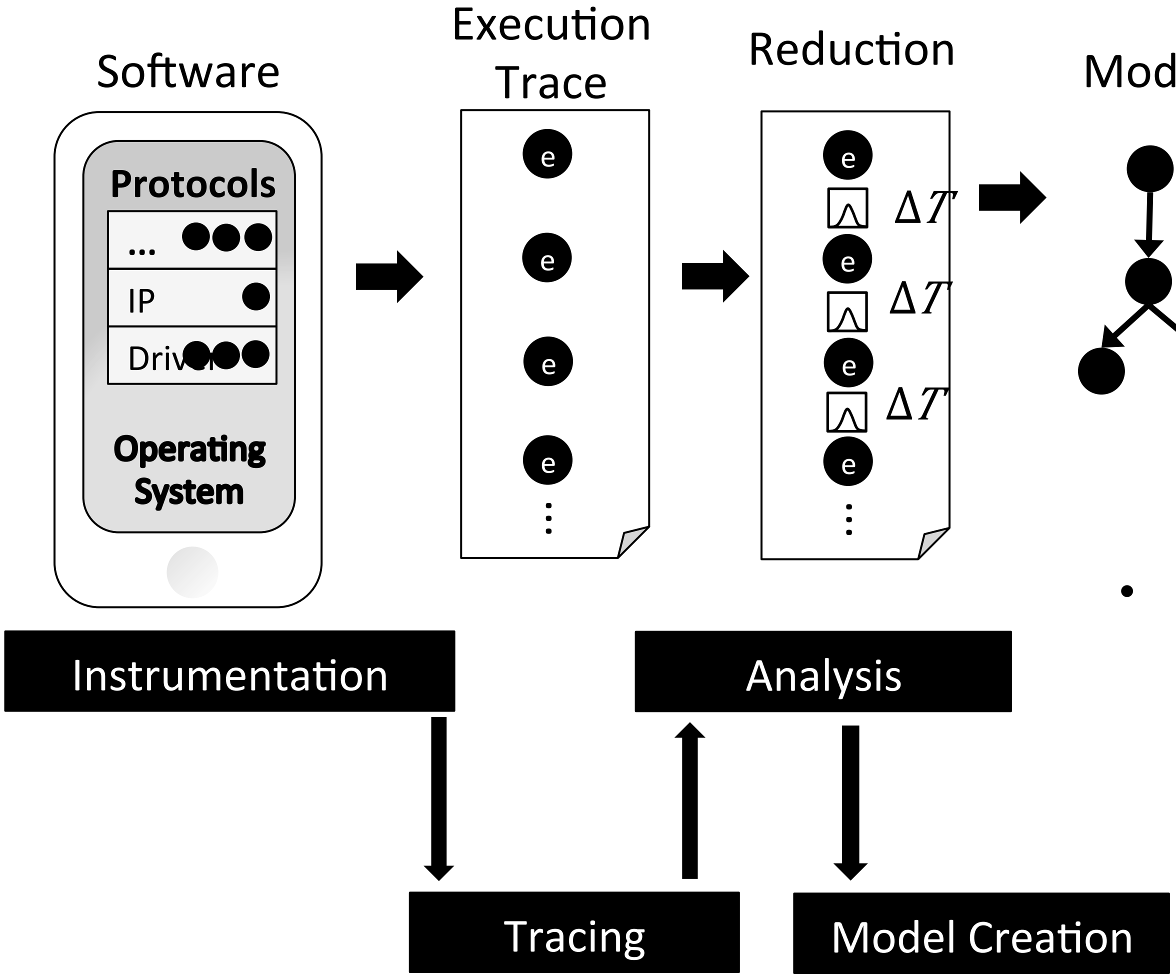
Analysis

Tracing

Model Creation

```
void fetchFromNIC(Packet
*pkt)
{
  if(size(pkt) > 500) {
    DMATranferNIC(pkt);
    condition_wait(dma);
  }
  else {
    DMATranferNIC (pkt);
    while(!dma_done) {};
  }
}
```

Capturing Impact of Multi-Threading



Software

Protocols

...

IP

Driver

Operating System

Execution Trace

e

e

e

e

Reduction

e

ΔT

e

ΔT

e

ΔT

e

Model

Instrumentation

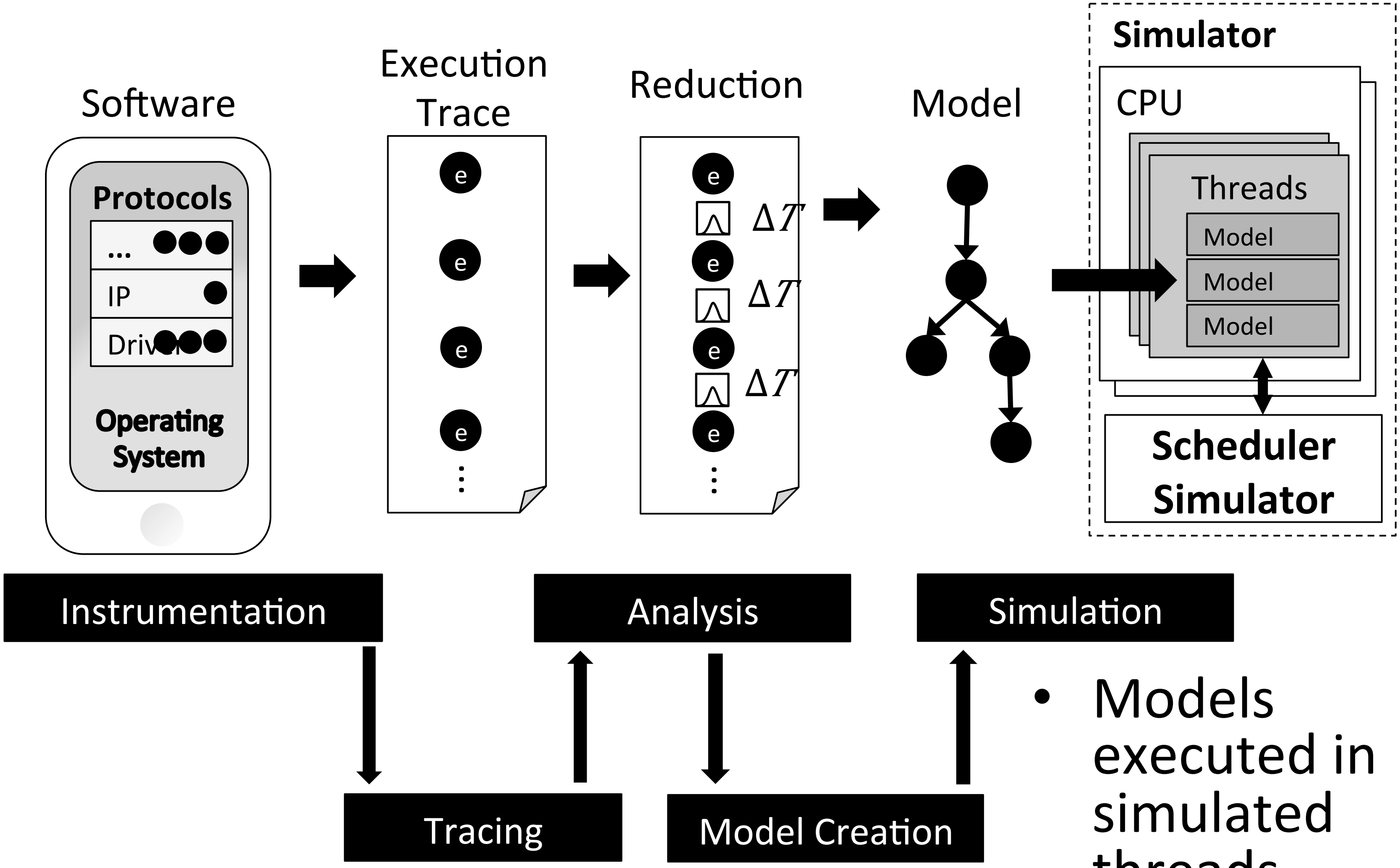
Analysis

Tracing

Model Creation

- Overall performance affected by multi-threading
  - Workload
  - Scheduling policy
  - Synchronization

Simulation



Software

Protocols

...

IP

Driver

Operating System

Execution Trace

e

e

e

e

⋮

Reduction

e

ΔT

e

ΔT

e

ΔT

e

⋮

Model

Simulator

CPU

Threads

Model

Model

Model

Scheduler Simulator

Instrumentation

Analysis

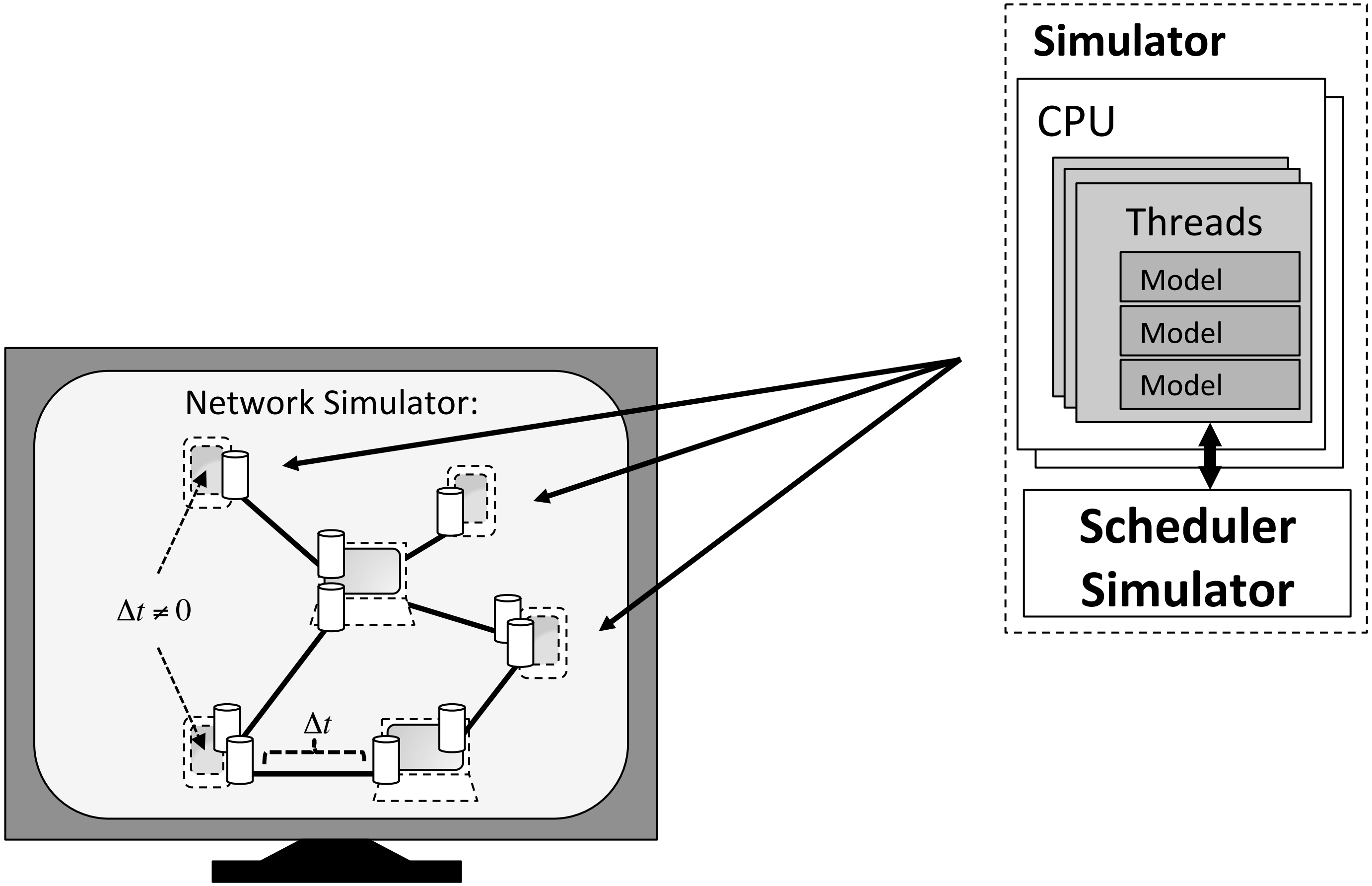
Simulation

Tracing

Model Creation

- Models executed in simulated threads

Simulation



Simulator

CPU

Threads

Model

Model

Model

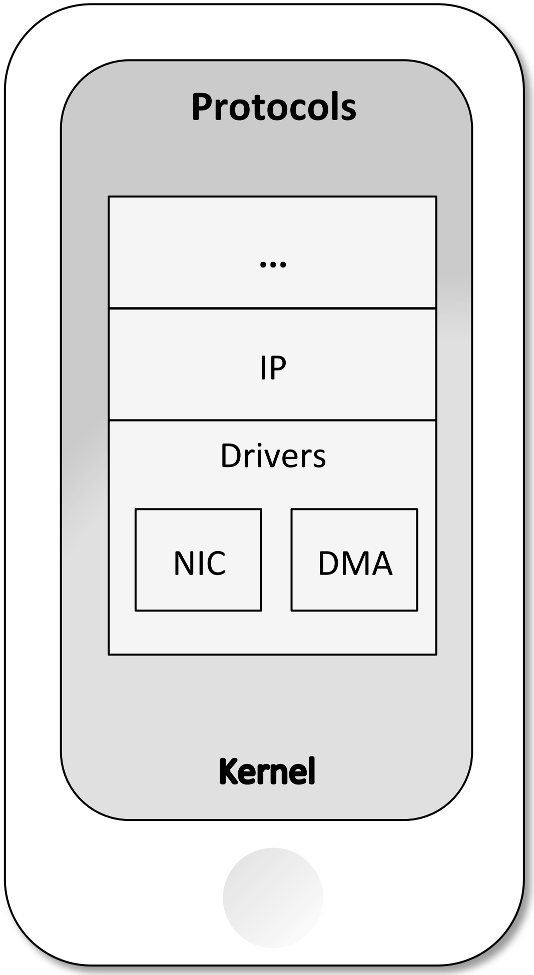
Scheduler Simulator

Network Simulator:

$\Delta t \neq 0$

Δt

Google Nexus One: Instrumentation



Instrumentation:

Networking sub-system, including IP
10 tracepoints

The rest of kernel
58 tracepoints

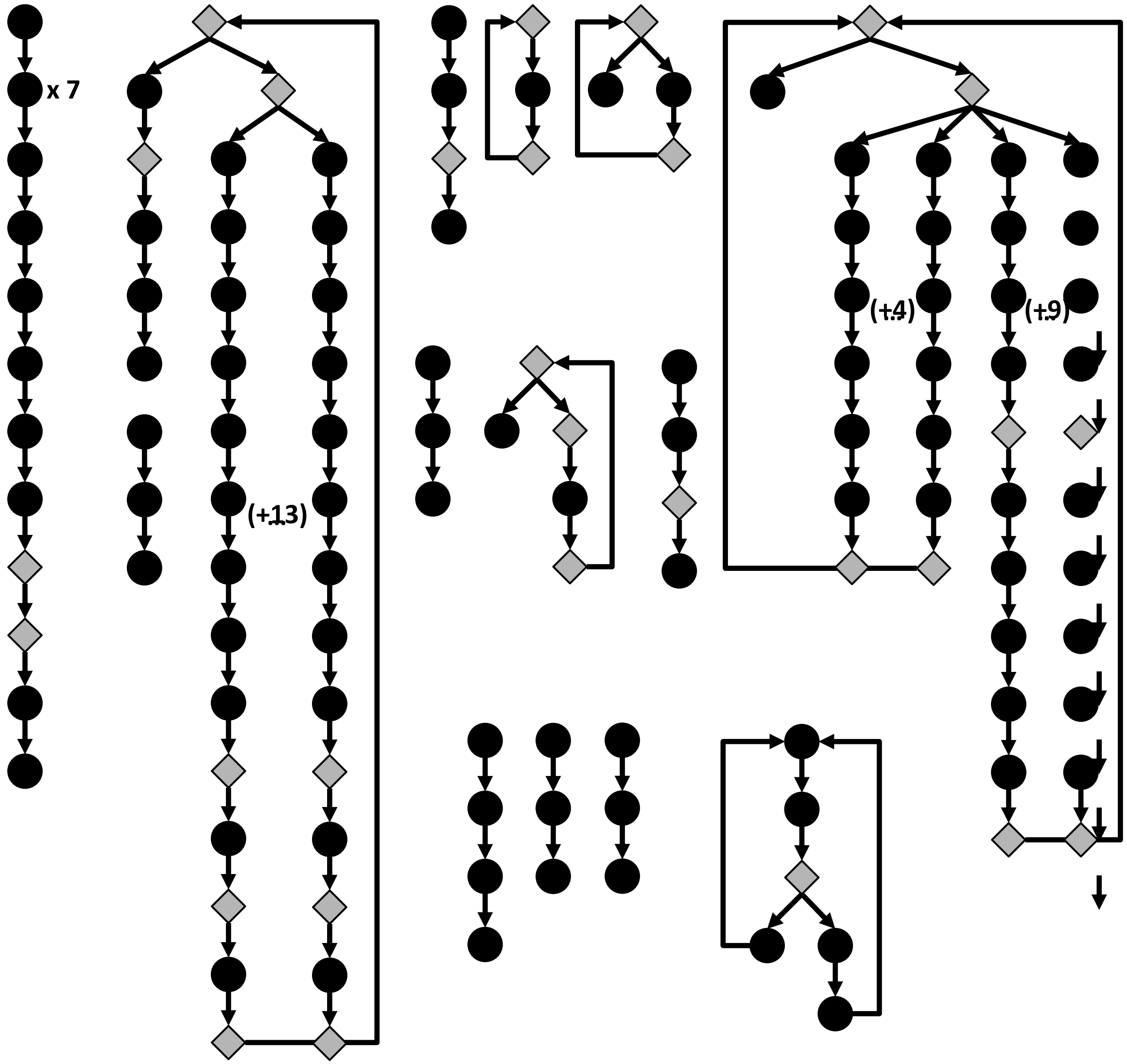
- Work scheduling
- Task scheduler and synchronization primitives
- Interrupts

NIC Driver
17 tracepoints

- NIC, IP and driver TX queues (4), receive and transmit services and loops (8), service context (5)

Parallel execution (DMA, …):
6 Tracepoints

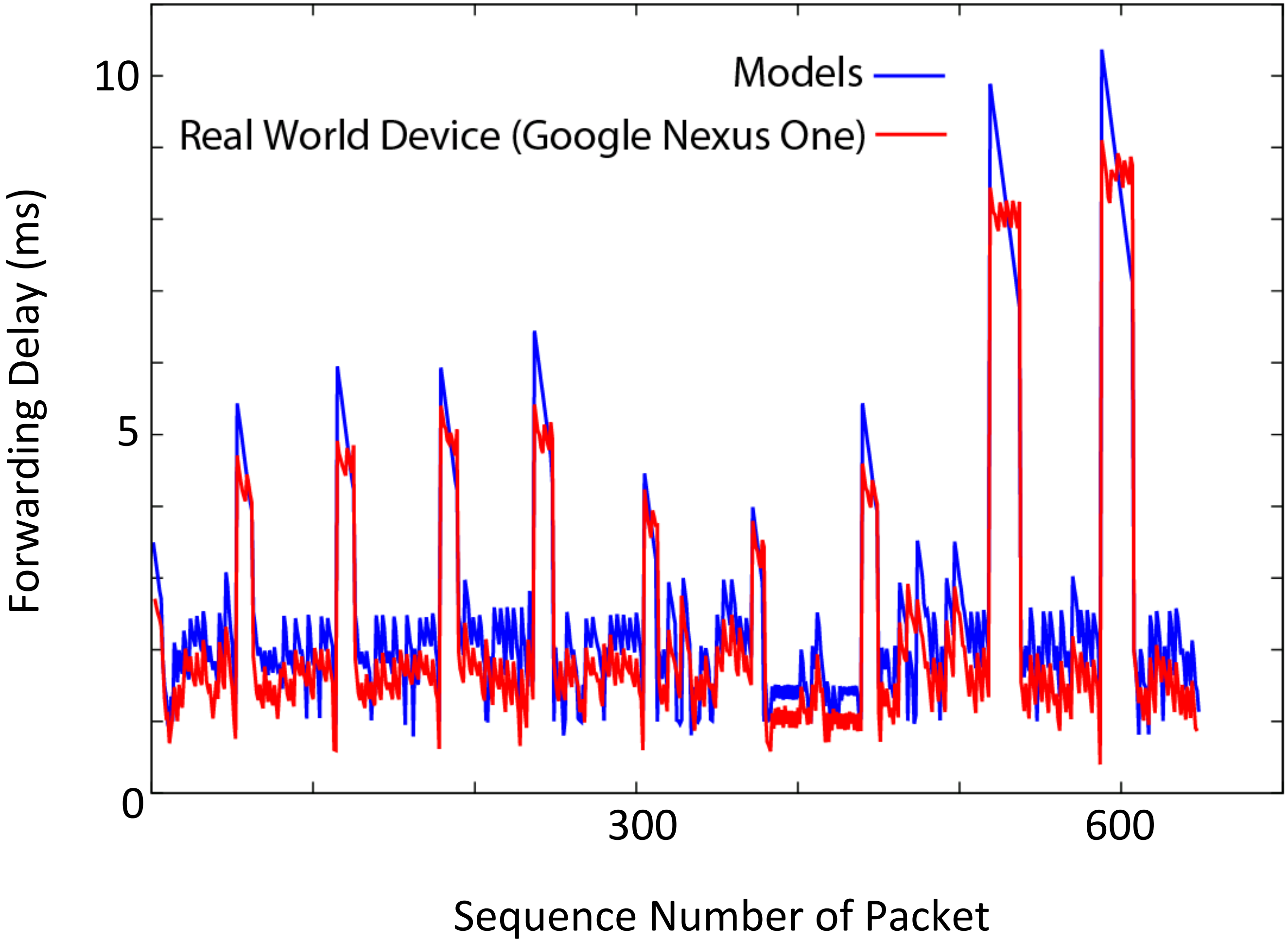
Google Nexus One: Final Models



2-300.000 instructions

about 100 statements

Google Nexus One: Key Result



Possible Master Theses

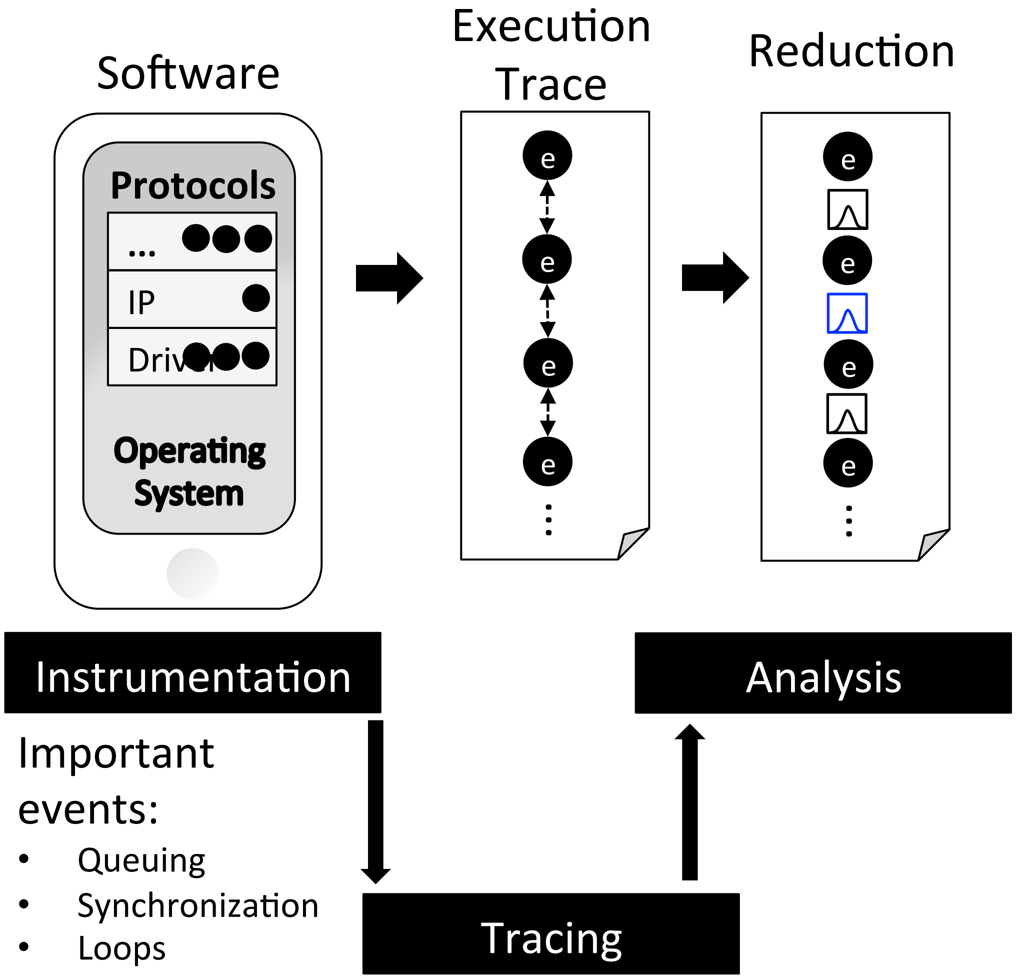
- Focus on programming:
  - Extend tracing framework for multi-core devices
  - Visualization
- Focus on modelling:
  - Additional protocols on the GN1
  - The N900
  - Sensors with TinyOS, Contiki, …
  - Software based routers
- Focus on experimentation:
  - Medium- to large-scale experiments
  - Replicate published experiments and compare results

Thank you for your Attention!

Questions?

(… or contact me for more information!)

Core Idea



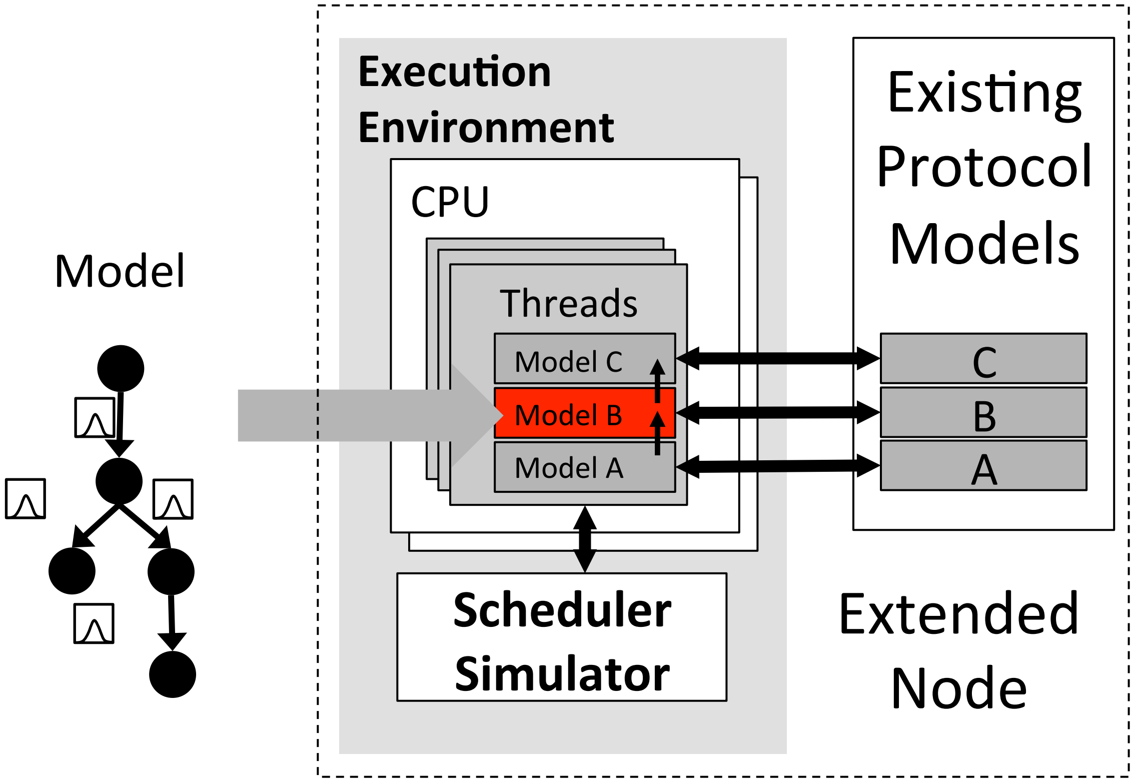
Important events:

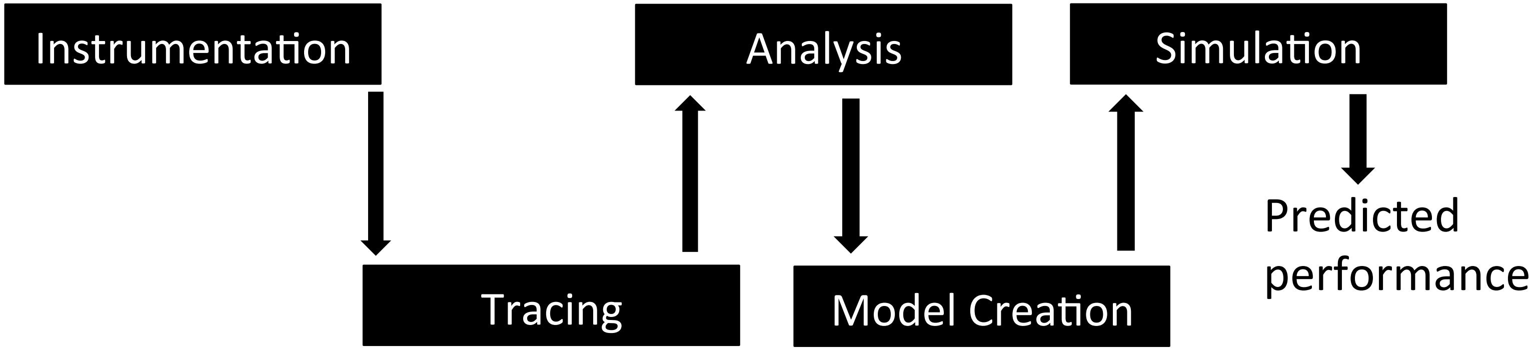
- Queuing
- Synchronization
- Loops
- Interactions with HW
- **Impact of Context**

Behaviour affected by execution context,e.g.,:

- Packet size, type, …
- Routing table size, …
- Bus state, …

Core Idea





Instrumentation

Tracing

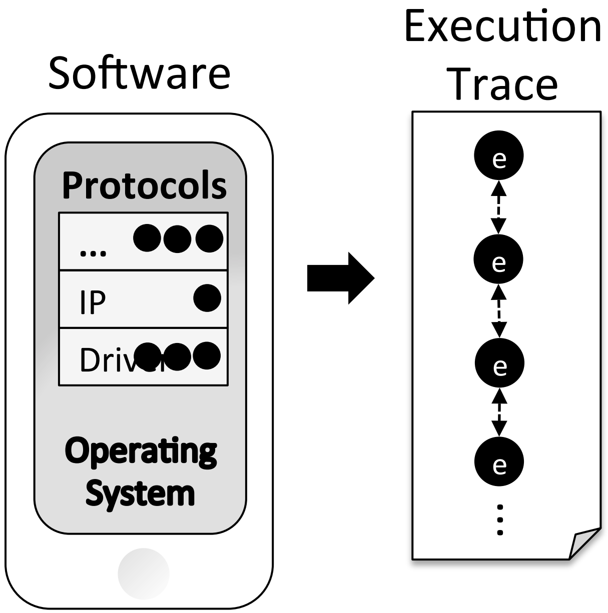
Analysis

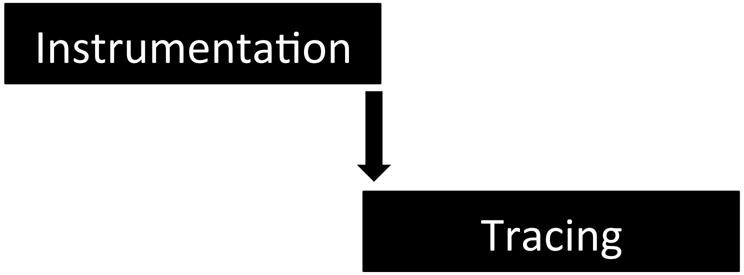
Model Creation

Simulation

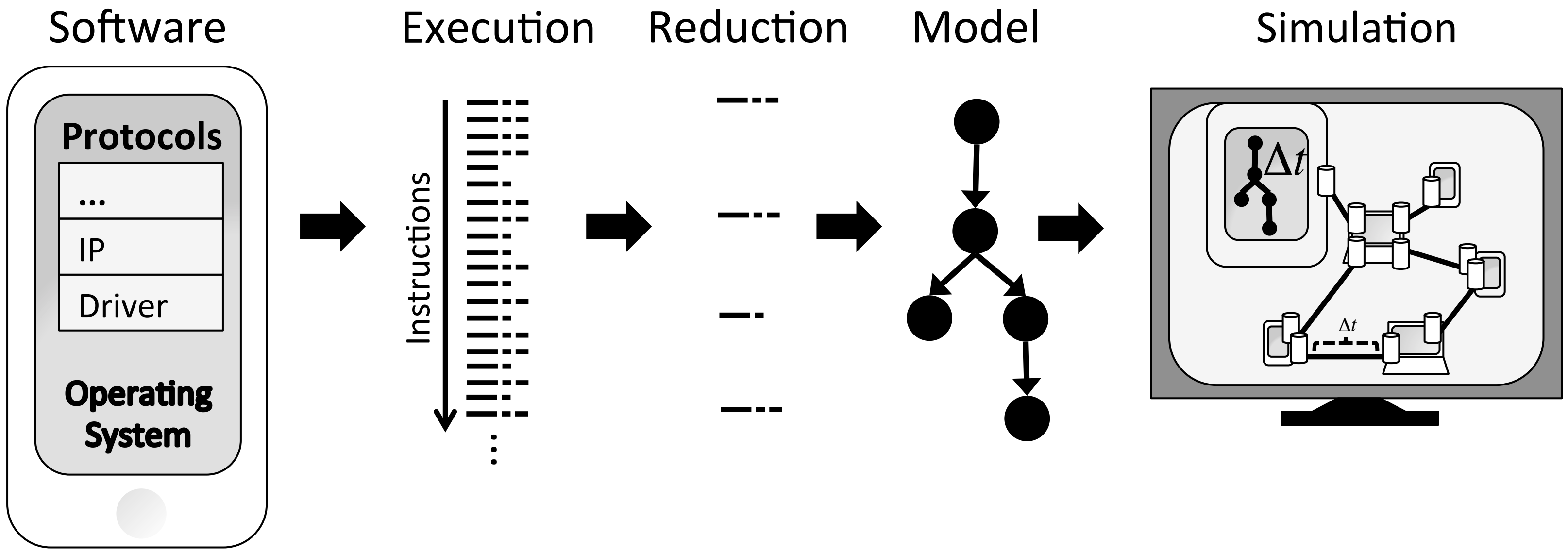
Predicted performance

Core Idea


Software
Protocols
...
IP
Driver
Operating System
Execution Trace
e
e
e
e
⋮


Instrumentation
Tracing

Core Idea



Software

Protocols

...

IP

Driver

Operating System

Execution

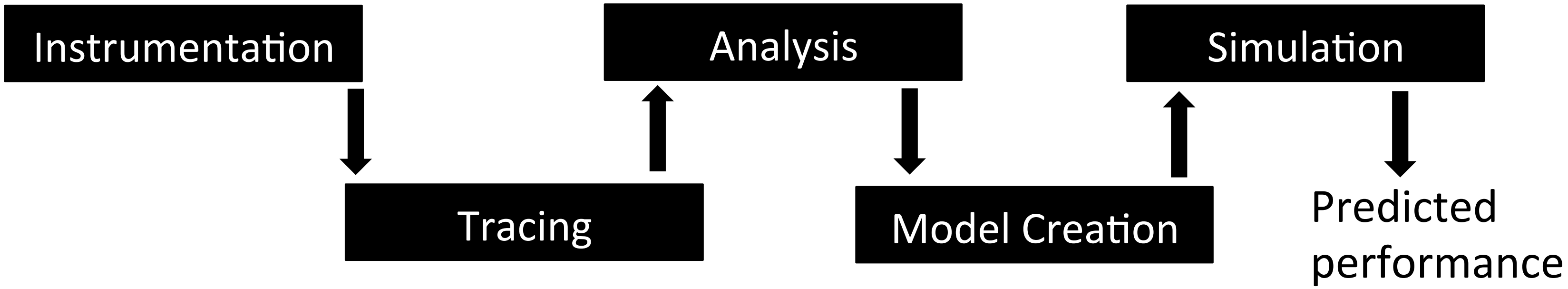
Instructions

Reduction

Model

Simulation

Δt



Instrumentation

Tracing

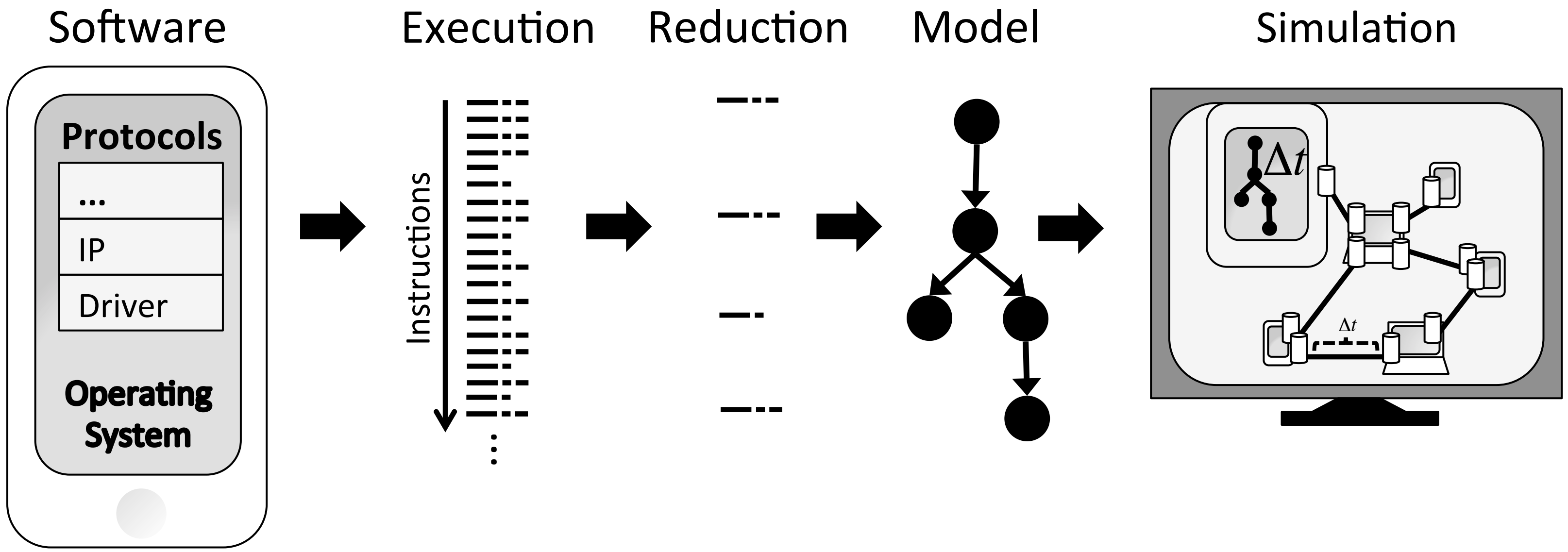
Analysis

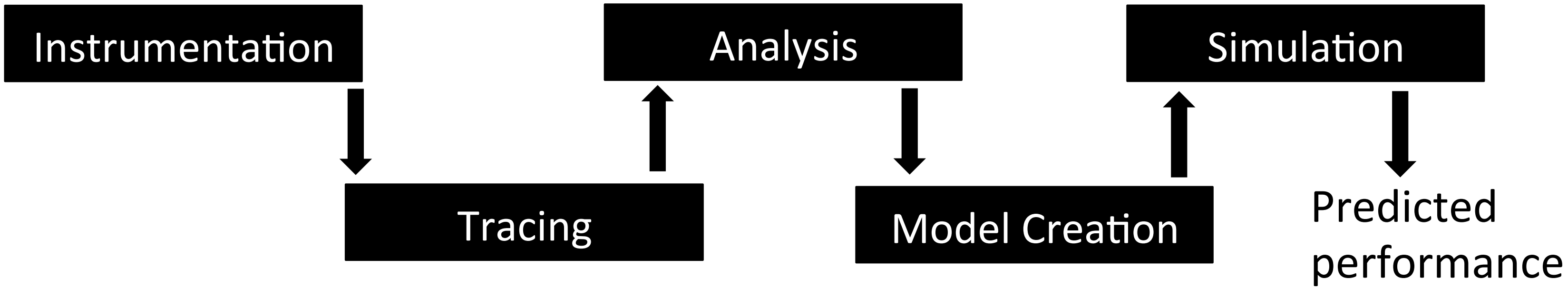
Model Creation

Simulation

Predicted performance

Core Idea





Instrumentation

Tracing

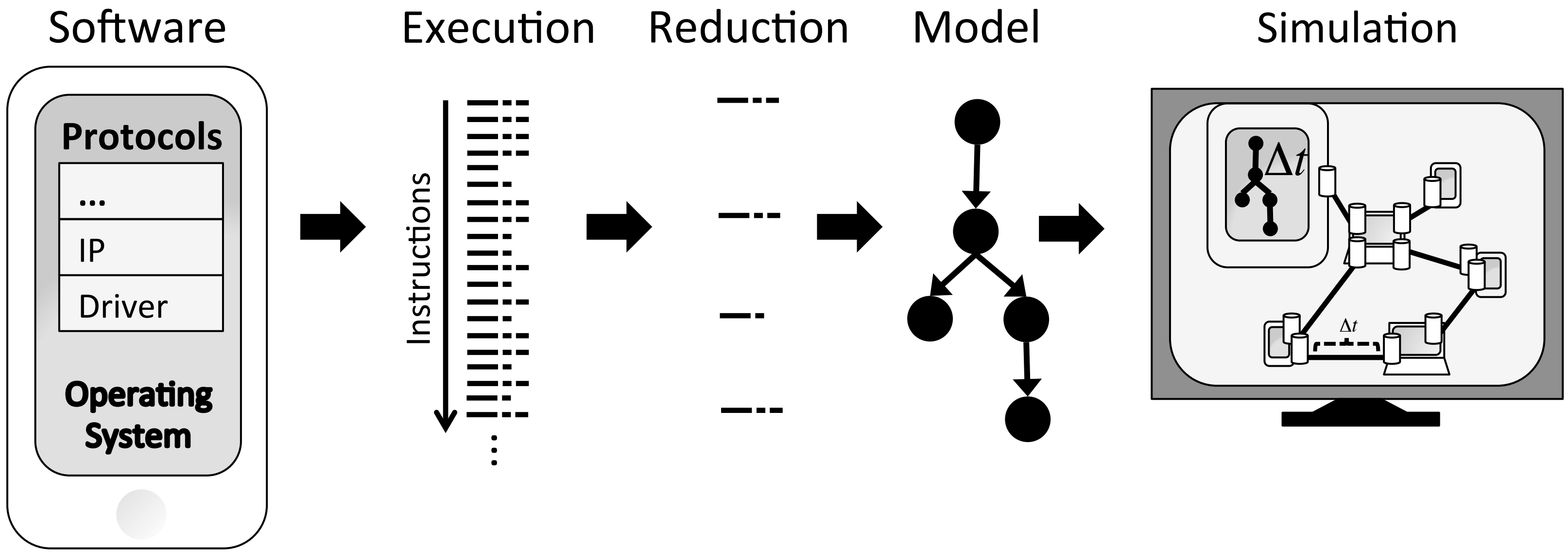
Analysis

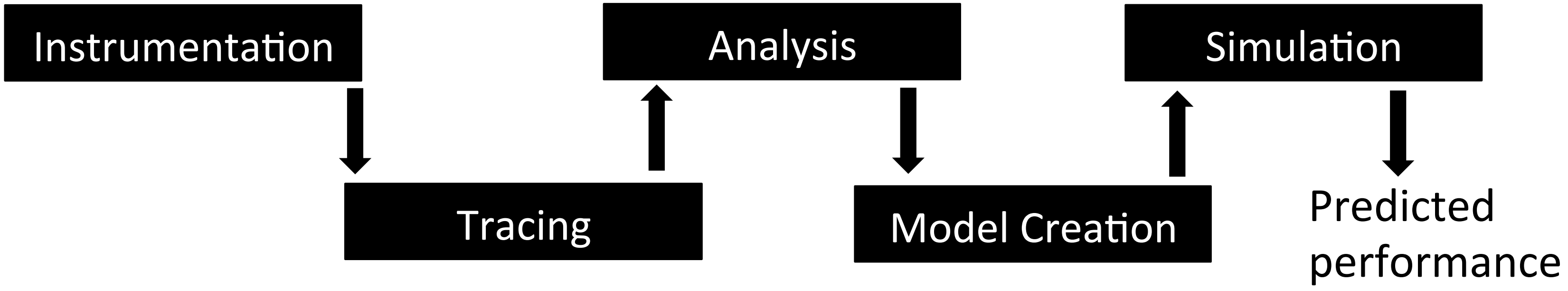
Model Creation

Simulation

Predicted performance

Core Idea





Instrumentation

Tracing

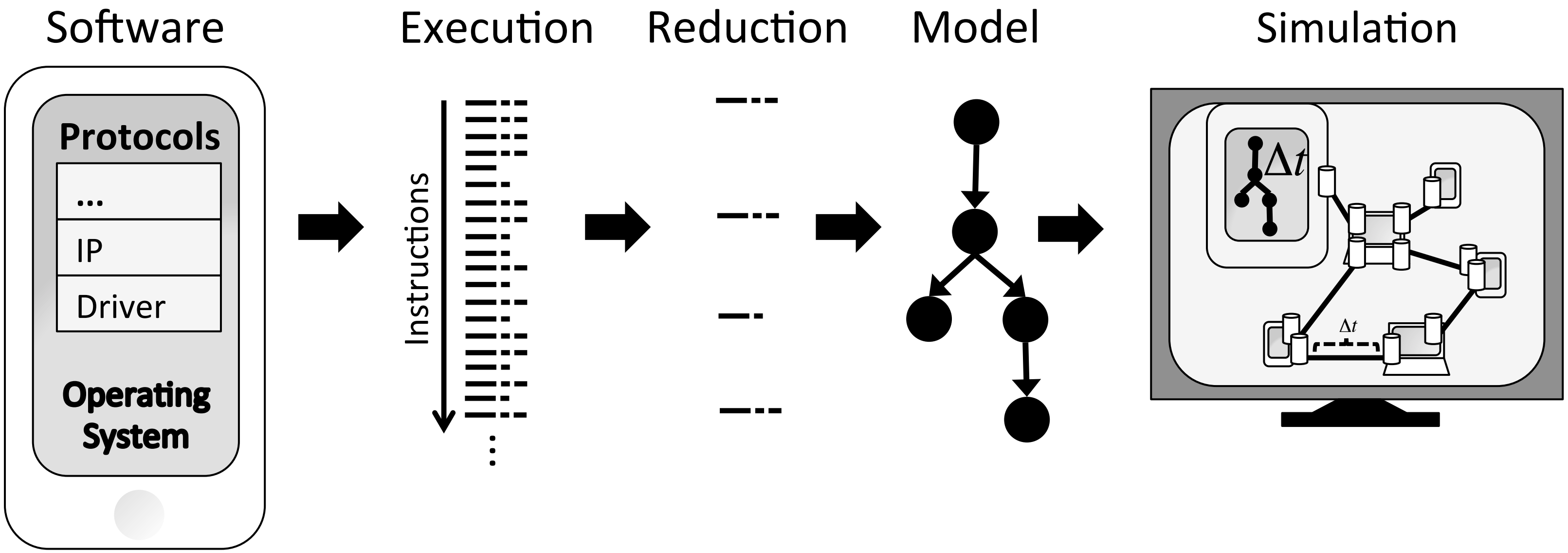
Analysis

Model Creation

Simulation

Predicted performance

Core Idea



Software

Protocols

...

IP

Driver

Operating System

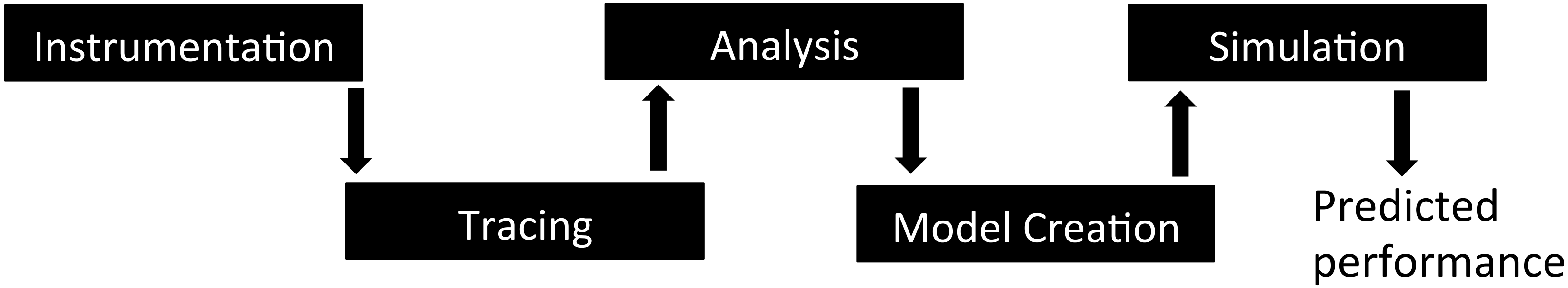
Execution

Instructions

Reduction

Model

Simulation



Instrumentation

Tracing

Analysis

Model Creation

Simulation

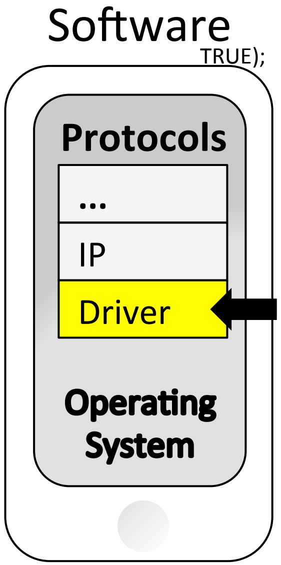
Predicted performance

Important Events

Reason	Required Events
Separating services	**Class 1:** • SRVStart, SRVStop • CTXSwitch • HIRQStart, HIRQStop
CPU processing	**Class 2:** • CPU cycles • (and/or hardware performance counters)
Interactions with task scheduler	**Class 3:** • Awake • Sleep • Synch
Work scheduling in threads and interrupts	**Class 4:** • SRVStart, SRVStop • LoopStart, LoopRestart, LoopStop • Enqueue, Dequeue
Parallel processing	**Class 5:** • PEUStart • HIRQStart • Synch
Execution context	**Class 6:** • CondRead/Write • Dequeue • SRVEntry

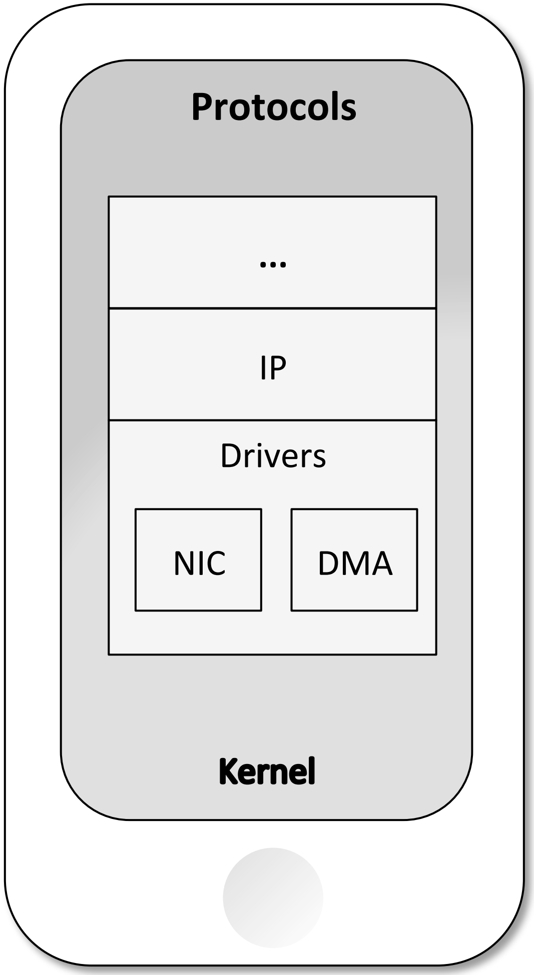
Example Instrumentation

```
uint
dhdsdio_sendfromq(dhd_bus_t *bus, uint maxframes, bool rxdone) {
    void *pkt;
    uint32 intstatus = 0;
    uint retries = 0;
    int ret = 0, prec_out;
    uint cnt = 0;
    uint datalen;
    uint8 tx_prec_map;

    dhd_pub_t *dhd = bus->dhd;
    sdpcmd_regs_t *regs = bus->regs;

    DHD_TRACE(("%s: Enter\n", __FUNCTION__));
    trace_sepext_queuecondition(rxdone, PKTQ_NIC, PKTQ_NIC);

    tx_prec_map = ~bus->flowcontrol;

    trace_sepext_loopstart(maxframes, 0, PKTQ_BCM4329_TXQ,
         PKTQ_BCM4329_TXQ);
    /* Send frames until the limit or some other event */
    for (cnt = 0; (cnt < maxframes) && DATAOK(bus); cnt++) {
        dhd_os_sdlock_txq(bus->dhd);

        trace_sepext_looprestart();

        if ((pkt = pktq_mdeq(&bus->txq, tx_prec_map, &prec_out)) == NULL) {
            dhd_os_sdunlock_txq(bus->dhd);
            break;
        }

        trace_sepext_pktqueue(((struct sk_buff *) pkt)->data,
         PKTQ_BCM4329_TXQ, 0, QUEUE_DEQUEUE);

        dhd_os_sdunlock_txq(bus->dhd);
        datalen = PKTLEN(bus->dhd->osh, pkt) - SDPCM_HDRLEN;
```


Software

```
#ifndef SDTEST
        ret = dhdsdio_txpkt(bus, pkt, SDPCM_DATA_CHANNEL, TRUE);
#else
        ret = dhdsdio_txpkt(bus, pkt,
            (bus->ext_loop ? SDPCM_TEST_CHANNEL : SDPCM_DATA_CHANNEL),
TRUE);


        if (ret)
            bus->dhd->tx_errors++;
        else
            bus->dhd->dstats.tx_bytes += datalen;

        /* In poll mode, need to check for other events */
        if (!bus->intr && cnt)
        {
            /* Check device status, signal pending interrupt */
            R_SDREG(intstatus, &regs->intstatus, retries);
            bus->f2txdata++;
            if (bcmsdh_regfail(bus->sdh)) {
             break;
            }
            if (intstatus & bus->hostintmask)
                bus->ipend = TRUE;
        }
    }
         trace_sepext_loopstop();

    /* Deflow-control stack if needed */
    if (dhd_doflow && dhd->up && (dhd->busstate == DHD_BUS_DATA) &&
      dhd->txoff && (pktq_len(&bus->txq) < FCLOW))
        dhd_txflowcontrol(dhd, 0, OFF);

         return cnt;
     }
```

Google Nexus One: Instrumentation



Instrumentation:

Networking sub-system, including IP
10 tracepoints

The rest of kernel
58 tracepoints

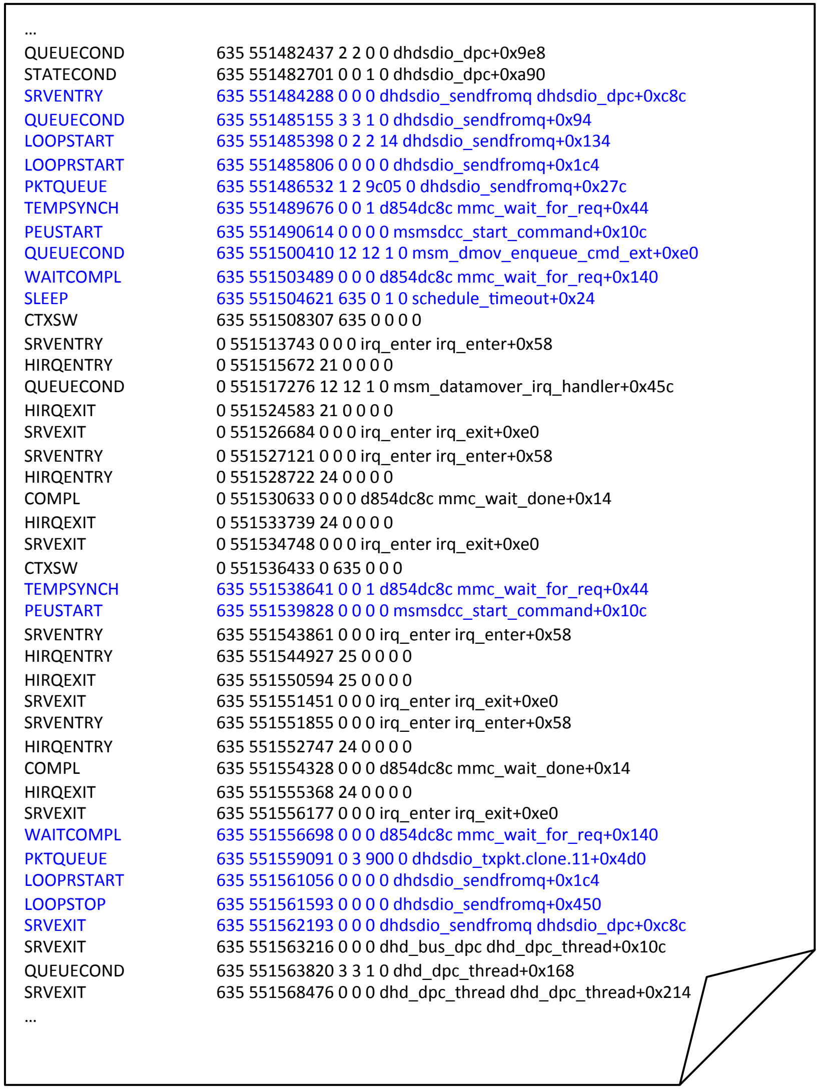
- Work scheduling
- Task scheduler and synchronization primitives
- Interrupts

NIC Driver
17 tracepoints

- NIC, IP and driver TX queues (4), receive and transmit services and loops (8), service context (5)

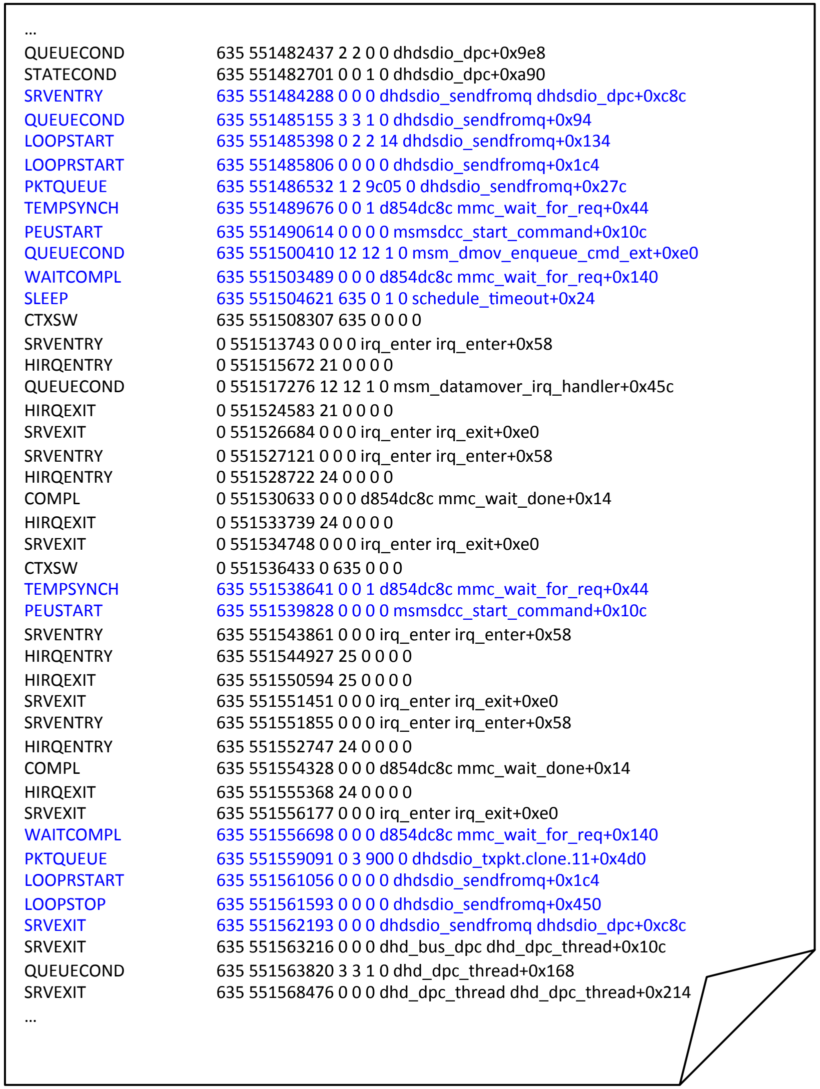
Parallel execution (DMA, …):
6 Tracepoints

Resulting Trace File

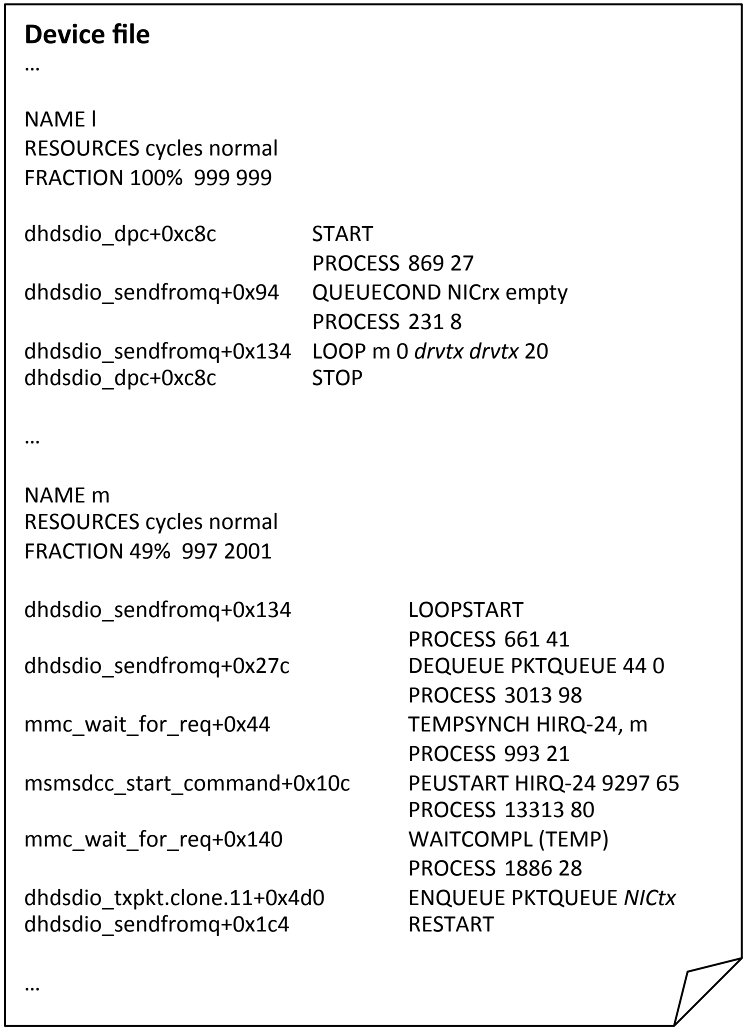


```
…
QUEUECOND        635 551482437 2 2 0 0 dhdsdio_dpc+0x9e8
STATECOND        635 551482701 0 0 1 0 dhdsdio_dpc+0xa90
SRVENTRY         635 551484288 0 0 0 dhdsdio_sendfromq dhdsdio_dpc+0xc8c
QUEUECOND        635 551485155 3 3 1 0 dhdsdio_sendfromq+0x94
LOOPSTART        635 551485398 0 2 2 14 dhdsdio_sendfromq+0x134
LOOPRSTART       635 551485806 0 0 0 0 dhdsdio_sendfromq+0x1c4
PKTQUEUE         635 551486532 1 2 9c05 0 dhdsdio_sendfromq+0x27c
TEMPSYNCH        635 551489676 0 0 1 d854dc8c mmc_wait_for_req+0x44
PEUSTART         635 551490614 0 0 0 0 msmsdcc_start_command+0x10c
QUEUECOND        635 551500410 12 12 1 0 msm_dmov_enqueue_cmd_ext+0xe0
WAITCOMPL        635 551503489 0 0 0 d854dc8c mmc_wait_for_req+0x140
SLEEP            635 551504621 635 0 1 0 schedule_timeout+0x24
CTXSW            635 551508307 635 0 0 0 0
SRVENTRY         0 551513743 0 0 0 irq_enter irq_enter+0x58
HIRQENTRY        0 551515672 21 0 0 0 0
QUEUECOND        0 551517276 12 12 1 0 msm_datamover_irq_handler+0x45c
HIRQEXIT         0 551524583 21 0 0 0 0
SRVEXIT          0 551526684 0 0 0 irq_enter irq_exit+0xe0
SRVENTRY         0 551527121 0 0 0 irq_enter irq_enter+0x58
HIRQENTRY        0 551528722 24 0 0 0 0
COMPL            0 551530633 0 0 0 d854dc8c mmc_wait_done+0x14
HIRQEXIT         0 551533739 24 0 0 0 0
SRVEXIT          0 551534748 0 0 0 irq_enter irq_exit+0xe0
CTXSW            0 551536433 0 635 0 0 0
TEMPSYNCH        635 551538641 0 0 1 d854dc8c mmc_wait_for_req+0x44
PEUSTART         635 551539828 0 0 0 0 msmsdcc_start_command+0x10c
SRVENTRY         635 551543861 0 0 0 irq_enter irq_enter+0x58
HIRQENTRY        635 551544927 25 0 0 0 0
HIRQEXIT         635 551550594 25 0 0 0 0
SRVEXIT          635 551551451 0 0 0 irq_enter irq_exit+0xe0
SRVENTRY         635 551551855 0 0 0 irq_enter irq_enter+0x58
HIRQENTRY        635 551552747 24 0 0 0 0
COMPL            635 551554328 0 0 0 d854dc8c mmc_wait_done+0x14
HIRQEXIT         635 551555368 24 0 0 0 0
SRVEXIT          635 551556177 0 0 0 irq_enter irq_exit+0xe0
WAITCOMPL        635 551556698 0 0 0 d854dc8c mmc_wait_for_req+0x140
PKTQUEUE         635 551559091 0 3 900 0 dhdsdio_txpkt.clone.11+0x4d0
LOOPRSTART       635 551561056 0 0 0 0 dhdsdio_sendfromq+0x1c4
LOOPSTOP         635 551561593 0 0 0 0 dhdsdio_sendfromq+0x450
SRVEXIT          635 551562193 0 0 0 dhdsdio_sendfromq dhdsdio_dpc+0xc8c
SRVEXIT          635 551563216 0 0 0 dhd_bus_dpc dhd_dpc_thread+0x10c
QUEUECOND        635 551563820 3 3 1 0 dhd_dpc_thread+0x168
SRVEXIT          635 551568476 0 0 0 dhd_dpc_thread dhd_dpc_thread+0x214
…
```

Resulting Reduction



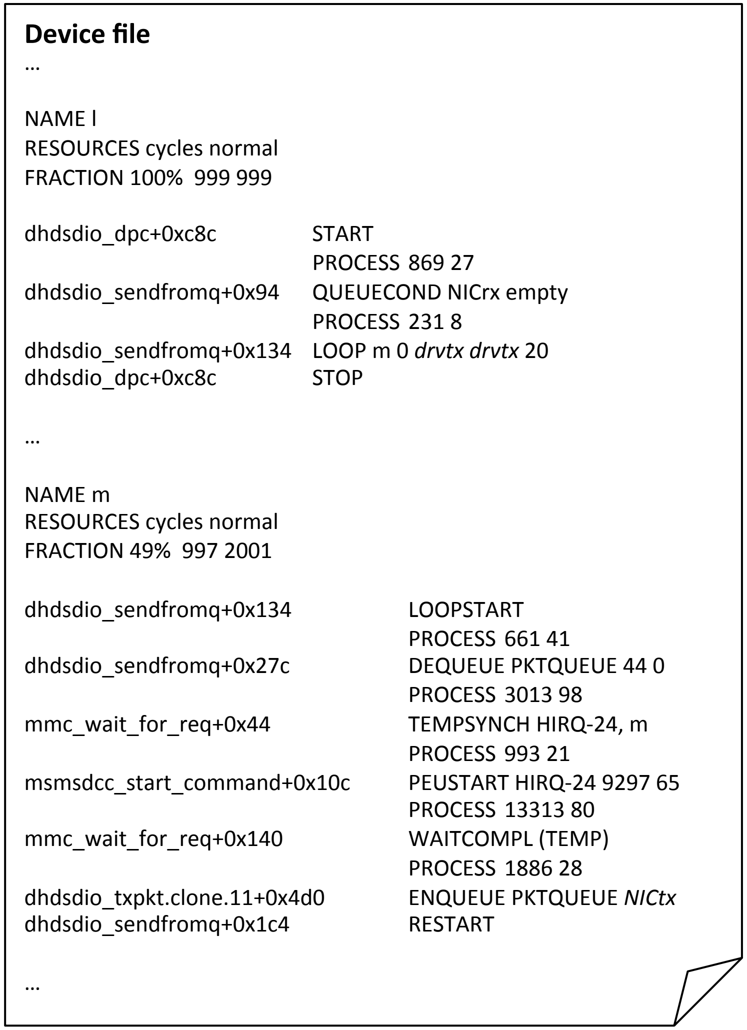
```
…
QUEUECOND      635 551482437 2 2 0 0 dhdsdio_dpc+0x9e8
STATECOND      635 551482701 0 0 1 0 dhdsdio_dpc+0xa90
SRVENTRY       635 551484288 0 0 0 dhdsdio_sendfromq dhdsdio_dpc+0xc8c
QUEUECOND      635 551485155 3 3 1 0 dhdsdio_sendfromq+0x94
LOOPSTART      635 551485398 0 2 2 14 dhdsdio_sendfromq+0x134
LOOPRSTART     635 551485806 0 0 0 0 dhdsdio_sendfromq+0x1c4
PKTQUEUE       635 551486532 1 2 9c05 0 dhdsdio_sendfromq+0x27c
TEMPSYNCH      635 551489676 0 0 1 d854dc8c mmc_wait_for_req+0x44
PEUSTART       635 551490614 0 0 0 0 msmsdcc_start_command+0x10c
QUEUECOND      635 551500410 12 12 1 0 msm_dmov_enqueue_cmd_ext+0xe0
WAITCOMPL      635 551503489 0 0 0 d854dc8c mmc_wait_for_req+0x140
SLEEP          635 551504621 635 0 1 0 schedule_timeout+0x24
CTXSW          635 551508307 635 0 0 0 0
SRVENTRY       0 551513743 0 0 0 irq_enter irq_enter+0x58
HIRQENTRY      0 551515672 21 0 0 0 0
QUEUECOND      0 551517276 12 12 1 0 msm_datamover_irq_handler+0x45c
HIRQEXIT       0 551524583 21 0 0 0 0
SRVEXIT        0 551526684 0 0 0 irq_enter irq_exit+0xe0
SRVENTRY       0 551527121 0 0 0 irq_enter irq_enter+0x58
HIRQENTRY      0 551528722 24 0 0 0 0
COMPL          0 551530633 0 0 0 d854dc8c mmc_wait_done+0x14
HIRQEXIT       0 551533739 24 0 0 0 0
SRVEXIT        0 551534748 0 0 0 irq_enter irq_exit+0xe0
CTXSW          0 551536433 0 635 0 0 0
TEMPSYNCH      635 551538641 0 0 1 d854dc8c mmc_wait_for_req+0x44
PEUSTART       635 551539828 0 0 0 0 msmsdcc_start_command+0x10c
SRVENTRY       635 551543861 0 0 0 irq_enter irq_enter+0x58
HIRQENTRY      635 551544927 25 0 0 0 0
HIRQEXIT       635 551550594 25 0 0 0 0
SRVEXIT        635 551551451 0 0 0 irq_enter irq_exit+0xe0
SRVENTRY       635 551551855 0 0 0 irq_enter irq_enter+0x58
HIRQENTRY      635 551552747 24 0 0 0 0
COMPL          635 551554328 0 0 0 d854dc8c mmc_wait_done+0x14
HIRQEXIT       635 551555368 24 0 0 0 0
SRVEXIT        635 551556177 0 0 0 irq_enter irq_exit+0xe0
WAITCOMPL      635 551556698 0 0 0 d854dc8c mmc_wait_for_req+0x140
PKTQUEUE       635 551559091 0 3 900 0 dhdsdio_txpkt.clone.11+0x4d0
LOOPRSTART     635 551561056 0 0 0 0 dhdsdio_sendfromq+0x1c4
LOOPSTOP       635 551561593 0 0 0 0 dhdsdio_sendfromq+0x450
SRVEXIT        635 551562193 0 0 0 dhdsdio_sendfromq dhdsdio_dpc+0xc8c
SRVEXIT        635 551563216 0 0 0 dhd_bus_dpc dhd_dpc_thread+0x10c
QUEUECOND      635 551563820 3 3 1 0 dhd_dpc_thread+0x168
SRVEXIT        635 551568476 0 0 0 dhd_dpc_thread dhd_dpc_thread+0x214
…
```



Device file

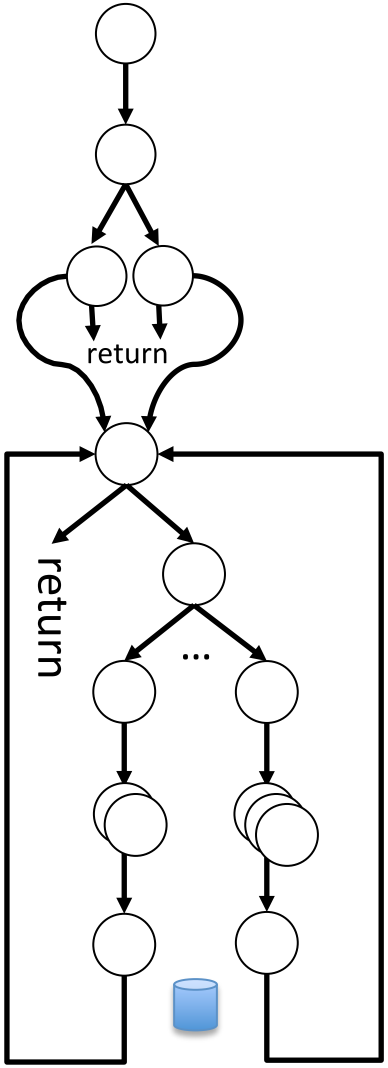
```
…

NAME l
RESOURCES cycles normal
FRACTION 100%  999 999

dhdsdio_dpc+0xc8c            START
                             PROCESS 869 27
dhdsdio_sendfromq+0x94       QUEUECOND NICrx empty
                             PROCESS 231 8
dhdsdio_sendfromq+0x134      LOOP m 0 drvtx drvtx 20
dhdsdio_dpc+0xc8c            STOP

…

NAME m
RESOURCES cycles normal
FRACTION 49%  997 2001

dhdsdio_sendfromq+0x134          LOOPSTART
                                 PROCESS 661 41
dhdsdio_sendfromq+0x27c          DEQUEUE PKTQUEUE 44 0
                                 PROCESS 3013 98
mmc_wait_for_req+0x44            TEMPSYNCH HIRQ-24, m
                                 PROCESS 993 21
msmsdcc_start_command+0x10c      PEUSTART HIRQ-24 9297 65
                                 PROCESS 13313 80
mmc_wait_for_req+0x140           WAITCOMPL (TEMP)
                                 PROCESS 1886 28
dhdsdio_txpkt.clone.11+0x4d0     ENQUEUE PKTQUEUE NICtx
dhdsdio_sendfromq+0x1c4          RESTART

…
```

Resulting Execution Model



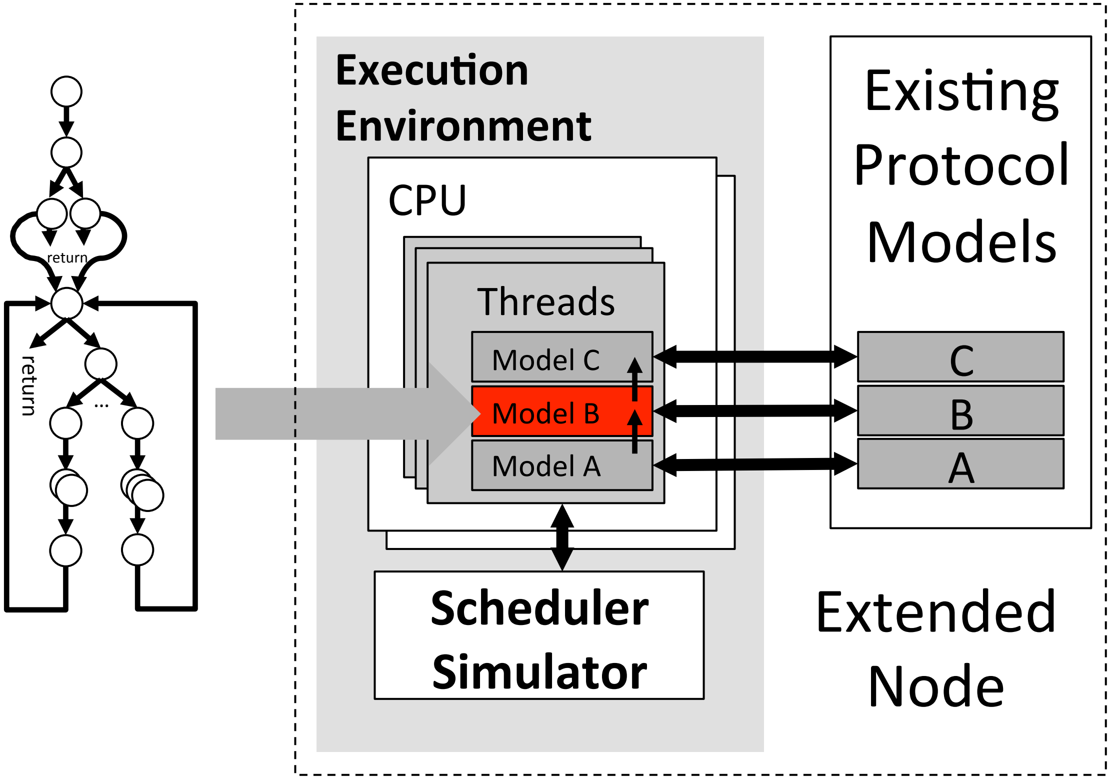
Device file

```
…

NAME l
RESOURCES cycles normal
FRACTION 100%  999 999

dhdsdio_dpc+0xc8c            START
                             PROCESS 869 27
dhdsdio_sendfromq+0x94       QUEUECOND NICrx empty
                             PROCESS 231 8
dhdsdio_sendfromq+0x134      LOOP m 0 drvtx drvtx 20
dhdsdio_dpc+0xc8c            STOP

…

NAME m
RESOURCES cycles normal
FRACTION 49%  997 2001

dhdsdio_sendfromq+0x134               LOOPSTART
                                      PROCESS 661 41
dhdsdio_sendfromq+0x27c               DEQUEUE PKTQUEUE 44 0
                                      PROCESS 3013 98
mmc_wait_for_req+0x44                 TEMPSYNCH HIRQ-24, m
                                      PROCESS 993 21
msmsdcc_start_command+0x10c           PEUSTART HIRQ-24 9297 65
                                      PROCESS 13313 80
mmc_wait_for_req+0x140                WAITCOMPL (TEMP)
                                      PROCESS 1886 28
dhdsdio_txpkt.clone.11+0x4d0          ENQUEUE PKTQUEUE NICtx
dhdsdio_sendfromq+0x1c4               RESTART

…
```

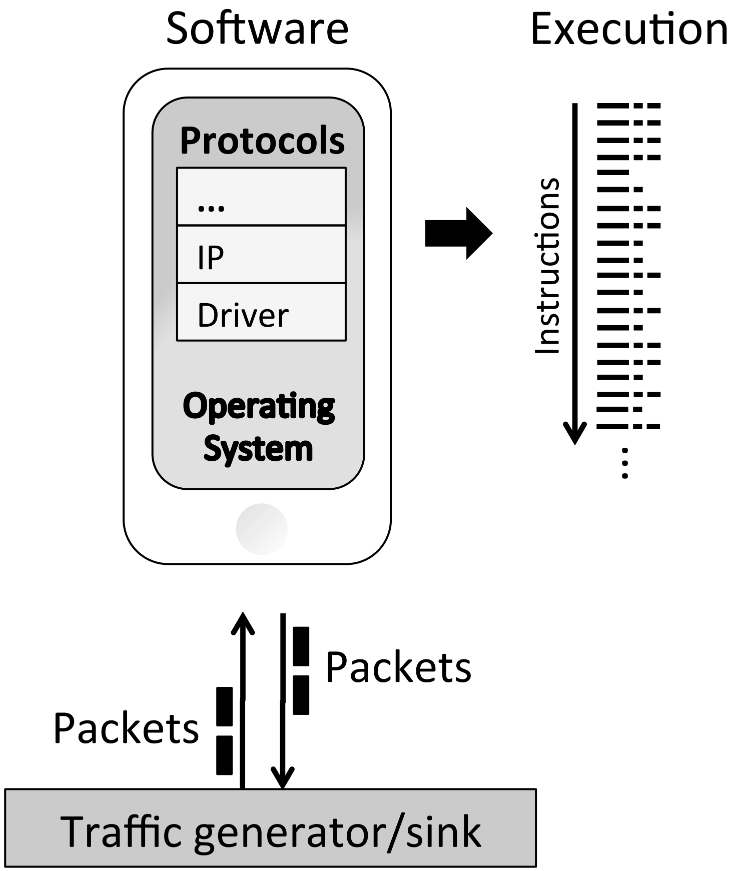


Extension of Simulator

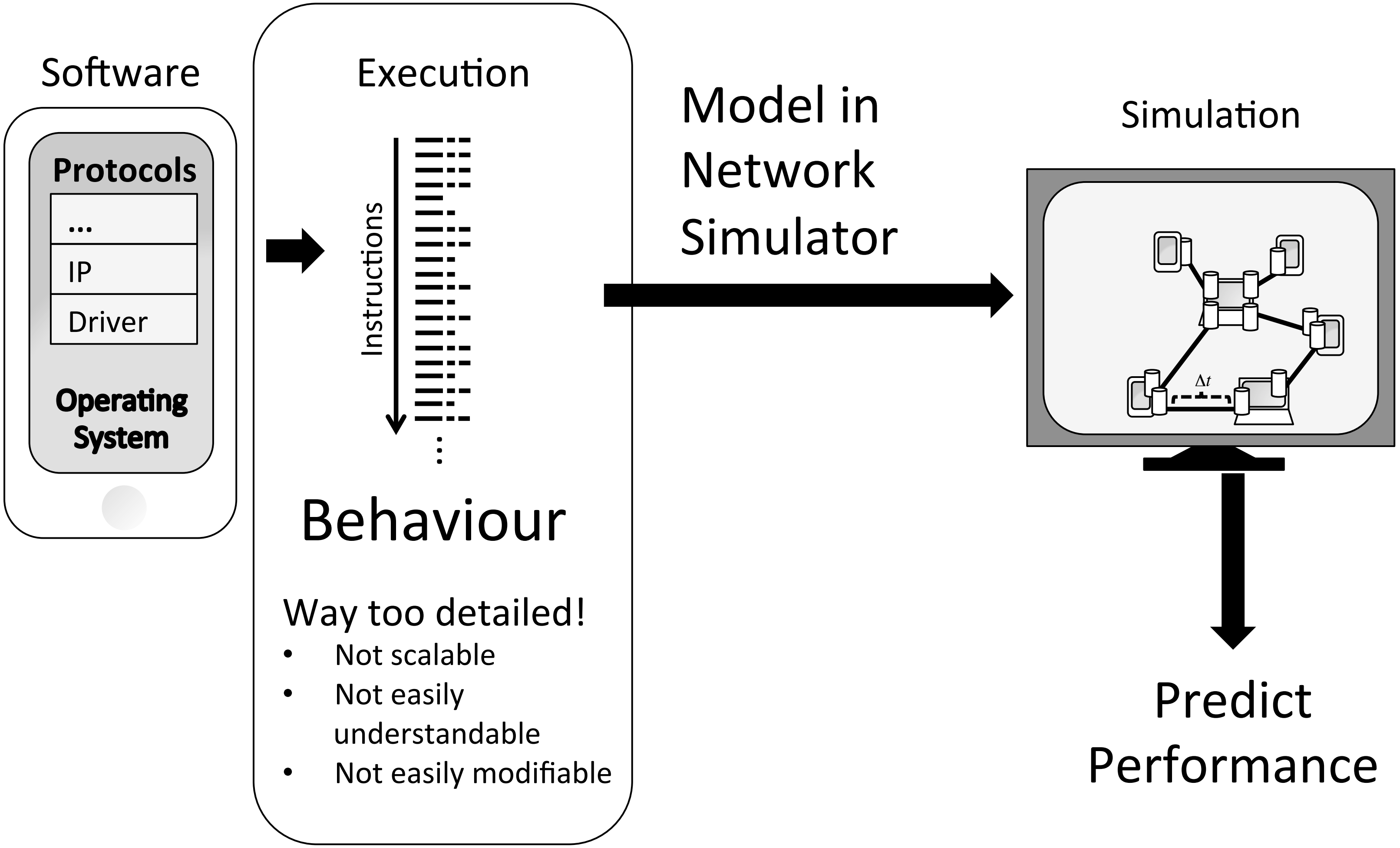
- Simulation:
  - Execute models in threads and interrupts
  - Uses scheduler simulator
  - Synchronize with existing protocol models



5-Step Approach



Core Idea



Software

Protocols

...

IP

Driver

Operating System

Execution

Instructions

Behaviour

Way too detailed!

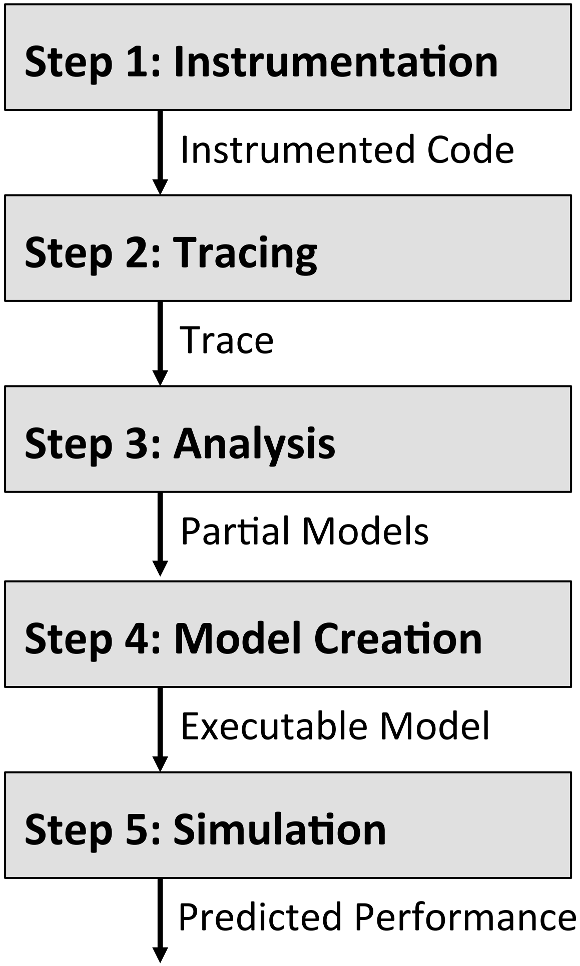
- Not scalable
- Not easily understandable
- Not easily modifiable

Model in Network Simulator

Simulation

Predict Performance

5-Step Appraoch to Simplify Models



Step 1: Instrumentation

Instrumented Code

Step 2: Tracing

Trace

Step 3: Analysis

Partial Models

Step 4: Model Creation

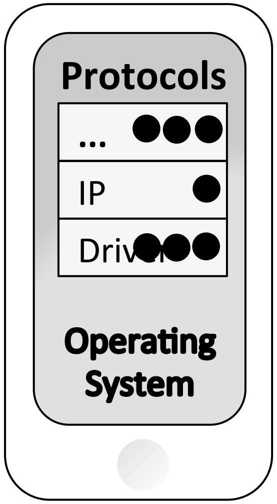
Executable Model

Step 5: Simulation

Predicted Performance

Step 1: Instrumentation

Software



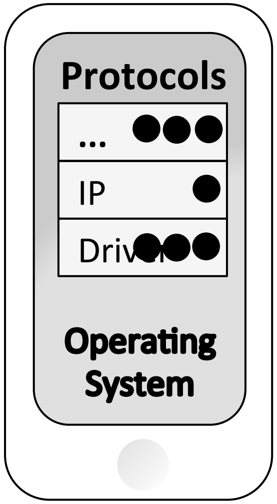
Instrumentation

Capture important events:

- Queuing
- Synchronization
- Loops
- Interactions with HW
- …

Step 1: Instrumentation

Software



Instrumentation

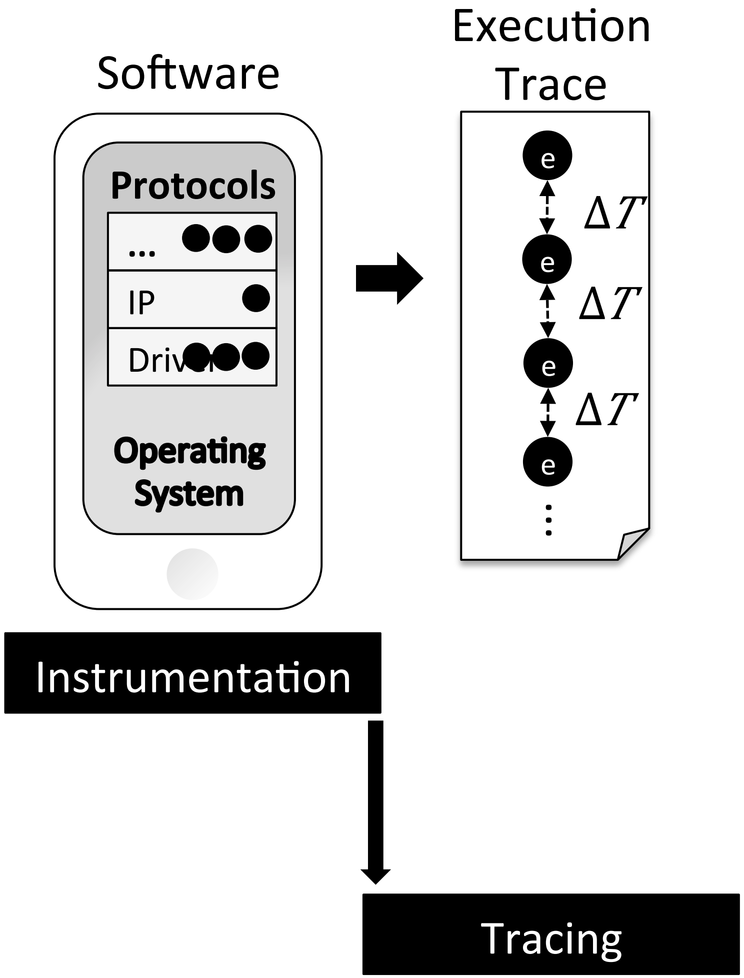
Capture important events:

- Queuing
- Synchronization
- Loops
- Interactions with HW
- …

Example:

```
…
pkt = pktq_mdeq(&bus->txq, tx_prec_map, &prec_out);



…
```

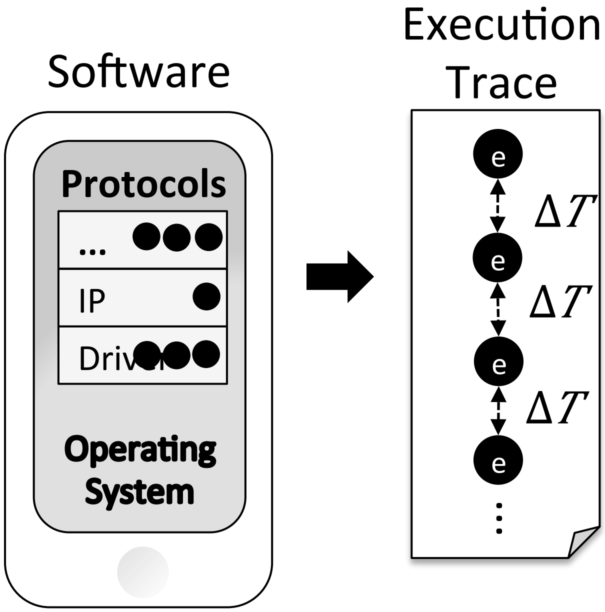
Step 2: Trace



Instrumentation

Tracing

Step 2: Trace



Example:

```
...
PKTQUEUE                ... 551486532 ...
SYNCH                   ... 551489676 ...
...
```

$\Delta T =$

551489676

$-$

551486532

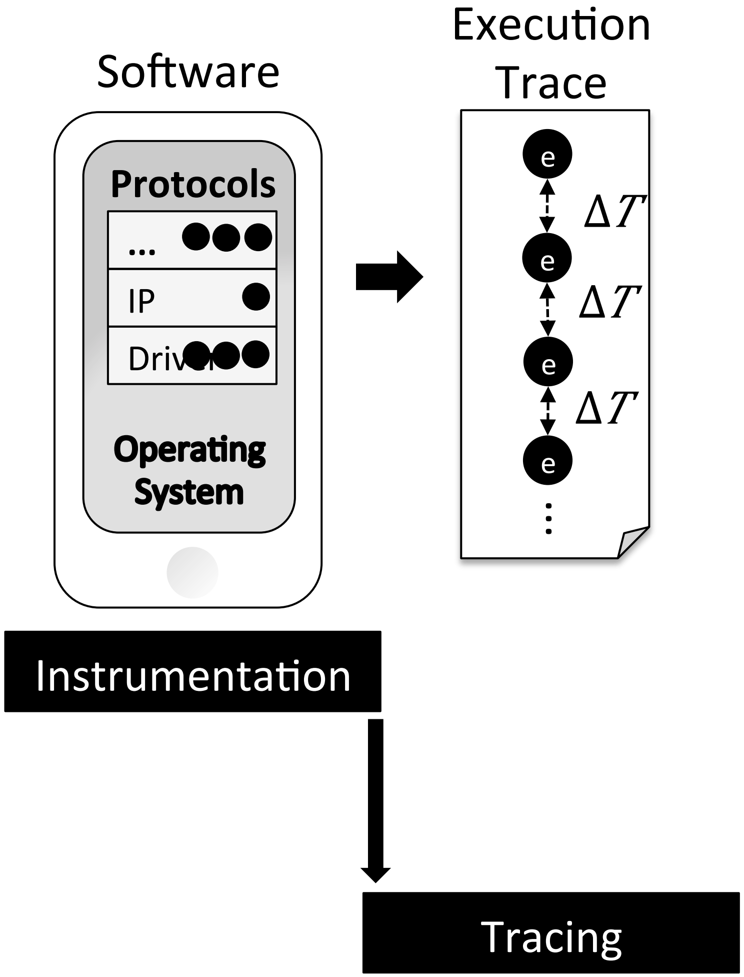
$=$

3144

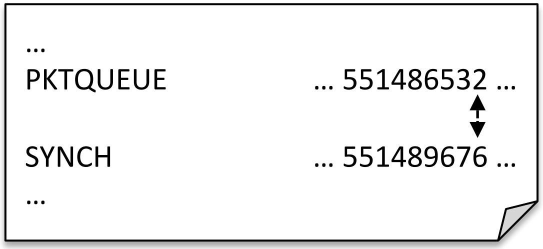
Instrumentation

Tracing

Step 2: Trace



Example:



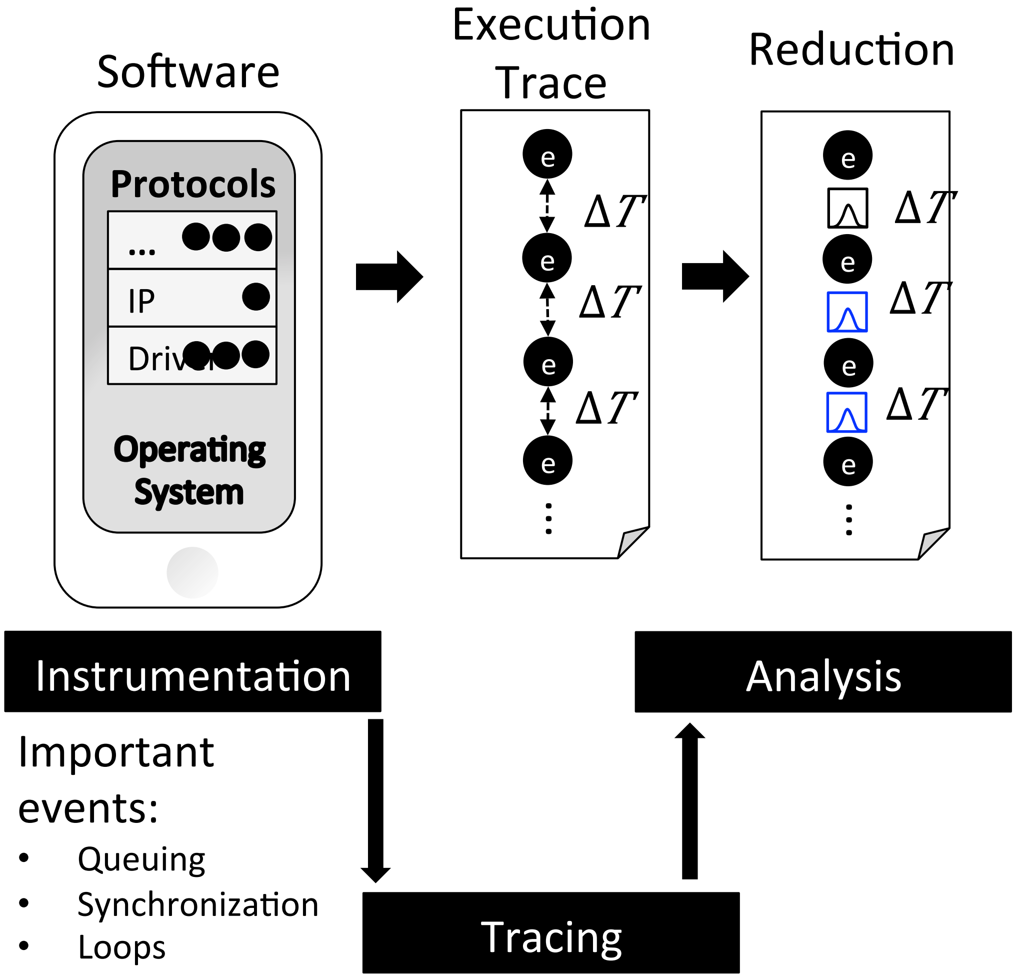
```
...
PKTQUEUE                 ... 551486532 ...

SYNCH                    ... 551489676 ...
...
```

ΔT=
551489676
–
551486532
=
3144

Small variations in consumed cycles

- Non-deterministic effects, e.g., memory caching

Step 3: Analyse



Example:

```
…
DEQUEUE PKTQUEUE …
PROCESS 3013 98
SYNCH …
PROCESS 993 21
…
```

Important events:

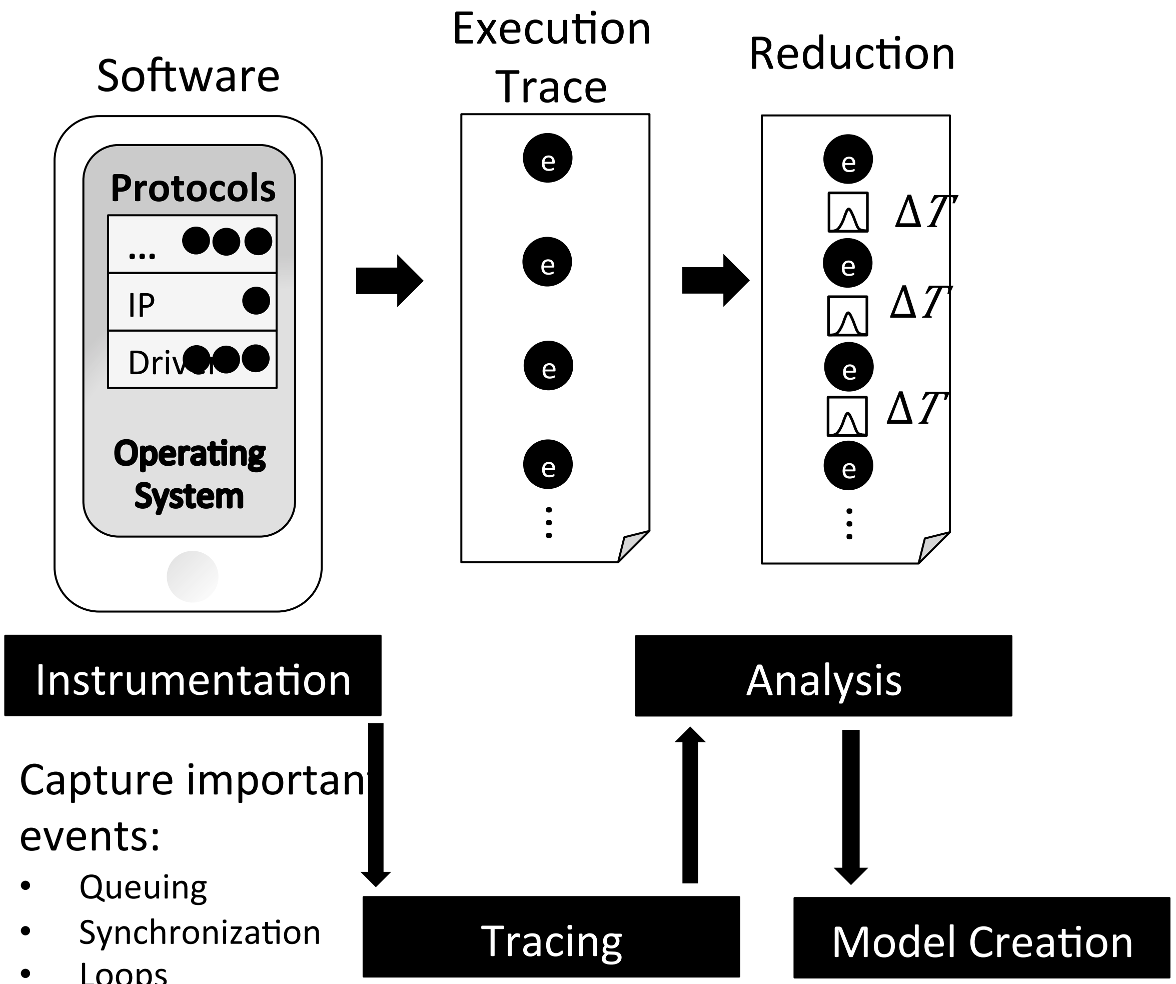
- Queuing
- Synchronization
- Loops
- Interactions with HW
- **Impact of Context**

Small variations in consumed cycles

- Non-deterministic effects, e.g., memory caching

Collect many traces and calculate statistics

Step 3: Analyse



Capture important events:

- Queuing
- Synchronization
- Loops
- Interactions with HW
- Impact of context

Example:

```
void fetchFromNIC(Packet *pkt)
{
  trace_semext_pktchar(*pkt);
  if(size(pkt) > 500) {
    DMATranferNIC(pkt);
    condition_wait(dma);
  }
  else {
    DMATranferNIC (pkt);
    while(!dma_done) {};
  }
}
```

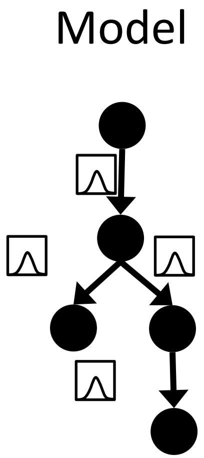
Behaviour affected by execution context,e.g.,:

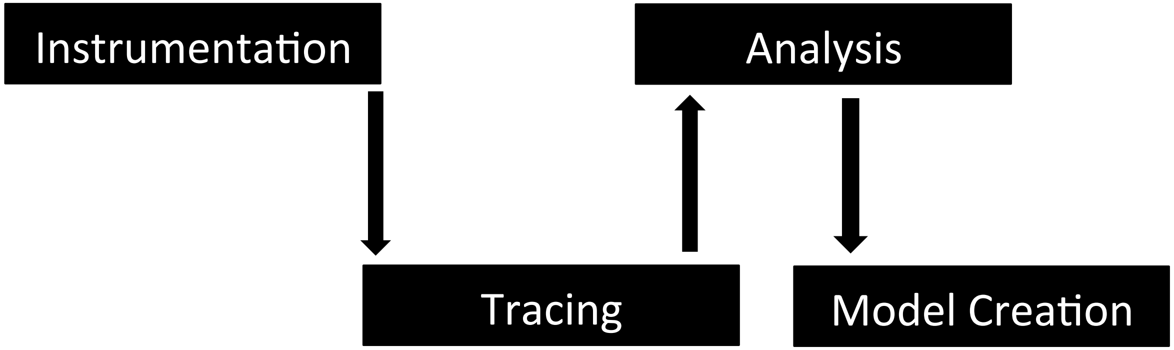
- Packet size, type, …
- Routing table size, …
- Bus state, …

Capture events for when and how behaviour changes

Step 4: Create Models

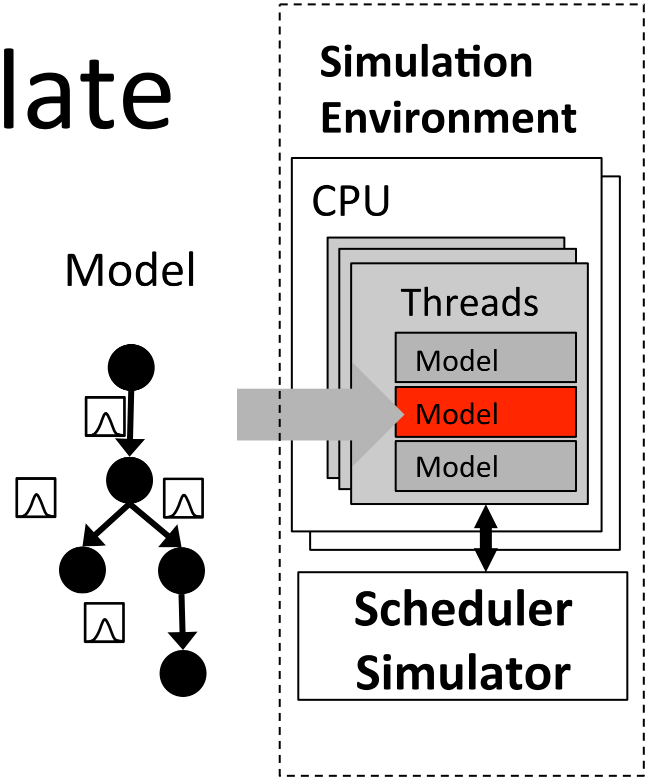
- Overall performance affected by multi-threading
  - Workload
  - Scheduling policy
  - Synchronization





Step 5: Simulate

- Overall performance affected by mutli-threading
  - Workload
  - Scheduling policy
  - Synchronization
- Simulation:
  - Execute models in threads and interrupts
  - Uses scheduler simulator



Model

Simulation Environment

CPU

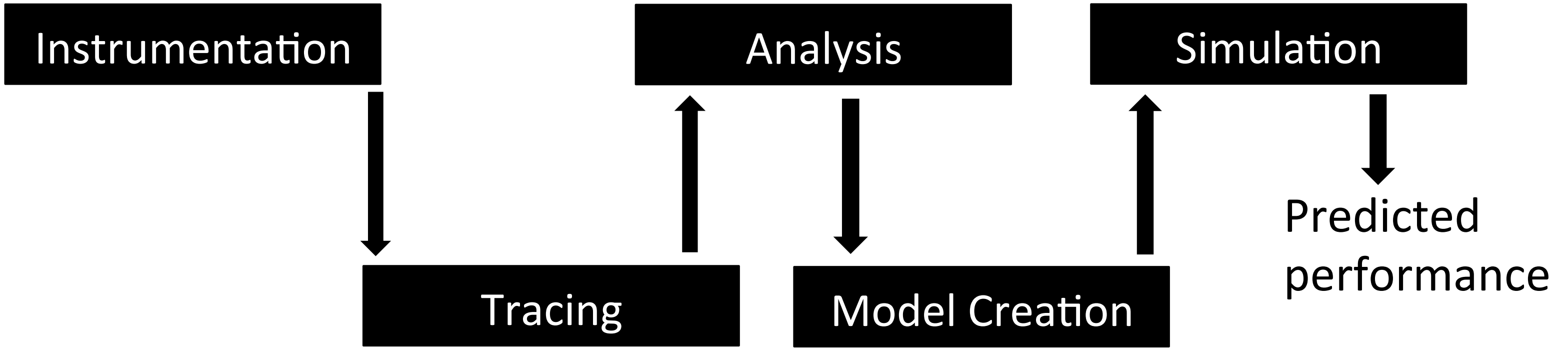
Threads

Model

Model

Model

Scheduler Simulator



Instrumentation

Tracing

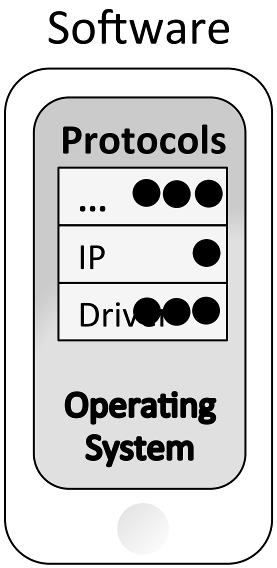
Analysis

Model Creation

Simulation

Predicted performance

Instrumentation



Instrumentation

Capture important events:

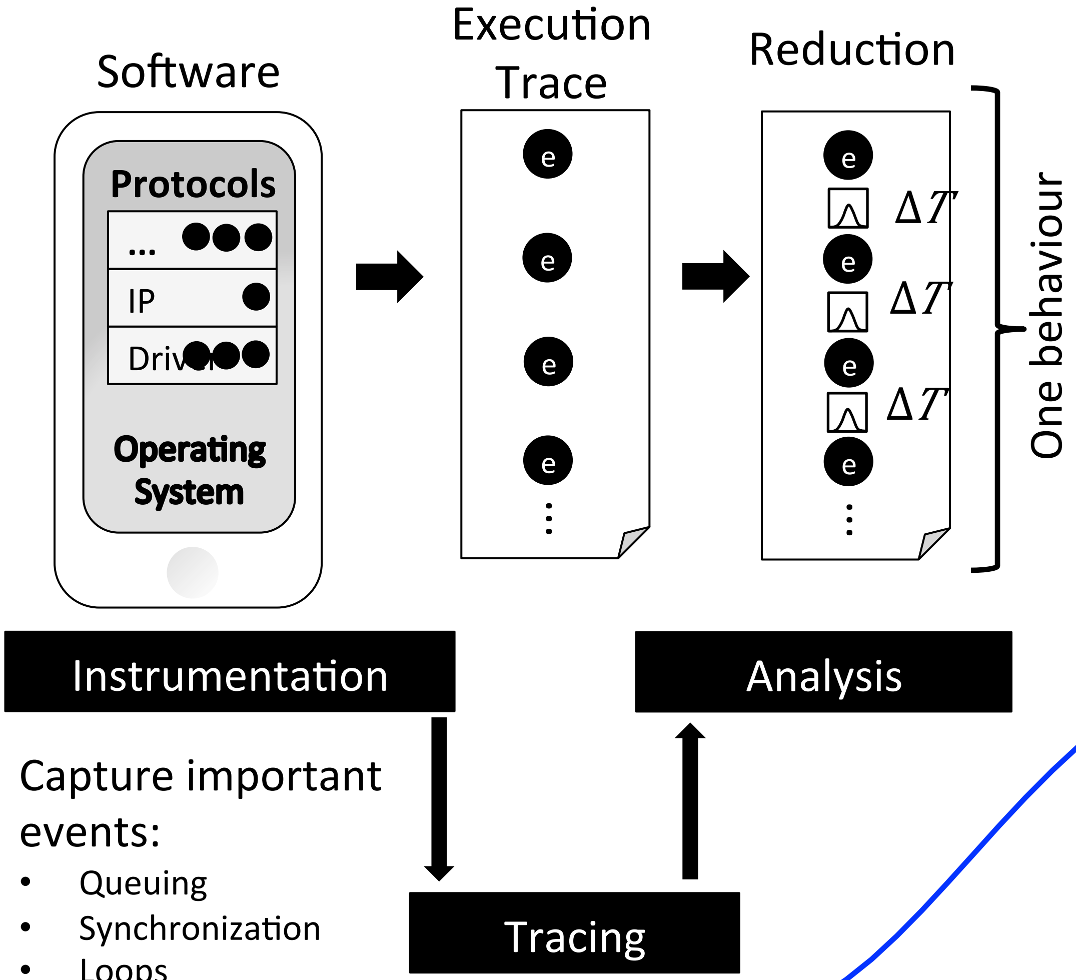
- Queuing
- Synchronization
- Loops
- Interactions with HW
- …

Example:

```
…
pkt = pktq_mdeq(&bus->txq, tx_prec_map, &prec_out);

trace(pkt, DRIVER_TXQ, QUEUE_DEQUEUE);
…
```

Store event in tracefile

Capturing Change in Behaviour



Capture important events:

- Queuing
- Synchronization
- Loops
- Interactions with HW
- **Impact of Context**

Behaviour affected by input data,e.g.,:

- Packet size, type, …
- Routing table size, …
- Bus state, …

= Many behaviours!

Example:

```
void fetchFromNIC(Packet *pkt)
{
  trace(size(pkt));
  if(size(pkt) > 500) {
    DMATranferNIC(pkt);
    condition_wait(dma);
  }
  else {
    DMATranferNIC (pkt);
    while(!dma_done) {};
  }
}
```