1.Netty

NIO JBOSS FTP,SMTP,HTTP

1. API
2.
3. ·
4.

Elasticsearch Hadoop avro Dubbo Netty

BIO

N:N ,CPU

Tomcat 7 BIO 7 NIO NIO

IO 模式

-> -> ->

-> ->

2. 五种 I/O 模型

I/O IO IO IO IO IO 4 IO

copy

3. 阻塞 IO

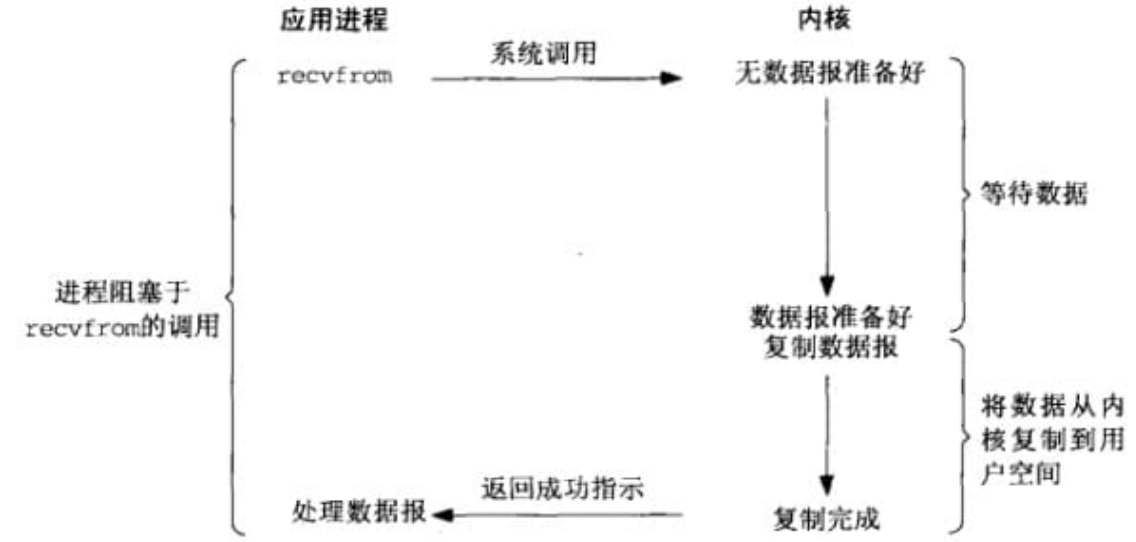


图6-1　阻塞式I/O模型

4. 非阻塞 IO

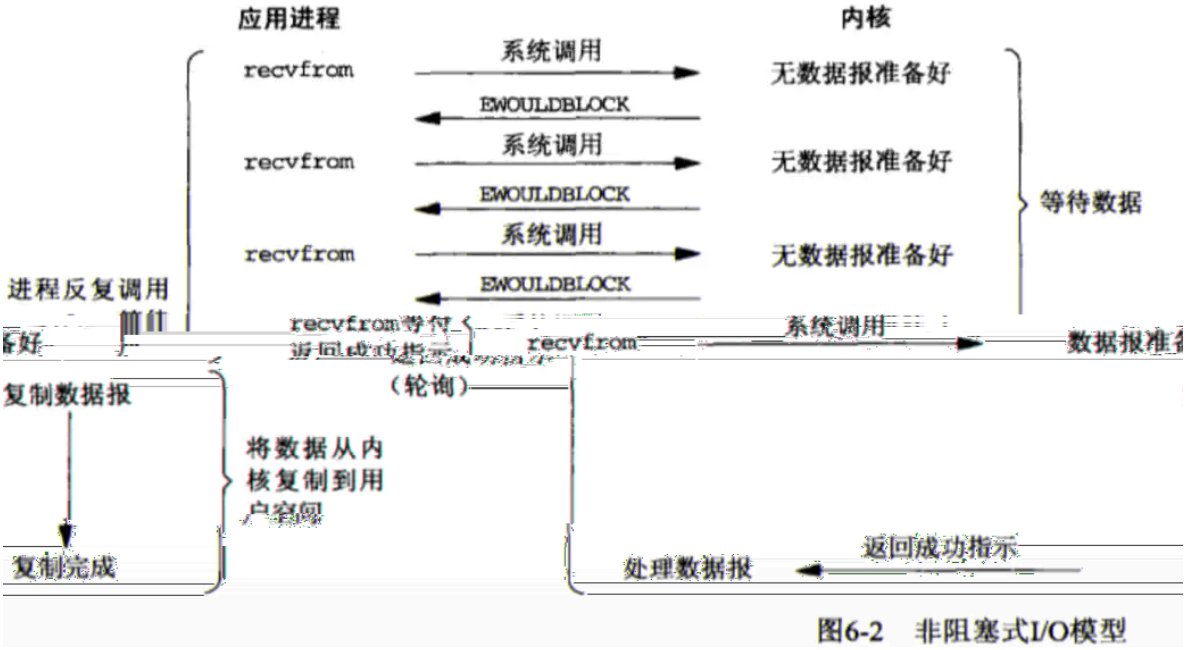


图6-2　非阻塞式I/O模型

5. IO 多路复用

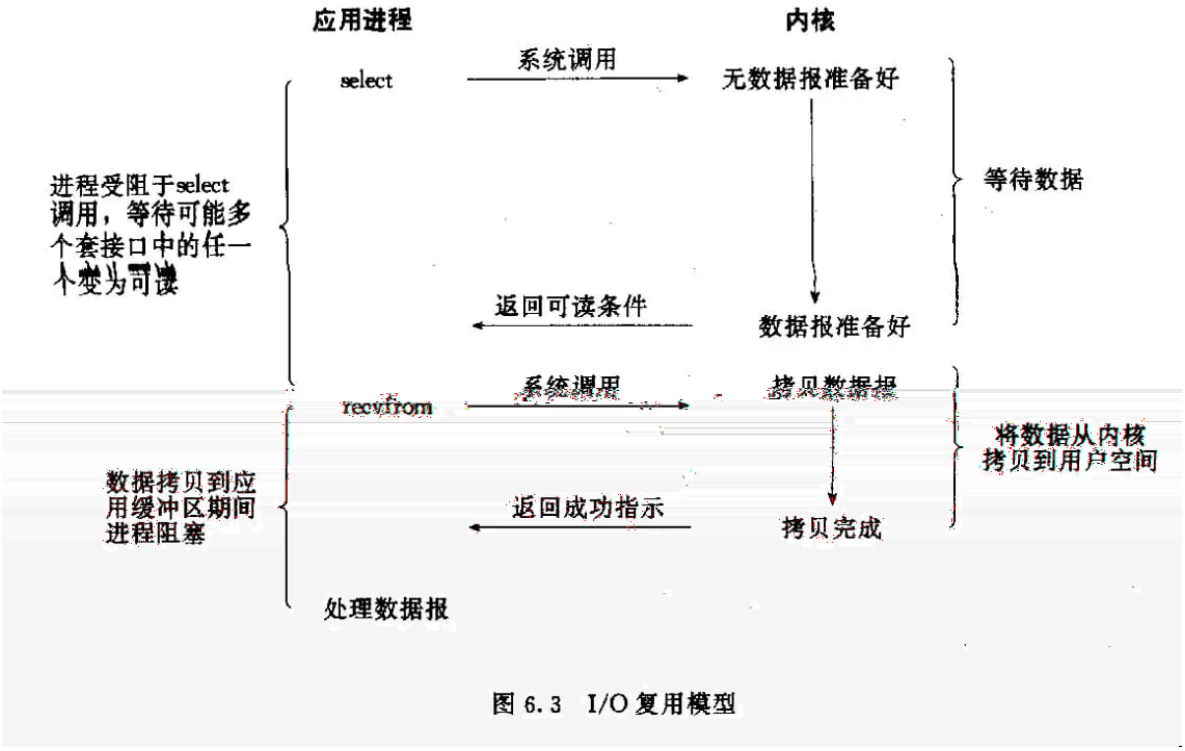


图 6.3 I/O 复用模型

connection select recvfrom socket non-blocking select block socket

\+ IO select

信号驱动

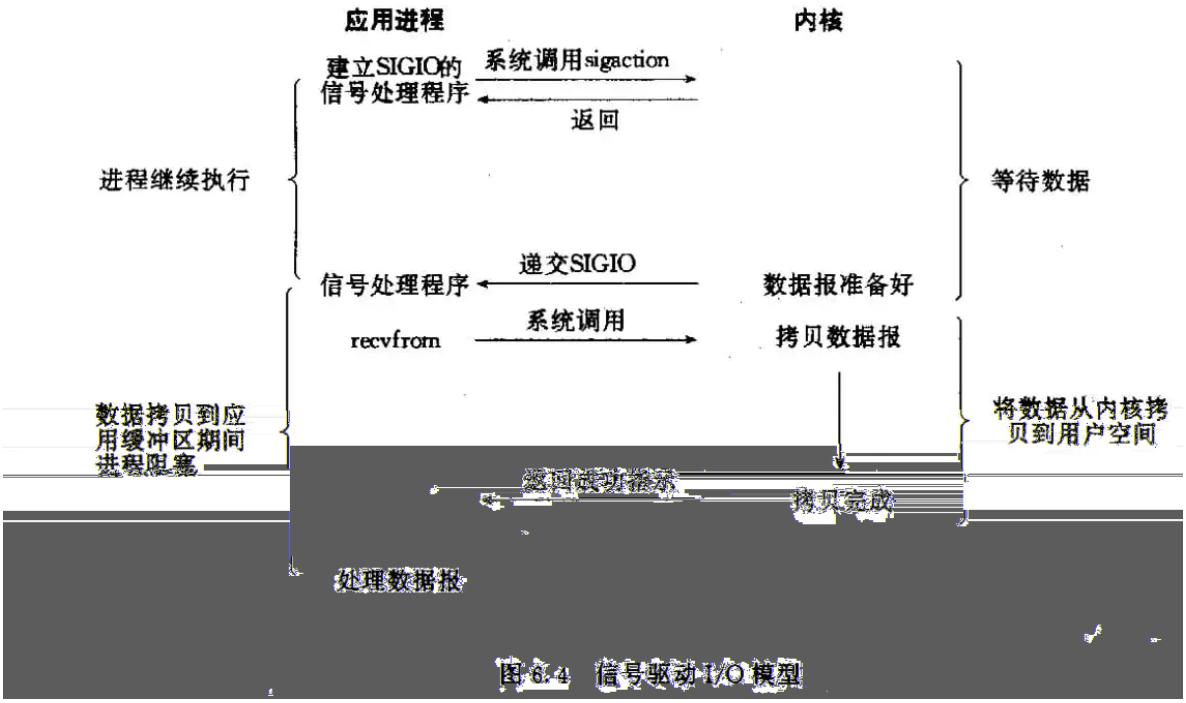
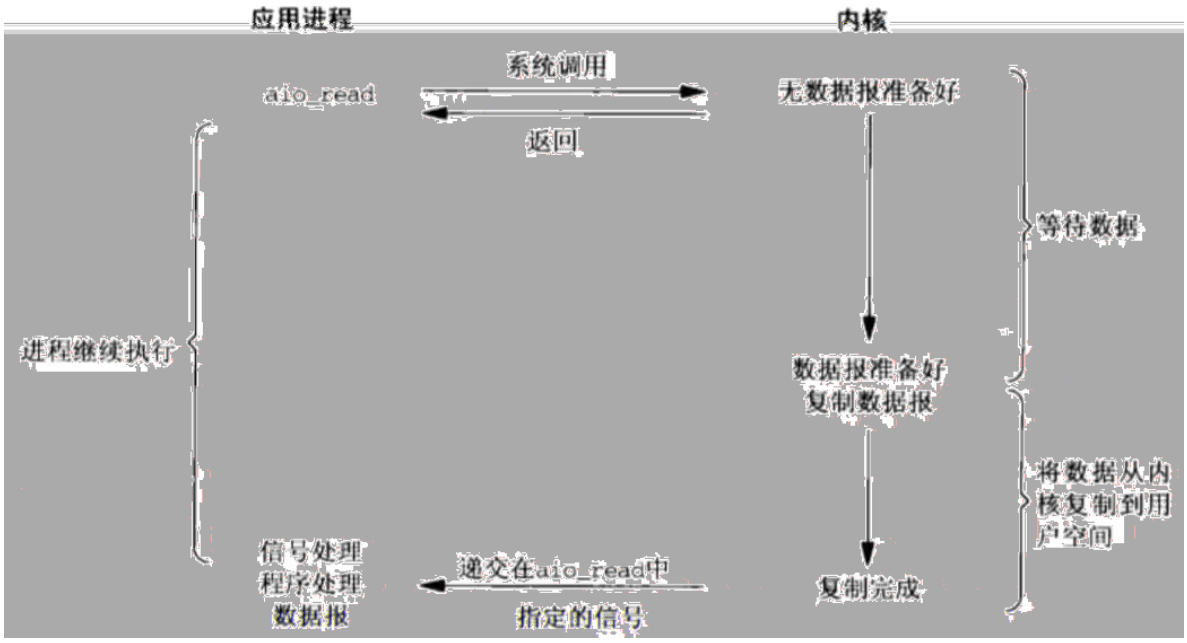


图 6.4 信号驱动 I/O 模型

异步 IO

Future-Listener



IO 2

1. IO
2. IO

IO IO IO
IO IO IO

IO IO

3.IO 多路复用

I/O I /O TCP
TCP

select

writefds readdfs exceptfds30w
select fdset
fd IO

1. select FD
2. slect () fd

3. FD 1024

poll

select fd
poll fd

epoll

epoll_ctl fd, fd
callback fd

1. fd FD 65535 1G 10W
   16G
2. FD
3. callback mmap

   select / poll
   linux
4. epoll_create() linux b+ epoll
   fd
5. epoll_ctl() epoll fd callback
6. epoll_wait() IO
   100W 1W select ,poll,epoll
   select : 1000 100W
   poll:100W
   epoll: fd,
   100W 95W poll epoll

4.Java的i/o

1. jdk1.4 BIO
   C/C++, IO AIO
2. jdk1.4 NIO, IO jdk1.7 NIO2.0 AIO
   IO

5.Netty 线程模型和 Reactor 模式

Reactor

IO

1. reactor
2.
3. reactor CPU

4.
5. reactor java selector select

Reactor 单线程模型

1. NIO TCP NIO TCP
2.

NIO

Reactor 多线程模型

Acceptor NIO
Acceptor

Reactor 主从线程模型

Acceptor NIO IO NIO 2
IO Netty BossGroup
WorkGroup

Netty 使用 NIO 而不是 AIO

linux AIO epoll NIO Netty
reactor epoll IO Netty Java NIO

6.Echo服务

7.源码剖析

EventLoop和EventLoopGroup

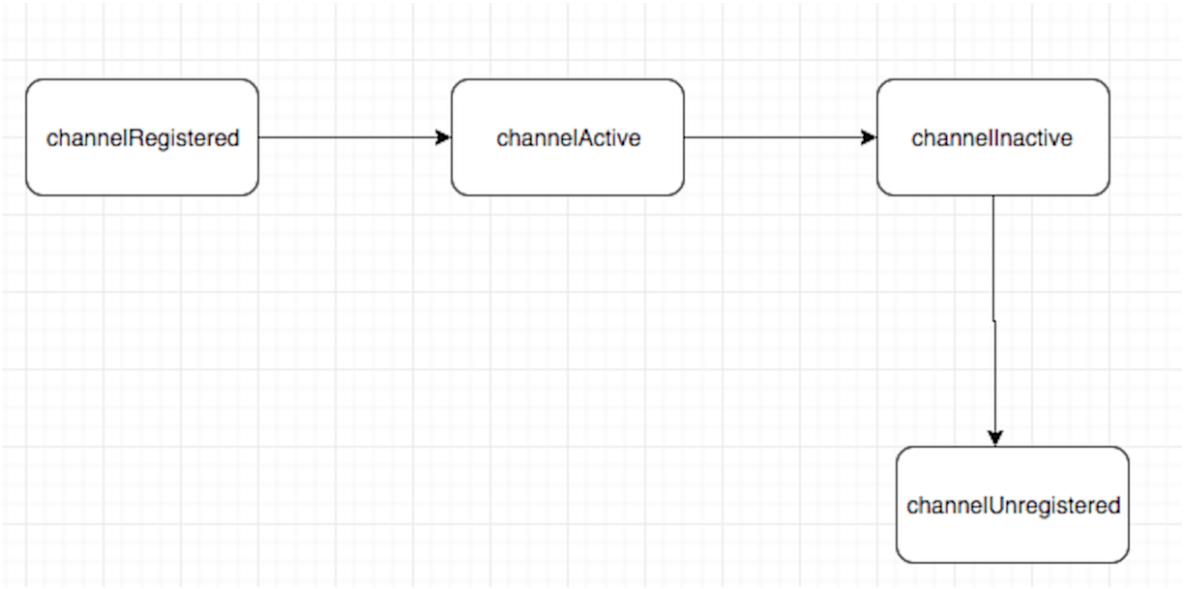
RPC 3 IO http,brotobuf/thrift EventLoop
EventLoop Channel, Channel EventLoop
EventLoop EventLoopGroup Cahnnel EventLoopGroup
EventLoop
NIO Channels BIO Channels
accept,connect,read,writeEventLoopGroup CPU * 2

Bootstrap

1. group: Reactor EventLoopGroup
   1)

- ChannelPipeline
  ChannelHandler

Channel ChannelPipeline ChannelHandler ChannelPipeline Channel

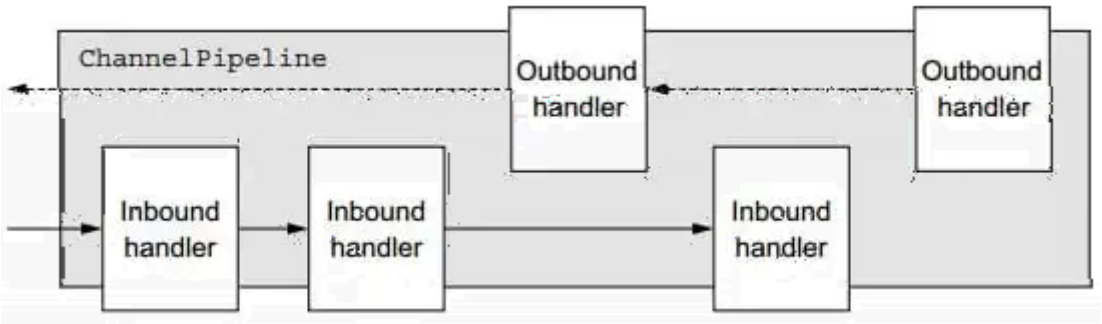


- channelRegistered
  channel EventLoop Selector
- channelUnRegistered
  channel EventLoop Selector
- channelActive

- channelInActive
  channel

ChannelHandler和ChannelPipeline

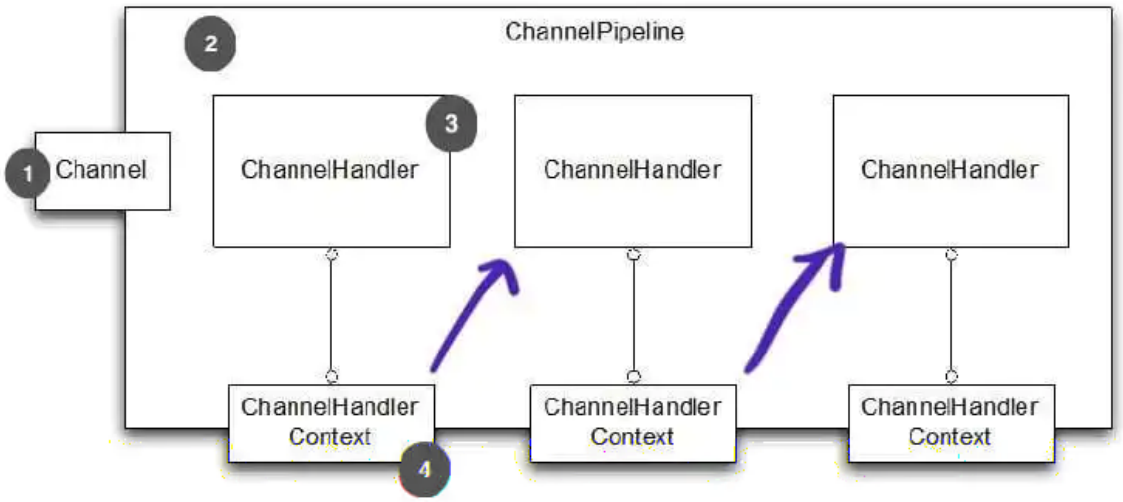
ChannelHandler handlerAdded: ChannelHandler ChannelPipeline handlerRemoved: ChannelHandler ChannelPipeline exceptionCaught: ChannelHandler 2



ChannelInboundHandler Channel ChannelInboundHandlerAdapter , SimpleChannelInboundHandlerChannelOutboundHandler Channel

ChannelPipeline: ChannelHandler ChannelHandler Channel event ChannelPileline ChannelHandler

ChannelHandlerContext



1. channelHandlerContext ChannelHandler ChannelPipeline
   ChannelHandlerContext Channel ChannelPipleline write
   Channel,ChannelPipeline,ChannelHandlerContext 2
   ChannelHandlerContext Handler
2. AbstractChannelHandlerContext
   ,next/prev
3. DefaultChannelHandlerContext
   Handler
   fire handler fire

Handler执行顺序

InboundHandler OutboundHandler

```
channel.pipeline().addLast(new OutboundHandler1());
channel.pipeline().addLast(new OutboundHandler2());
channel.pipeline().addLast(new InboundHandler1());
channel.pipeline().addLast(new InboundHandler2());复制代码
```

InboundHandler1 InboundHandler2 OutboundHandler2 OutboundHandler1InboundHandler1
fireChannelRead() InboundHandler ctx.write(msg),
OutboundHandlerctx.write(msg) Inbound outbound
outboundHandler channel.write(msg), pipline.write(msg)
outbound inbound
inbound outbound

ChannelFuture

netty I/0 I/0 ChannelFuture
I/0 I/0
I/0 I/0
Future

IO Future sync await channelhandler sync await

ChannelPromise

ChannelFuture IO

编解码

java / url ,base64 java

1.
2.
3.

PB Thrift Marshalling Kyro

Netty

- InboundHandler
- : OutBoundHandler

Netty
Encoder ,Decoder ,Codec

Netty解码器 Decoder

Decoder ChannelInboundHandler

- decode :
- decodeLast: cahnnel

- ByteToMessageDecoder

- ReplayingDecoder
  ByteToMessageDecoder
  ByteToMessageDecoder
- MessageToMessageDecoder
  POJO POJO
  TCP

- DelimiterBasedFrameDecoder
- LineBasedFrameDecoder
- FixedLengthFrameDecoder
- LengthFieldBasedFrameDecoder: message = header + body,
- StringDecoder:
  handler

Netty 编码器 Encoder

Encoder ChannelOutboundHandler

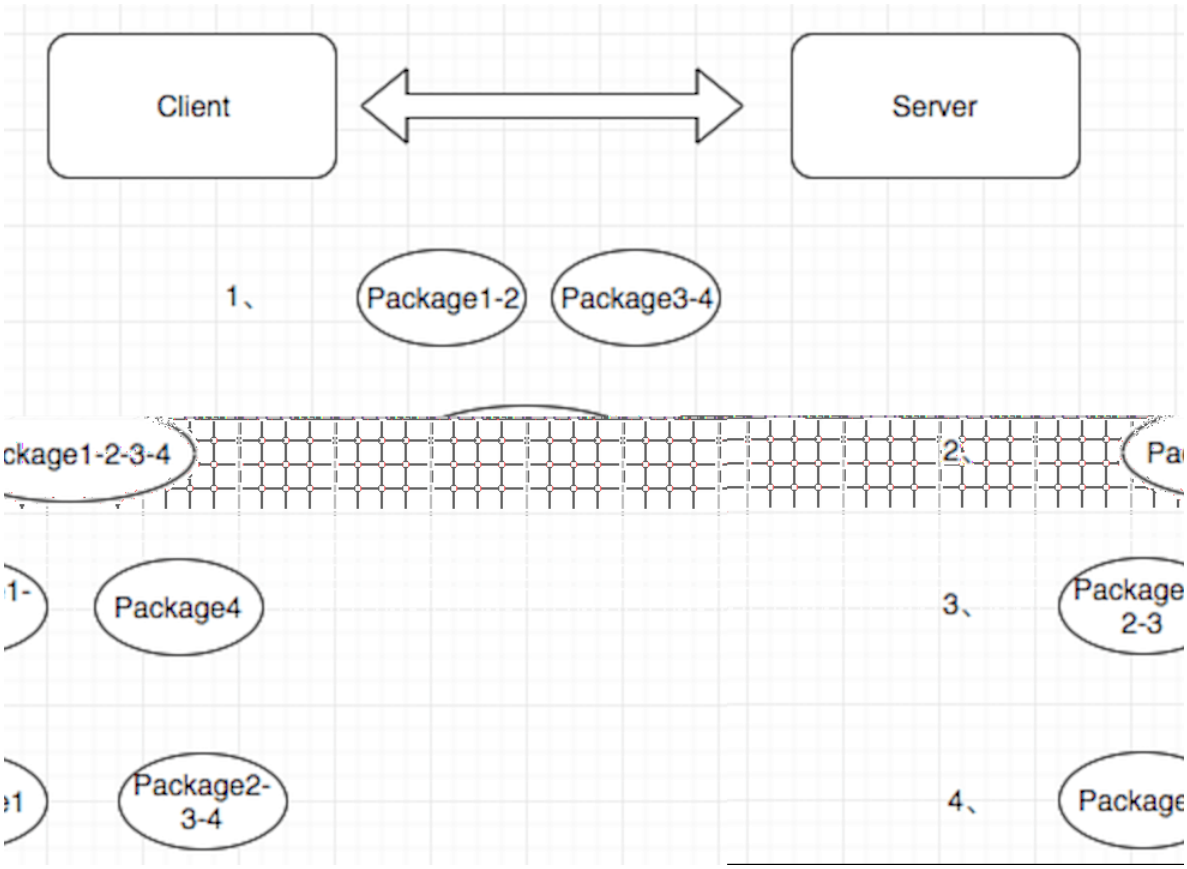
- MessageToByteEncoder
  write

- MessageToMessageEncoder

Netty 组合编解码器 Codec

- ByteToMessageCodec
- MessageToMessageCode

TCP 粘包，拆包



TCP TCP TCP
,client server
TCP Nagle TCP
UDP

TCP 半包读写解决方案

Nagle TCP

1. (10)
   abcdefgh11abcdefgh11abcdefgh11
2. (?)
   dfdsfdsfdf?dsfsdfdsf$dsfdsfsdf
3.
   header + body

Netty 自带解决 TCP 半包读写方案

- DelimiterBasedFrameDecoder:
- LineBasedFrameDecoder:
- FixedLengthFrameDecoder:
- LengthFieldBasedFrameDecoder : message = header + body

实战半包读写

LineBasedFrameDecoder: StringDecoder

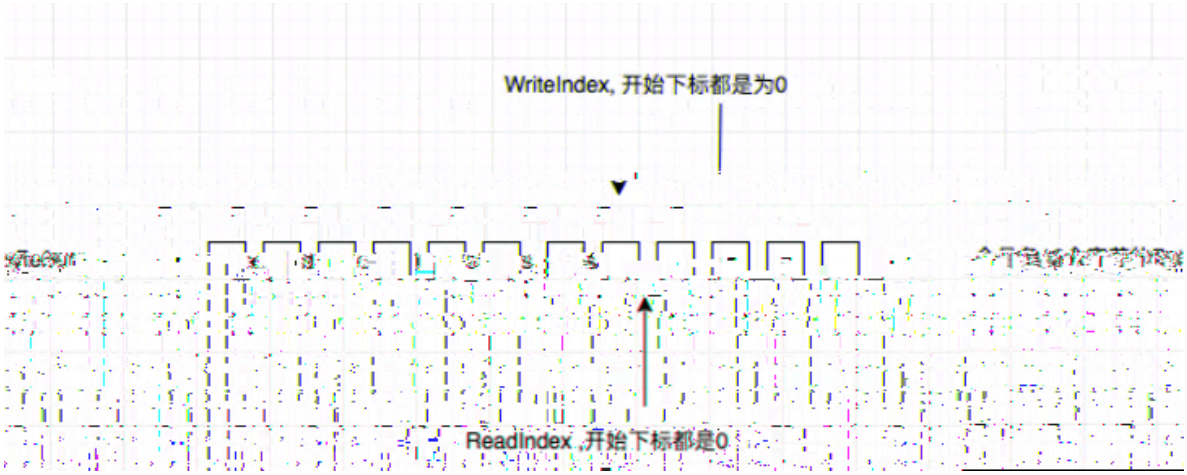
自定义分隔符解决 TCP 读写问题

DelimiterBasedFrameDecodermaxLength:
TooLongFrameExceptionfailFast: true, maxLength
TooLongFrameException, false,
TooLongFrameExceptionstripDelimiter: delimiters: ByteBuf

自定义长度半包读写器 LengthFieldBasedFrameDecoder

maxFrameLength lengthFieldOffset
lengthFieldLength
lengthAdjustment Header + Body ,
Header Netty initialBytesToStrip

ByteBuf



- JDK ByteBuffer
  Flip()
- Netty ByteBuf

ByteBuf 创建方法与常见的模式

ByteBuf: ByteBuf

1. ByteBufAllocator
   Netty 4.x PooledByteBufAllocator
   UnPooledByteBufAllocator
2. Unpooled: ByteBuf

ByteBuf

1.
   heap buffer jvm

2.
   Direct buffer
   JVM

3.
   ByteBuf,
   IO

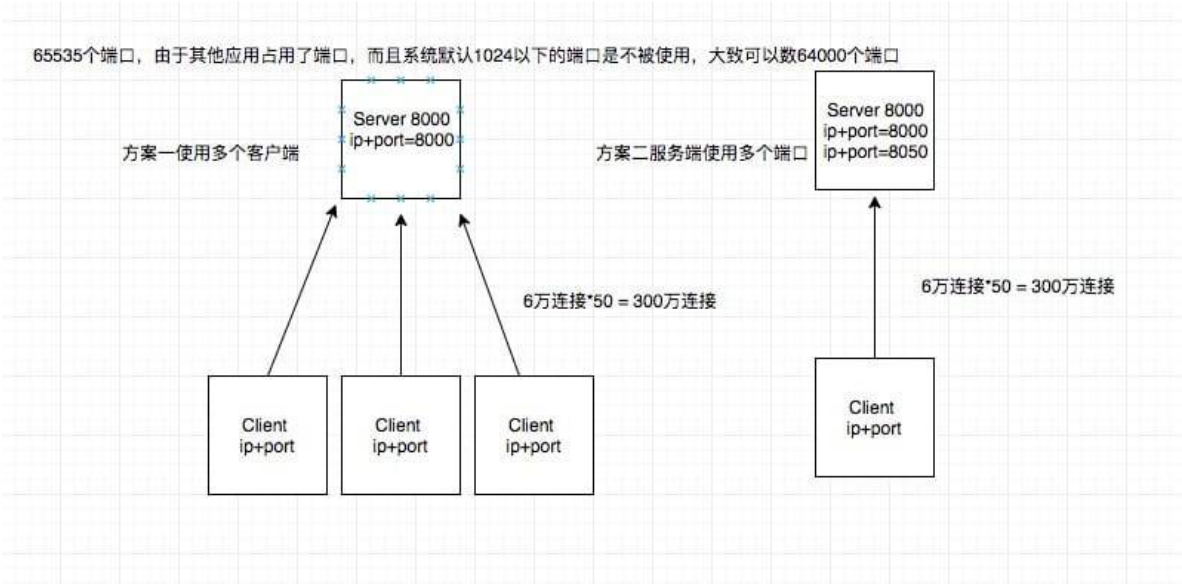
Netty 设计模式

Builder ServerBootstrap pipeline channel
HandlerAdapter

Netty 单机百万实战

1. IO
2. Linux

3. TCP
   TCP IP IP
   1024~65535




```
65545
优化：
```

4. sudo vim /etc/security/limits.conf fd ulimit -n FD

```
root soft nofile 1000000
root hard nofile 1000000
* soft nofile 1000000
* hard nofile 1000000复制代码
```

1. sudo vim /etc/sysctl.conf fd

```
fs.file-max=1000000复制代码
```

sysctl -p cat /proc/sys/fs/file-max fd

1. reboot

-Xms5g -Xmx5g -XX:NewSize=3g -XX:MaxNewSize=3g

数据链路

- -

DNS - DNS - IP- - - -