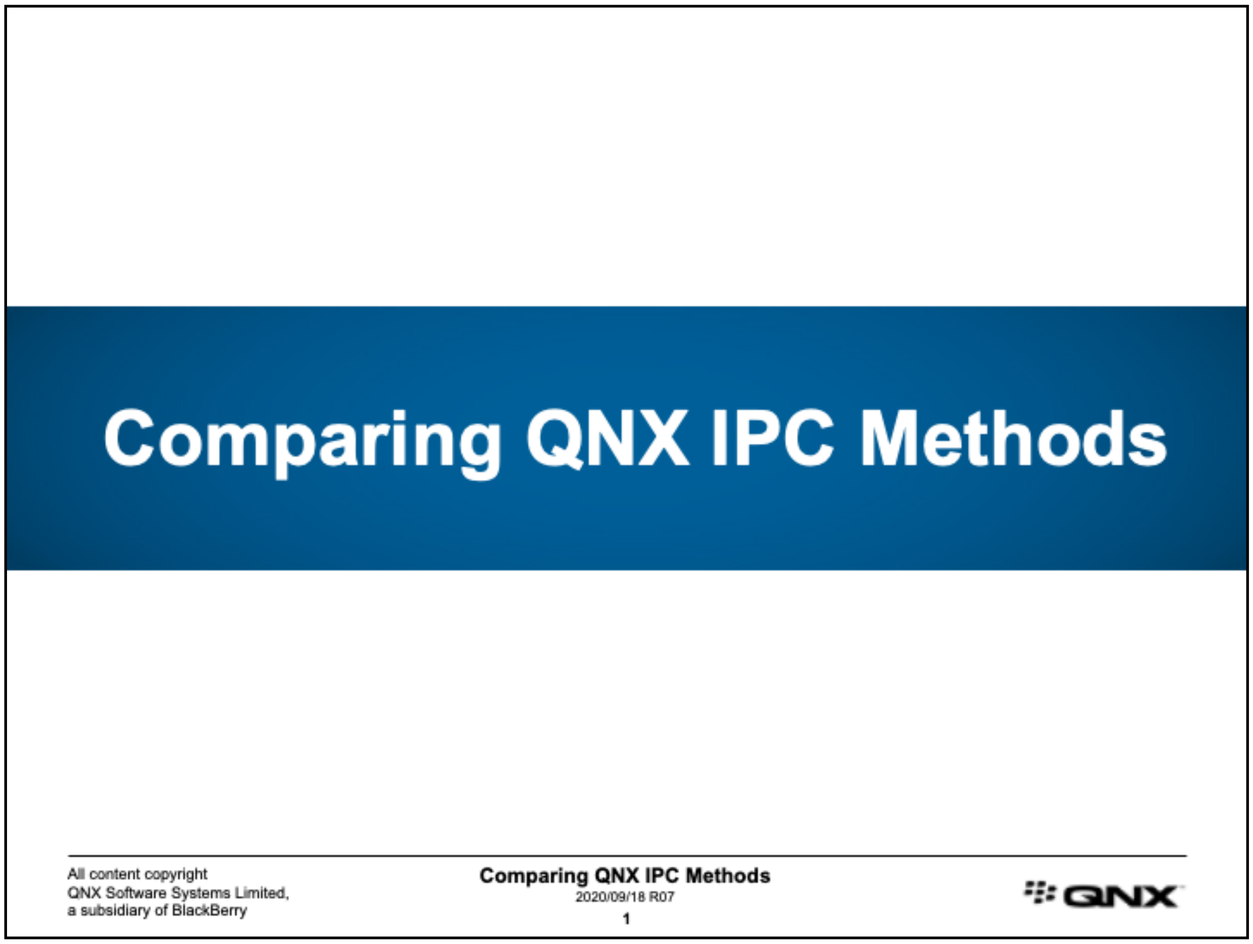
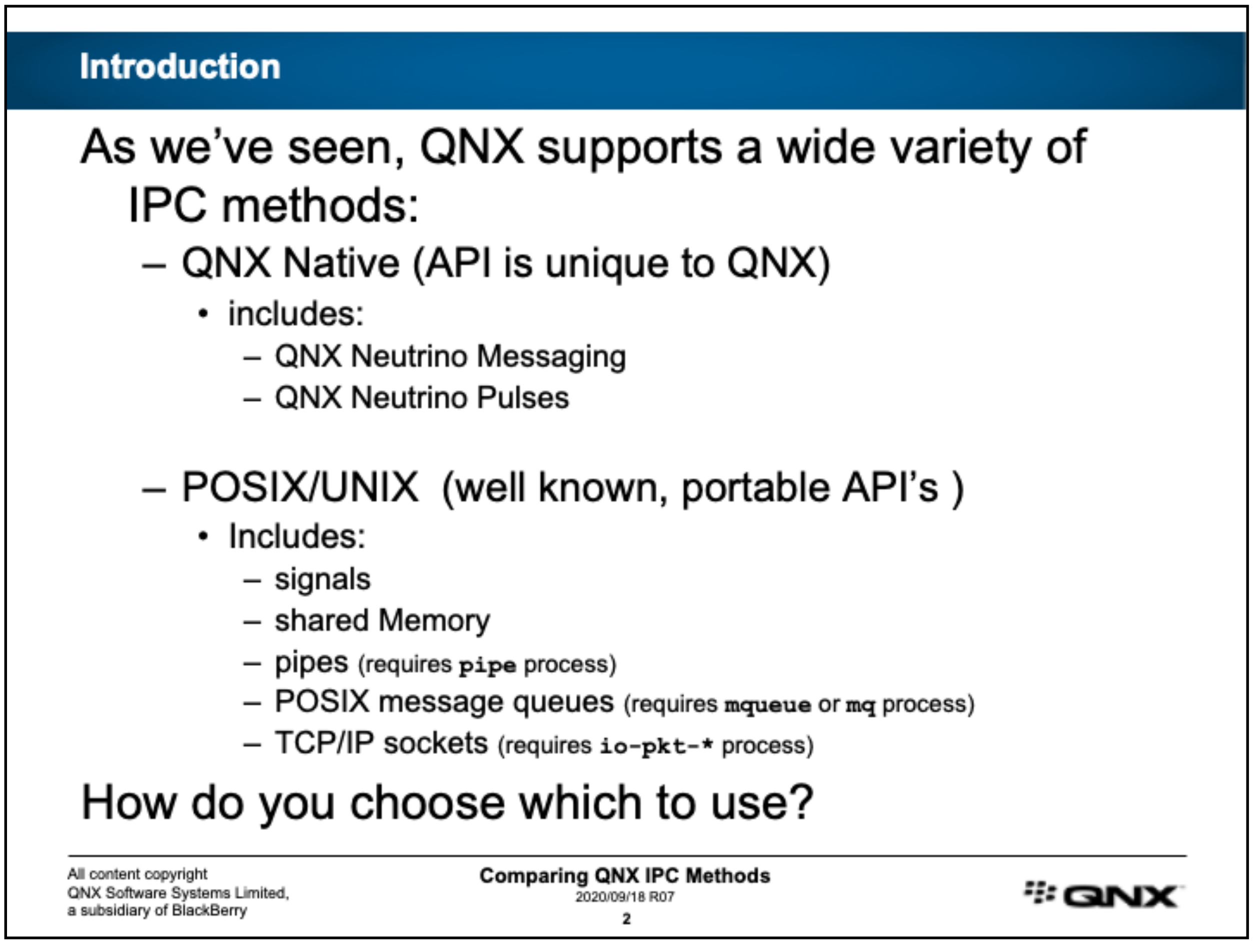


Comparing QNX IPC Methods

NOTES:



Introduction

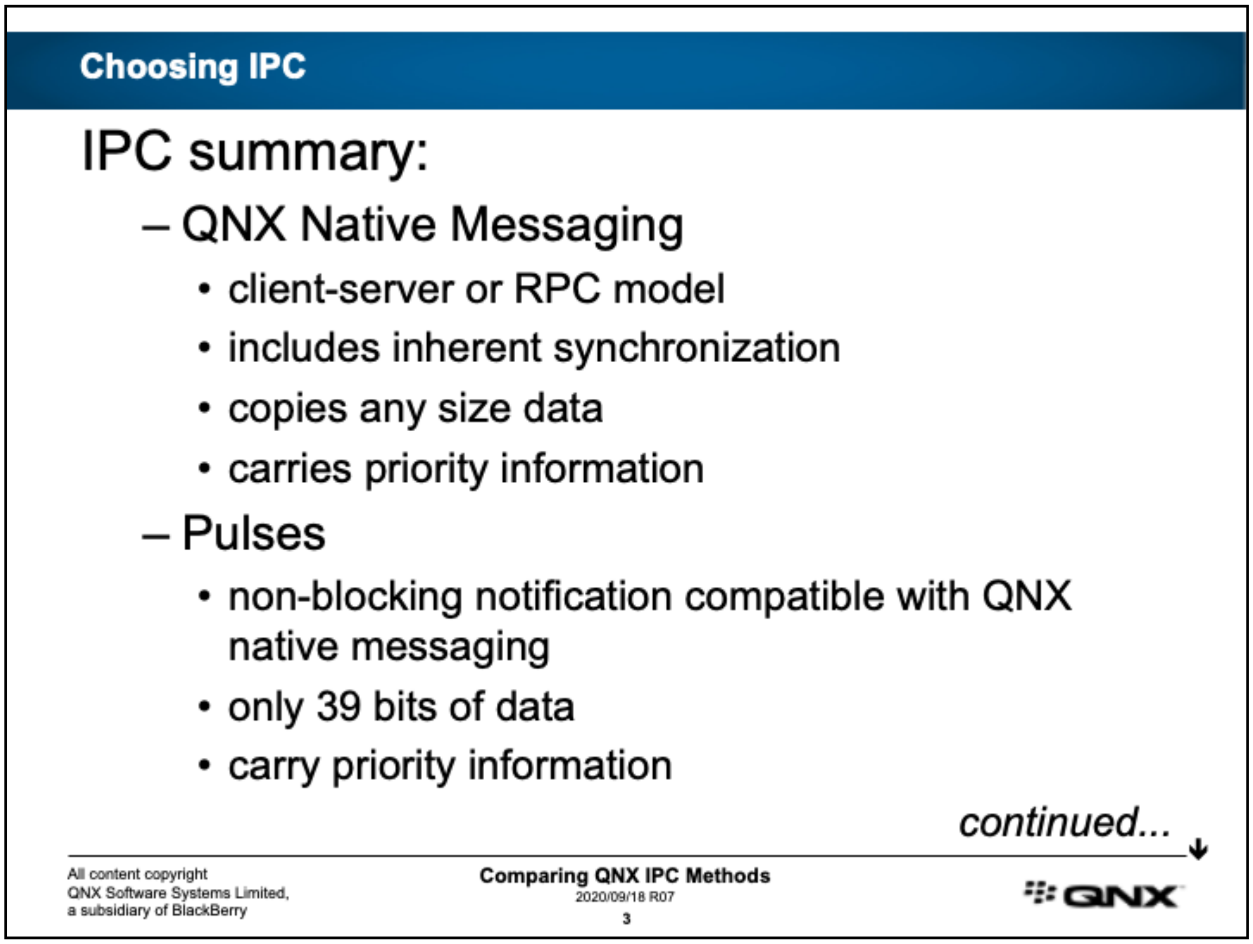
As we've seen, QNX supports a wide variety of IPC methods:

- QNX Native (API is unique to QNX)
  - includes:
    - QNX Neutrino Messaging
    - QNX Neutrino Pulses
- POSIX/UNIX (well known, portable API's)
  - Includes:
    - signals
    - shared Memory
    - pipes (requires **`pipe`** process)
    - POSIX message queues (requires **`mqueue`** or **`mq`** process)
    - TCP/IP sockets (requires **`io-pkt-*`** process)

How do you choose which to use?

NOTES:

API: Application Programming Interface



Choosing IPC

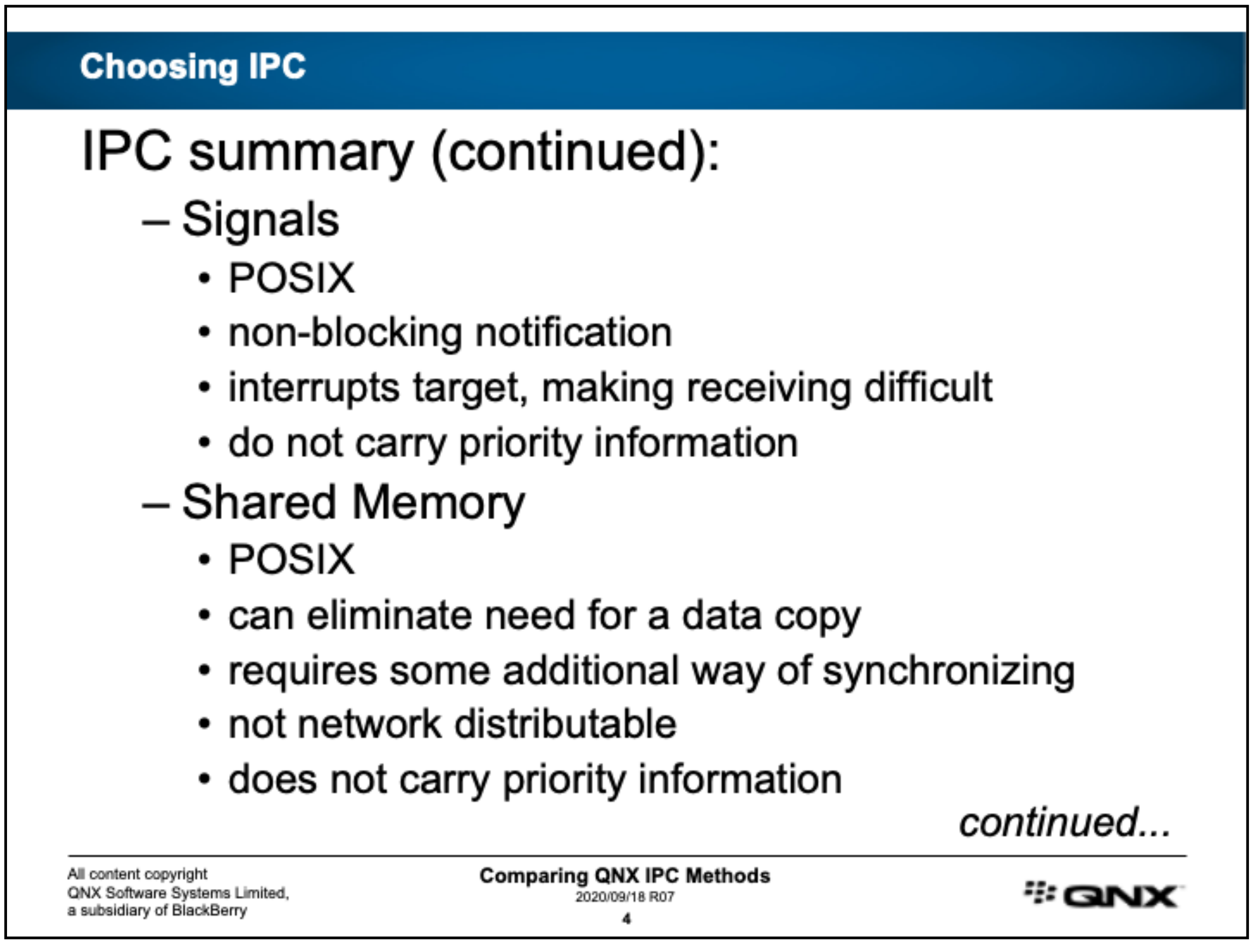
IPC summary:

- QNX Native Messaging
  - client-server or RPC model
  - includes inherent synchronization
  - copies any size data
  - carries priority information
- Pulses
  - non-blocking notification compatible with QNX native messaging
  - only 39 bits of data
  - carry priority information

continued...

NOTES:

The fact that they carry priority information means that priority inversion issues are addressed.



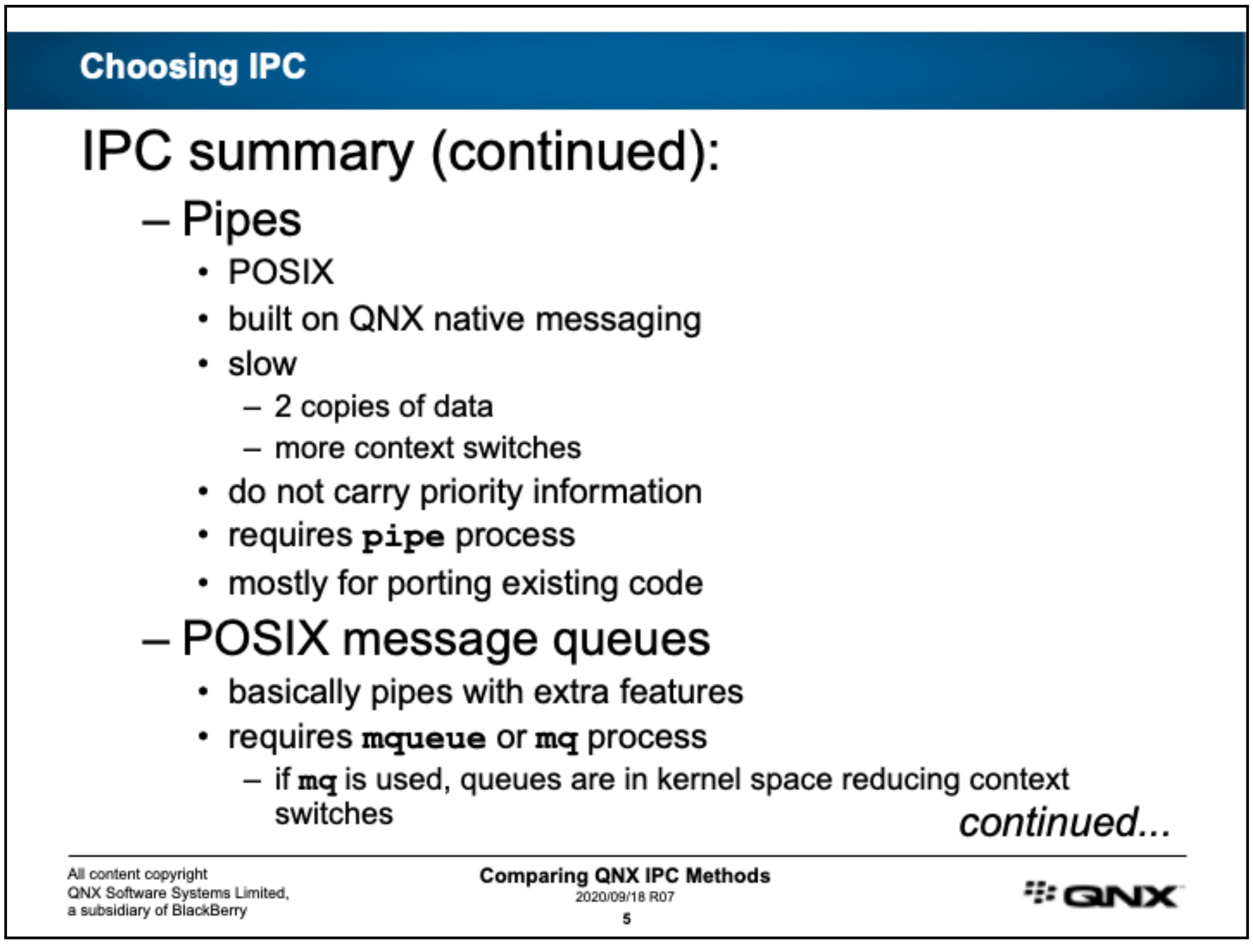
Choosing IPC

IPC summary (continued):

- Signals
  - POSIX
  - non-blocking notification
  - interrupts target, making receiving difficult
  - do not carry priority information
- Shared Memory
  - POSIX
  - can eliminate need for a data copy
  - requires some additional way of synchronizing
  - not network distributable
  - does not carry priority information

continued...

NOTES:

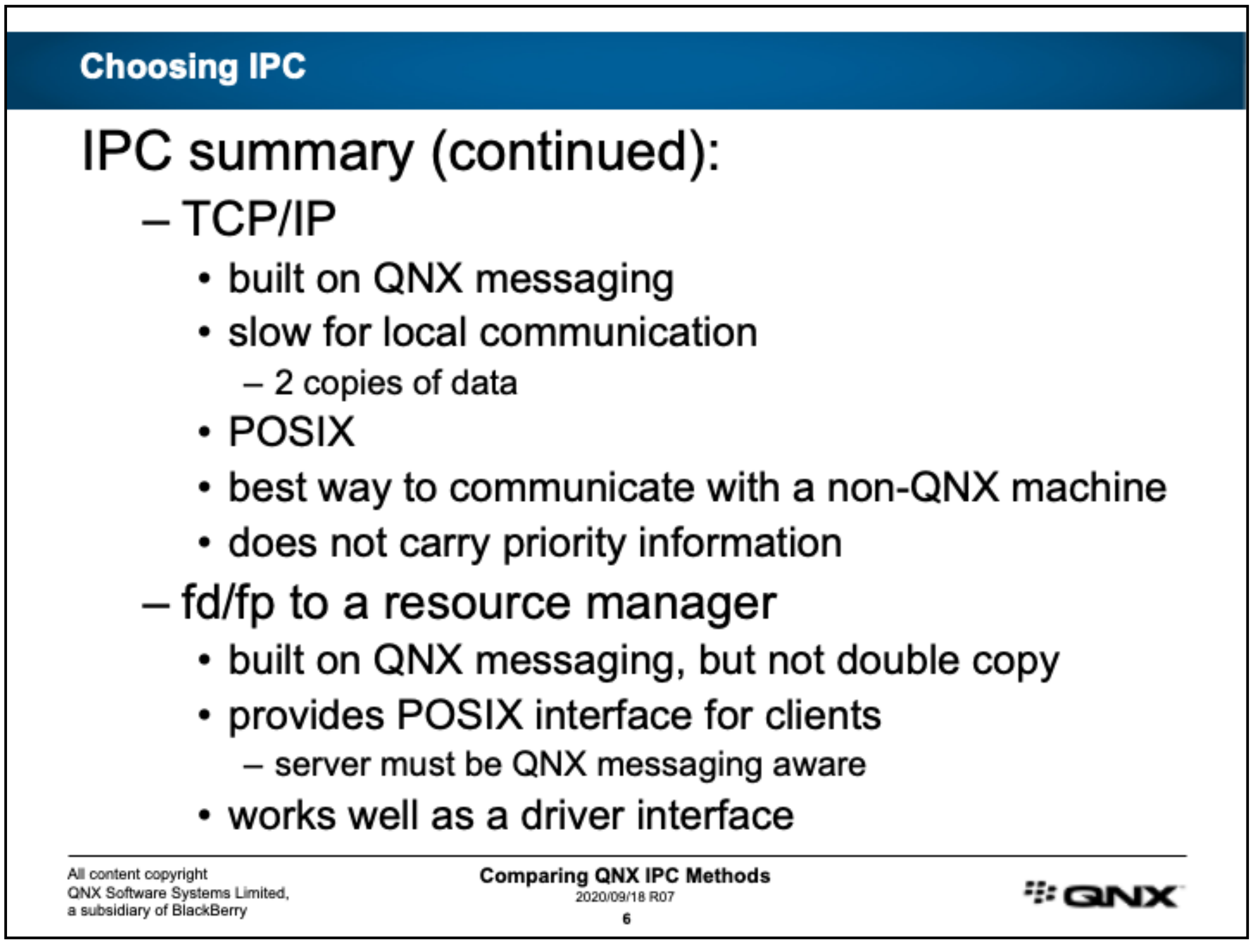


IPC summary (continued):

- Pipes
  - POSIX
  - built on QNX native messaging
  - slow
    - 2 copies of data
    - more context switches
  - do not carry priority information
  - requires **`pipe`** process
  - mostly for porting existing code
- POSIX message queues
  - basically pipes with extra features
  - requires **`mqueue`** or **`mq`** process
    - if **`mq`** is used, queues are in kernel space reducing context switches

continued...

NOTES:

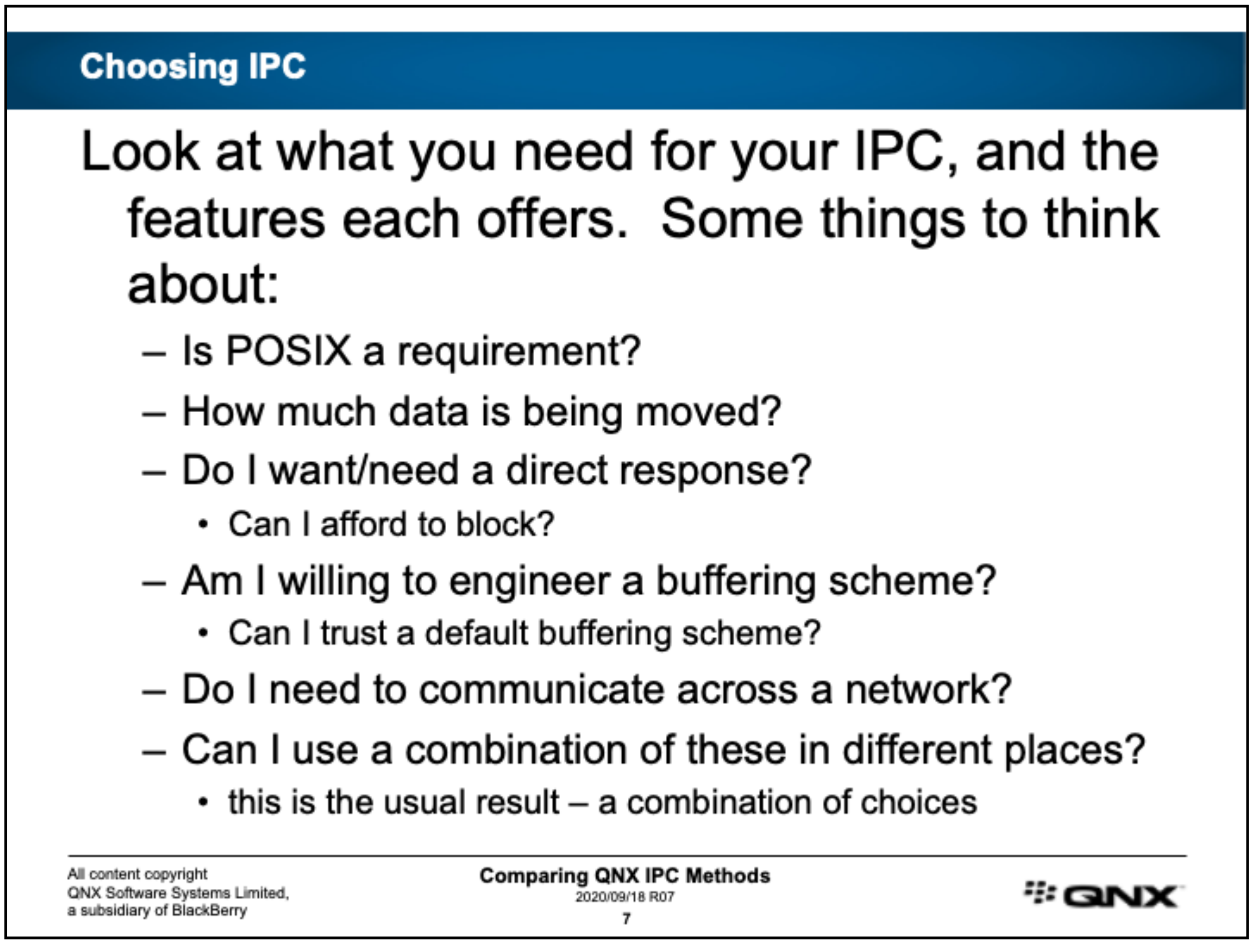


Choosing IPC

IPC summary (continued):

- TCP/IP
  - built on QNX messaging
  - slow for local communication
    - 2 copies of data
  - POSIX
  - best way to communicate with a non-QNX machine
  - does not carry priority information
- fd/fp to a resource manager
  - built on QNX messaging, but not double copy
  - provides POSIX interface for clients
    - server must be QNX messaging aware
  - works well as a driver interface

NOTES:



Choosing IPC

Look at what you need for your IPC, and the features each offers. Some things to think about:

- Is POSIX a requirement?
- How much data is being moved?
- Do I want/need a direct response?
  - Can I afford to block?
- Am I willing to engineer a buffering scheme?
  - Can I trust a default buffering scheme?
- Do I need to communicate across a network?
- Can I use a combination of these in different places?
  - this is the usual result – a combination of choices

NOTES: