

# UDI Network Drivers

## Network Interface Metalanguage

<http://www.sco.com/forum1999/conference/developfast/F13>

**Barry Feild**

**SCO Server Products Group**

E-mail: [barryf@sco.com](mailto:barryf@sco.com)



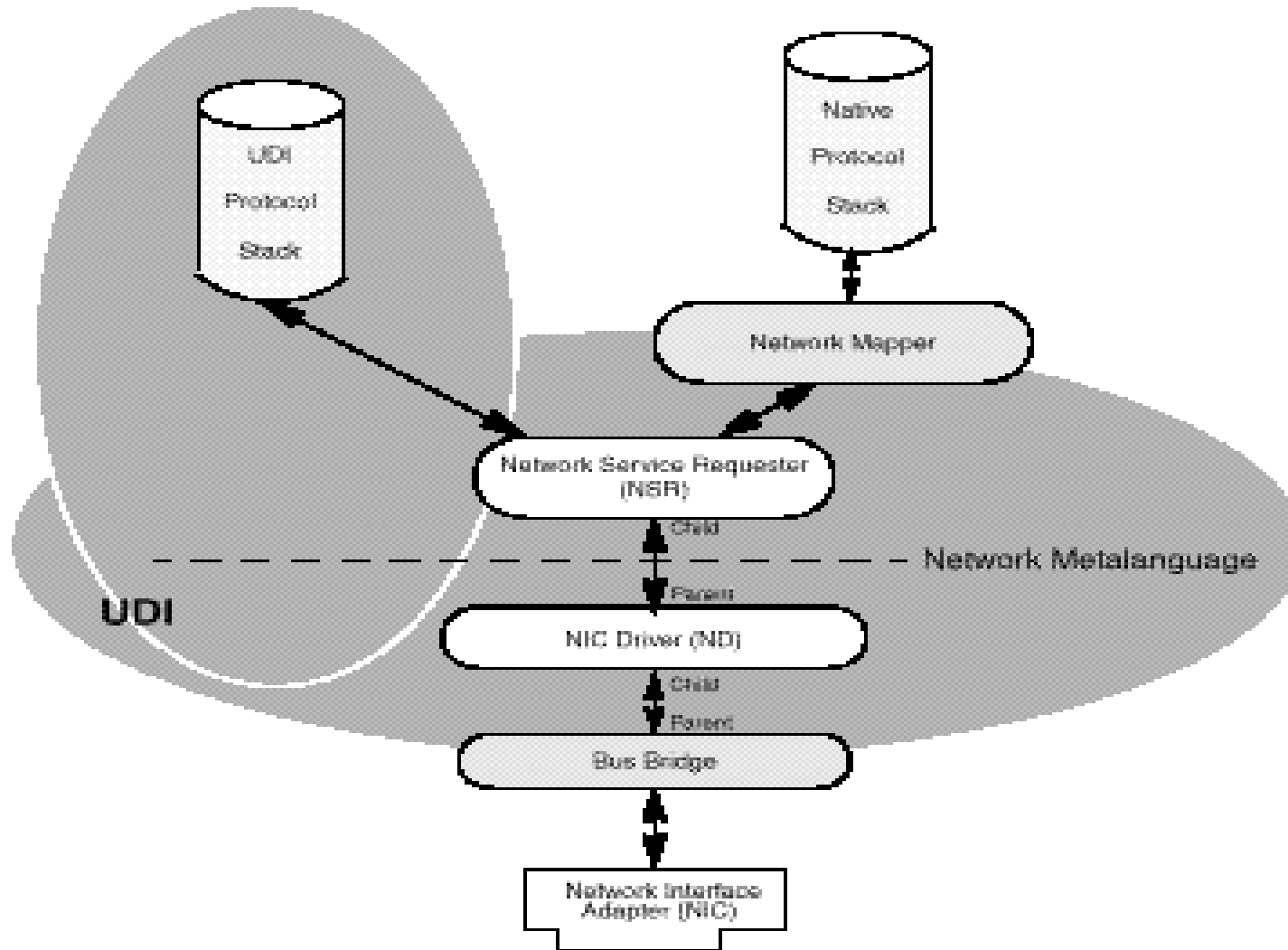
# Agenda

- **SCO UDI Network Driver Architecture**
- **Network Interface Metalanguage Features**
- **Network Driver Initialization**
- **Network Driver Operations**
- **Functional Comparison of MDI and UDI**



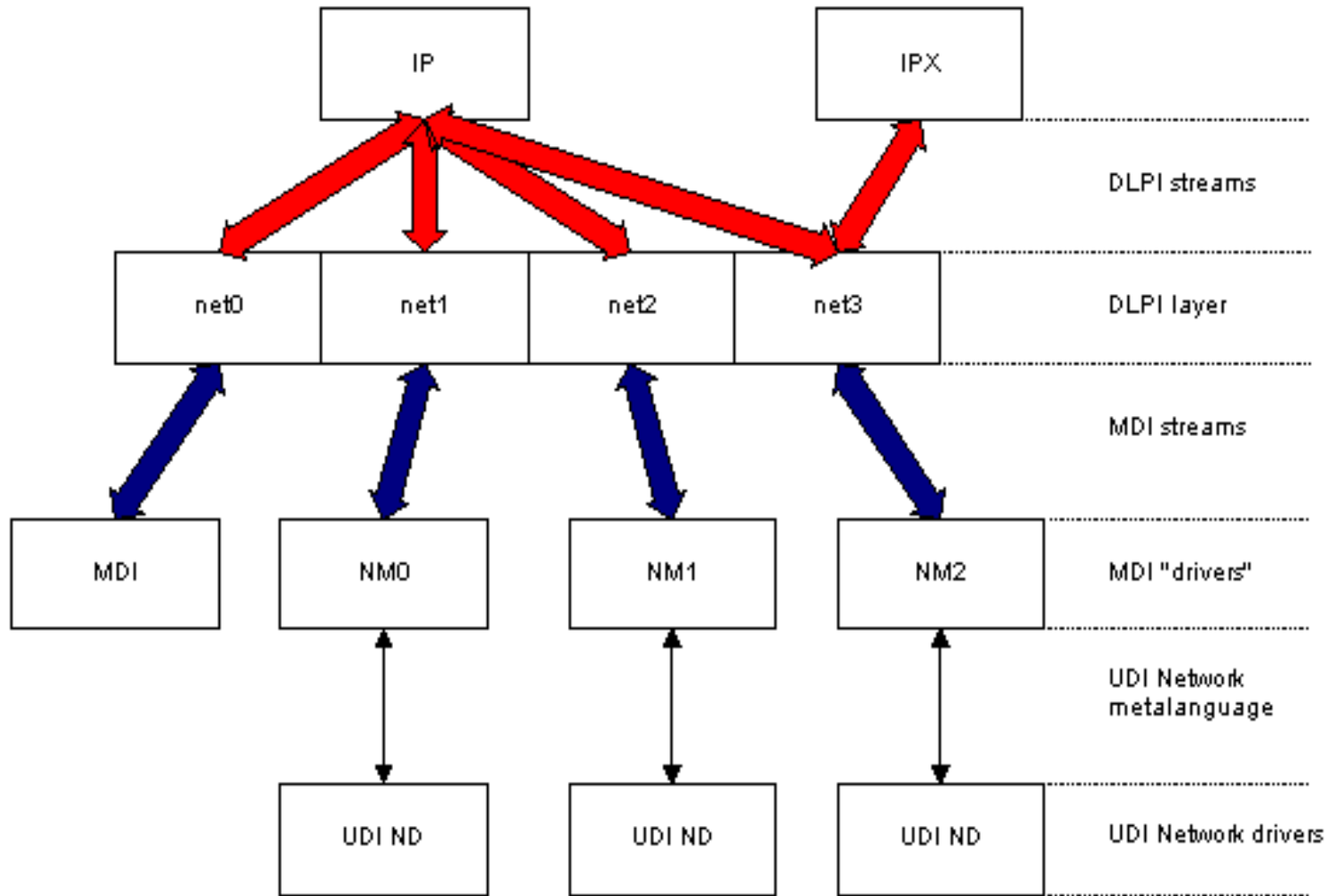
# UDI Network Driver Architecture

## UDI Networking Environment



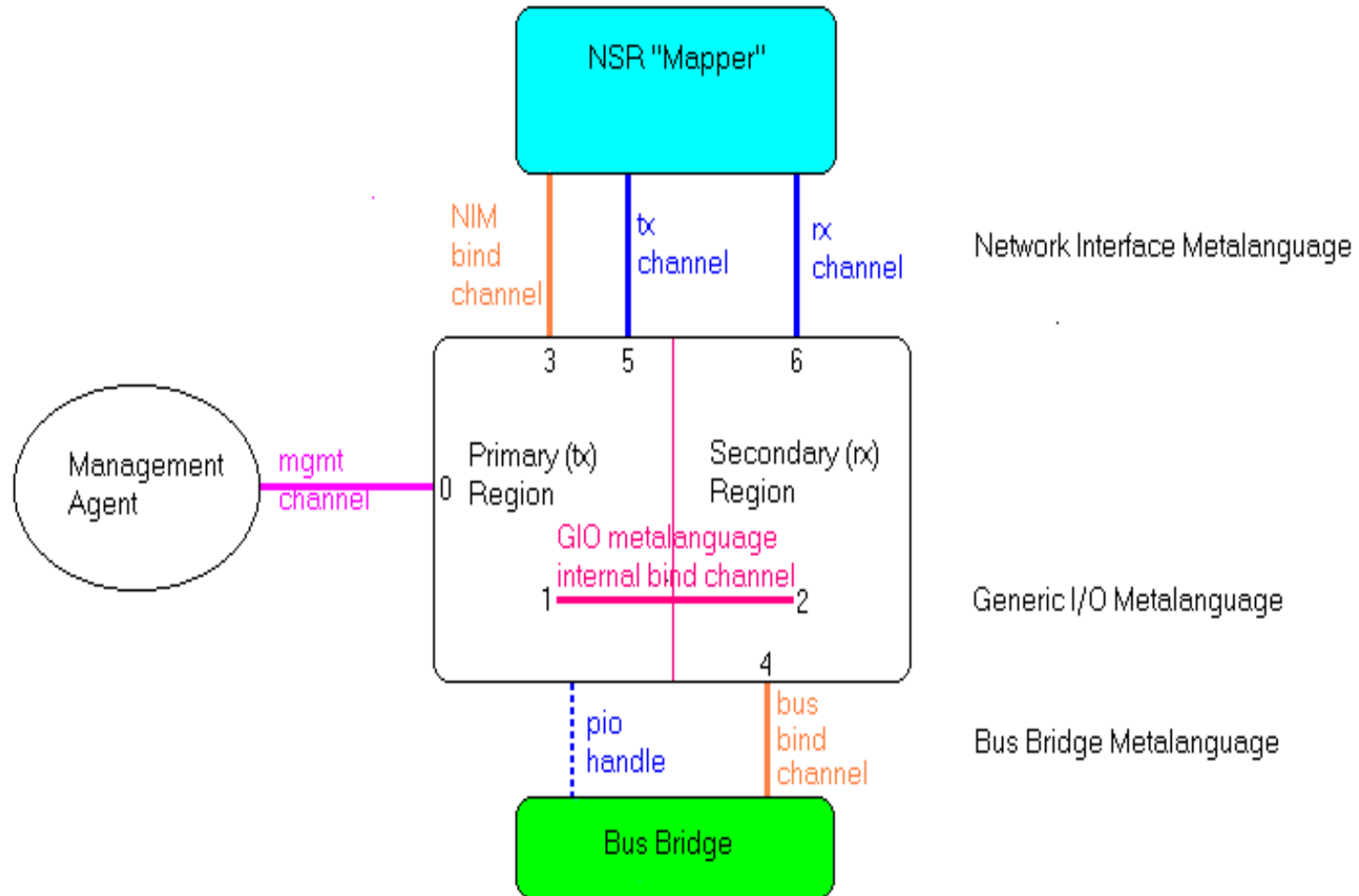
# SCO Network Driver Architecture

## Initial Release



# SCO Network Driver Architecture

shrk UDI Driver



# Network Interface Metalanguage

## UDI and Native Network Driver Comparison

- **UDI Advantages**

- Common programming model
- Portability
- Binary compatibility
- Implicit MP synchronization
- Simple ND interface

- **UDI Challenges**

- New technology



# Network Interface Metalanguage

## Network Driver/Network Service Requestor Interface

- **NIM: a universal set of connectionless network-related functions**
- **ND's primary task is send/receive data**
- **Single NSR-ND Interface active at any given time; separate channels are defined for:**
  - control/status
  - transmit data
  - receive data



# Network Interface Metalanguage

## Network Service Requestor Responsibilities

- **Builds datalink headers for transmit packets**
- **Parse datalink headers on receive packets; NSR may perform SAP de-multiplexing**
- **Supports various MAC address lengths up to 20 bytes; not hard coded at a fixed size (such as 6 bytes)**



# Network Interface Metalanguage

## NSR-ND Channels

- **Control Channel**
  - Bind/Unbind ND and NSR
  - Network Driver control operations (link state, MAC address registration, multicasting, statistics, etc.)
- **Receive Data Channel**
- **Transmit Data Channel**



# Network Interface Metalanguage

## Flow Control

- **Transmit Flow Control**

- Tx control blocks owned by ND, loaned to NSR
- ND returns control blocks to NSR on tx complete
- NSR never allocates transmit blocks

- **Receive Flow Control**

- Rx control blocks owned by NSR, filled by ND
- ND never allocates receive buffers
- Rx buffers may be recycled by NSR



# Network Interface Metalanguage

## Hardware Checksum Offloads

- **Buffer tags used to support CRC offload**
  - `udi_buf_tag_compute()`
  - `udi_buf_tag_apply()`
- **Transmit checksum capability handled separately from Receive checksum capability**
- **Buffer tags may be used for arbitrary per-buffer data**



# Network Driver Initialization

## Static Driver Properties

- **Static Driver Properties**
  - Region Declarations
  - Parent/Child/Internal bind\_ops Declarations
  - Custom Declarations
    - » Adapter-specific configuration parameters
    - » ND reads these attributes on link-enable requests using `udi_instance_attr_get()`



# Network Driver Initialization

## Network Driver Channel Ops Vector Registration

- **Control channel ops - udi\_nd\_ctrl\_ops\_t**
  - channel events
  - bind/unbind/enable/disable
  - control requests (MAC addrs, multicasting, stats)
- **Transmit channel ops - udi\_nd\_tx\_ops\_t**
  - normal/expedited transmit requests
- **Receive channel ops - udi\_nd\_rx\_ops\_t**
  - NSR gives receive control blocks/buffers to ND.



# Network Driver Initialization

## UDI Control Block Registration

- **Control blocks (and Ops vectors) registered in udi\_init\_info structures:**
  - udi\_cb\_init\_t
  - udi\_ops\_init\_t
- **All channel operations are done with appropriate ctrl or transfer control block**
- **All types of control blocks may be allocated internally with udi\_cb\_alloc()**



# Network Driver Initialization

## Network Driver Instantiation

- **Parent driver (Bus Bridge) instantiated, mgmt channel established; MA sends enumeration request to parent**
- **MA creates ND instance (primary and secondary regions, mgmt channel, internal bind channels) that corresponds to parent enumeration response info**
- **MA issues udi\_usage\_ind() on mgmt channel - first operation on newly instantiated driver instance**



# Network Driver Initialization

## Network Driver Instantiation (continued)

- **MA begins child/parent bind sequence; issues UDI\_CHANNEL\_BOUND on child (ND) end of ND-Bus bind channel**
  - may be in primary region, static or dynamic secondary region
- **ND performs internal initialization, reads instance attributes, binds to parent with udi\_bus\_bind\_req()**



# Network Driver Initialization

## Network Driver Instantiation (continued)

- **Parent (Bus Bridge) processes ND's bind request, issues `udi_bus_bind_ack()`**
- **ND completes initialization, issues `udi_channel_event_complete()` for the parent bind operation to the MA**
- **Process repeated for ND (parent) and NSR (child) using NIM bind operations**



# Network Driver Initialization

## Network Driver Enumeration Attributes

- **Network Driver Enumeration Attributes**
  - if\_num: port instance number (32bit unsigned)
  - if\_media: media type (macro)
  - identifier: media type (string)
  - address\_locator: port instance number (string)
  - physical\_locator: interface MAC address (string)



# Network Driver Initialization

## Network Interface Metalanguage Bind Operation

- **MA creates NSR-ND initial bind channel (control channel) using Network Interface Metalanguage ctrl\_ops role**
  - Network Interface Metalanguage control channel is now established
- **NSR issues ND udi\_nd\_bind\_req() on NSR bind channel**
  - ND propagates constraints (memory requirements for DMA, etc.) to NSR with udi\_constraints\_propagate()



# Network Driver Initialization

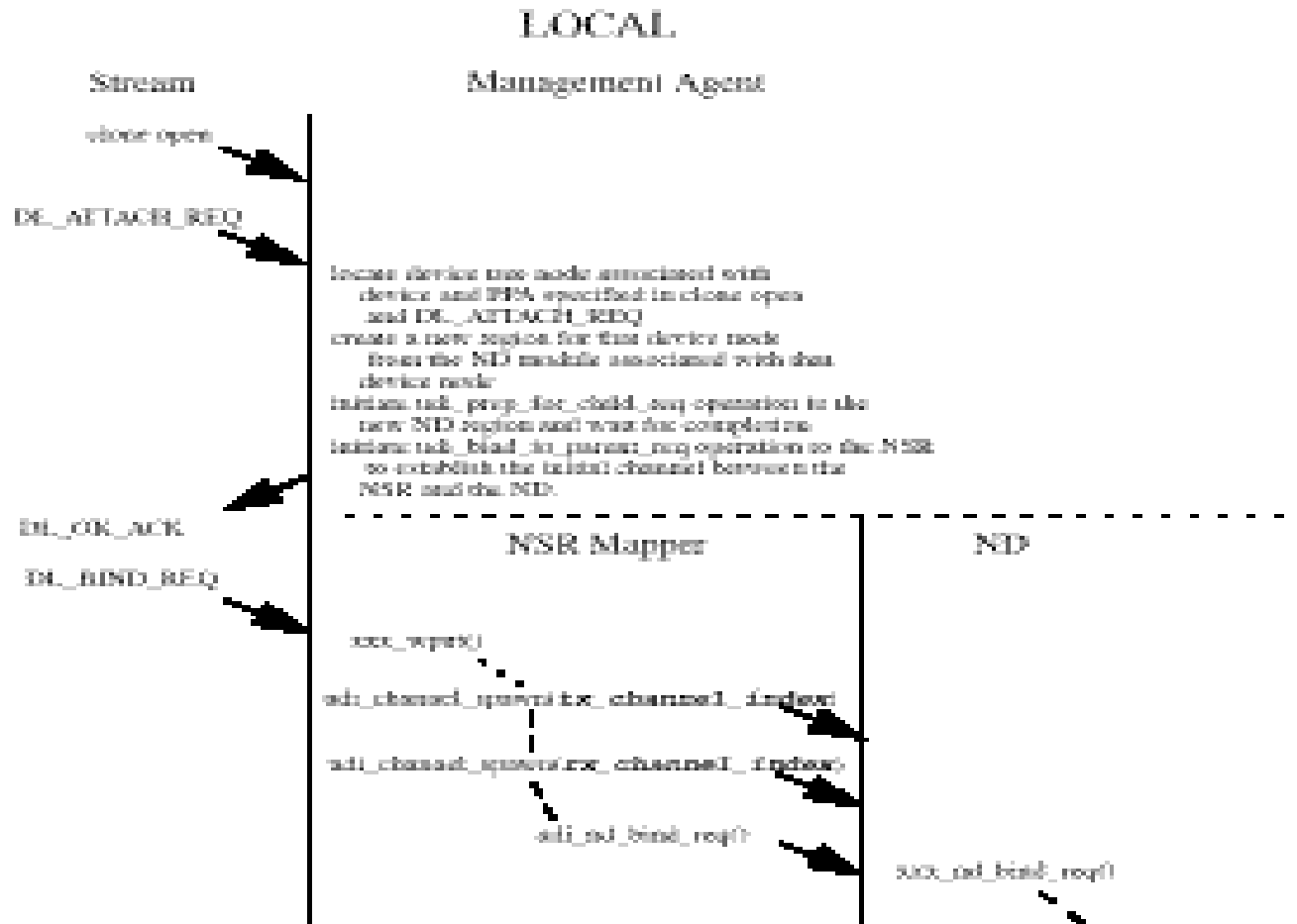
## Network Interface Metalanguage Bind Operation (cont)

- **ND actions on udi\_nd\_bind\_req() (cont)**
  - ND spawns and anchors transmit channel, using transmit channel ops vector index
  - ND spawns and anchors receive channel, using receive channel ops vector index
- **ND acks bind with udi\_nsr\_bind\_ack()**



# Network Driver Initialization

## Connectionless Network Bind Operation





# Network Driver Initialization

## Network Interface Metalanguage Unbind Operation

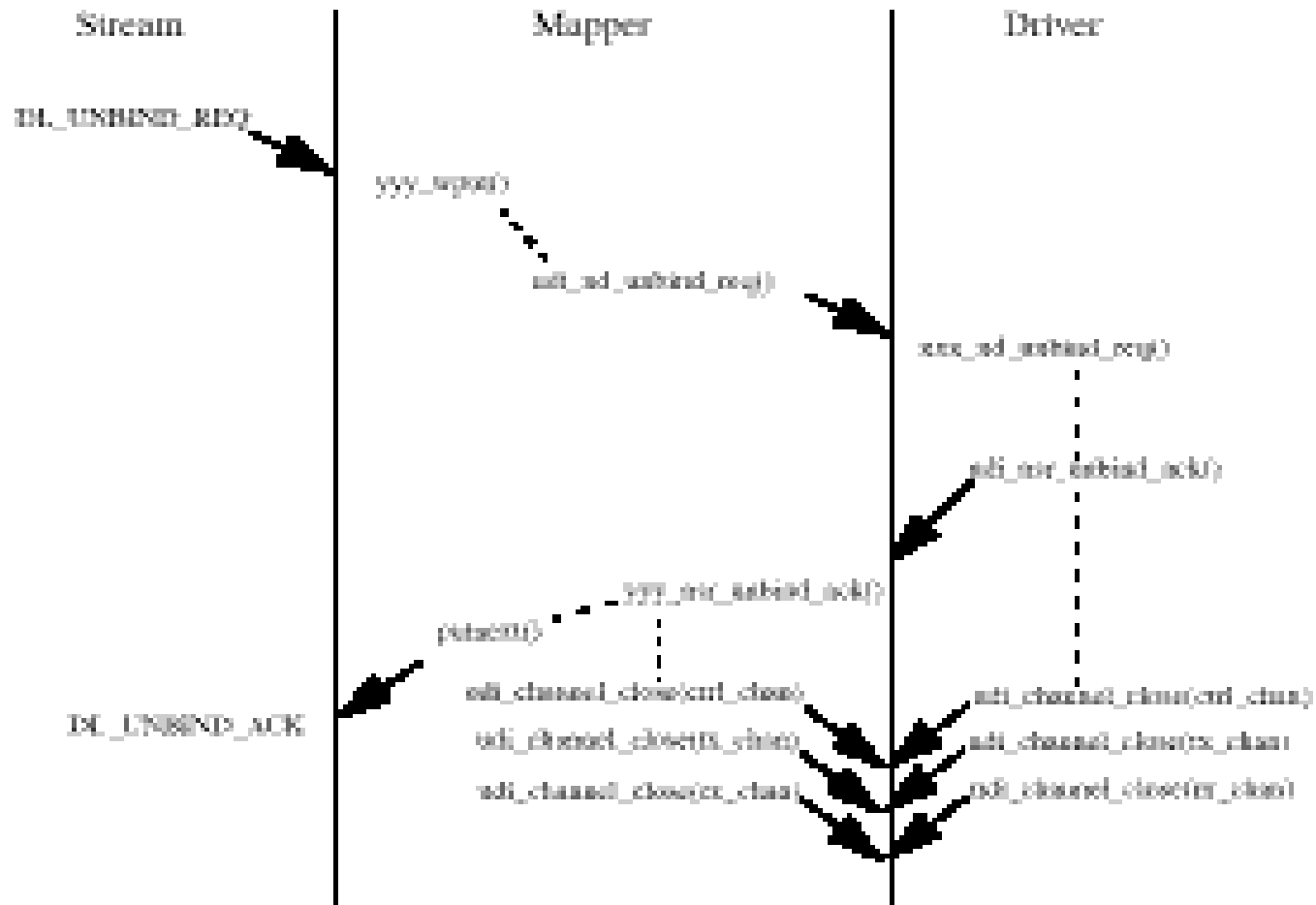
- **Unbind used to detach channels and release resources established during Network Bind operation**
- **ND acks with `udi_nsr_unbind_ack()`**
- **ND closes control and both data channels with `udi_channel_close()`**



# Network Driver Initialization

## Connectionless Network Unbind Operation

LOCAL



# Network Driver Operations

## Network Interface Control

- **Enable network interface link activity**
  - udi\_nd\_enable\_req()
  - udi\_nsr\_enable\_ack()
- **Disable network interface link activity**
  - udi\_nd\_disable\_req() - no ack
- **Indicate link state change**
  - udi\_nsr\_status\_ind()



# Network Driver Operations

## Control and Status Operations

- **Control Requests - udi\_nd\_ctrl\_req()**
  - UDI\_NET\_ADD\_MULTI
  - UDI\_NET\_DEL\_MULTI
  - UDI\_NET\_ALLMULTI\_ON
  - UDI\_NET\_ALLMULTI\_OFF
  - UDI\_NET\_GET\_CURR\_MAC
  - UDI\_NET\_SET\_CURR\_MAC
  - UDI\_NET\_GET\_FACT\_MAC



# Network Driver Operations

## Control and Status Operations (continued)

- **udi\_nd\_ctrl\_req() (continued)**
  - UDI\_NET\_PROMISC\_ON
  - UDI\_NET\_PROMISC\_OFF
  - UDI\_NET\_HW\_RESET
  - UDI\_NET\_BAD\_RXPKT
- **Control Request ack - udi\_nsr\_ctrl\_ack()**



# Network Driver Operations

## Control and Status Operations (continued)

- **Status Indications - udi\_nsr\_status\_ind()**
  - UDI\_NET\_LINK\_UP
  - UDI\_NET\_LINK\_DOWN
  - UDI\_NET\_LINK\_RESET
- **Statistics**
  - udi\_nd\_info\_req()
  - udi\_nsr\_info\_ack()
  - udi\_net\_info\_cb\_t



# Network Driver Operations

## Control and Status Operations (continued)

- **Statistics (cont) - udi\_net\_info\_cb\_t**

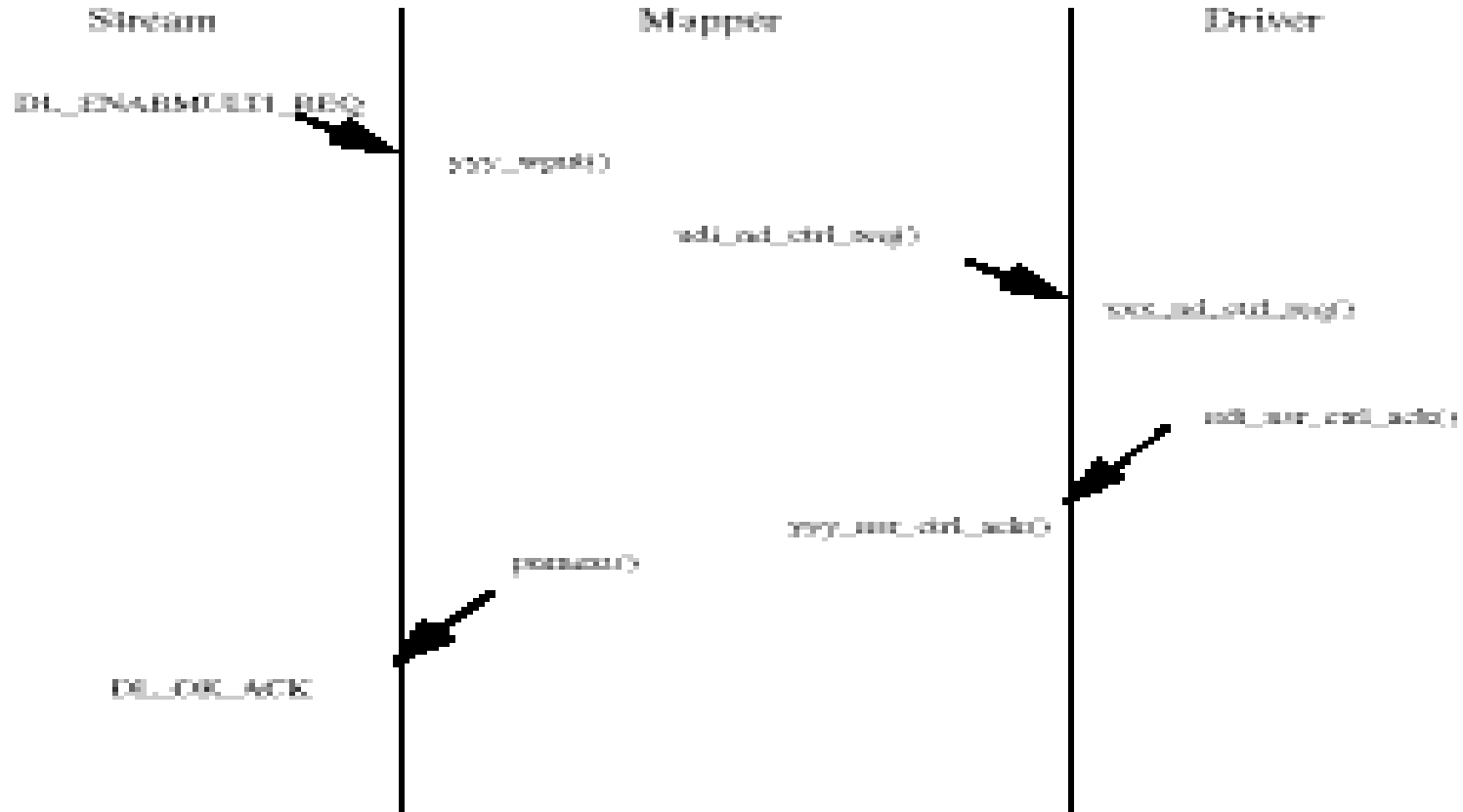
- interface\_is\_active
- link\_is\_active
- is\_full\_duplex
- link\_mbps
- link\_bps
- tx\_packets
- rx\_packets
- tx\_errors
- rx\_errors
- tx\_discards
- rx\_discards
- tx\_underrun
- rx\_overrun
- collisions



# Network Driver Operations

Control Operation (enable multicast address)

LOCAL



# Network Driver Operations

## Data Transfer - Transmit

- **ND owns transmit buffers/control blocks**
  - udi\_cb\_alloc()
- **ND loans transmit control blocks to NSR**
  - udi\_nsr\_tx\_rdy()
- **NSR sends transmit packets to ND**
  - udi\_nd\_tx\_req()
  - udi\_nd\_exp\_tx\_req()



# Network Driver Operations

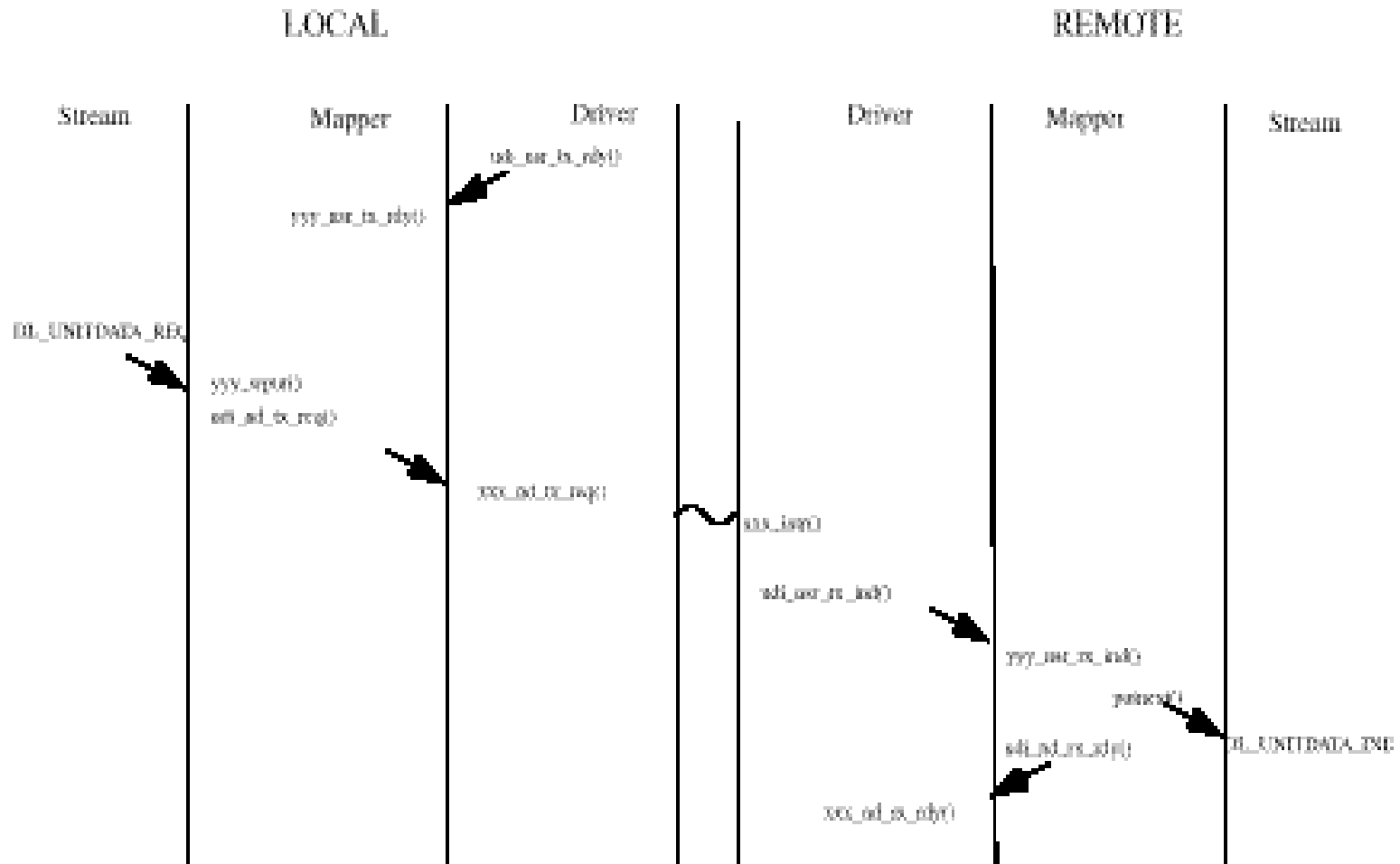
## Data Transfer - Receive

- **NSR owns receive control blocks**
  - udi\_cb\_alloc()
- **NSR loans receive control blocks to ND**
  - udi\_nd\_rx\_rdy()
- **ND sends receive packets to NSR**
  - udi\_nsr\_rx\_ind()
  - udi\_nsr\_exp\_rx\_ind()



# Network Driver Operations

## Connectionless Data Transfer Operations



# MDI/UDI Functional Comparison

## MDI Primitives

- **MDI Primitives**

- MAC\_BIND\_REQ
- MAC\_OK\_ACK
- MAC\_INFO\_REQ
- MAC\_INFO\_ACK
- MAC\_ERROR\_ACK
- MAC\_HWFFAIL\_IND
- MAC\_HWSUSPEND\_IND
- MAC\_HWRESUME\_IND

- **UDI Equivalent**

- udi\_nd\_bind\_req()
- udi\_nd\_bind\_ack()
- udi\_nd\_info\_req()
- udi\_nsr\_info\_rsp()
- udi\_nsr\_bind\_res error
- UDI\_NET\_HW\_RESET
- UDI\_DMGMGT\_SUSPEND
- UDI\_DMGMGT\_RESUME



# MDI/UDI Functional Comparison

## MDI ioctl

- **MDI ioctl**

- MACIOC\_SETALLMCA
- MACIOC\_DELALLMCA
- MACIOC\_SETMCA
- MACIOC\_DELMCA
- MACIOC\_GETADDR
- MACIOC\_GETRADDR
- MACIOC\_SETADDR
- MACIOC\_GETSTAT
- MACIOC\_PROMISC

- **UDI Equivalent ctrl\_ops**

- UDI\_NET\_ALLMULTI\_ON
- UDI\_NET\_ALLMULTI\_OFF
- UDI\_NET\_ADD\_MULTI
- UDI\_NET\_DEL\_MULTI
- UDI\_NET\_GET\_CURR\_MAC
- UDI\_NET\_GET\_FACT\_MAC
- UDI\_NET\_SET\_CURR\_MAC
- udi\_nd\_info\_req()
- UDI\_NET\_PROMISC\_ON/OFF



# MDI/UDI Functional Comparison

MDI ioctl (continued)

- **MDI ioctls with no UDI equivalent**
  - MACIOC\_SETSTAT
  - MACIOC\_CLRSTAT

