P135 (port )

Mach OS 會建立task，可視為mutiple thread與一些資源的process

MachOS使用message passing ,並用port當作mailbox來收發

(port有限且單向，所以雙向溝通時會送到對方port，收response則送到reply port)

Port可有多sender，但只能有一個receiver(比喻:家中信箱)

MachOS也用port表示thread、memory等等資源

\*Port right

使用port的權限，比如收message，此權限由port owns(創port的task)持有，而Port right也可以分享，比如T1的Port P1送給T2的P2一訊息，要求T2回應，則需要P2的MACH\_PORT\_RIGHT\_SEND，也因此thread能夠快速溝通與分享部分資源

\*Task self port& notify port

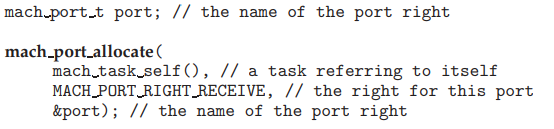
當task被創建時也會順便創建這兩個port

Kernel會擁有task self port的Recive權利，換言之Task可透過task self port傳信息給kernel

Notify port 則是kernel傳信息給task (當然task就有接收權)

\*port allocate

可以使用mach\_port\_allocate 來做到 (創建port，並要一塊空間做message queue)



注意: port本身用正整數代表 、 且要使用port要有port right，port right也是一個旗標

P136 Bootstrap server

Task會可以使用bootstrap port

可以讓task 註冊造好的port於bootstrap server，讓其他task查看並取的send該port的權利

Message可以使用指標來傳送較大的檔案(Out-of-line data)

P138 ALPC

Message大概有四種方式可以處理queue滿情況

1. 等待queue有空間
2. 等待一段固定時間
3. 不等待，直接return失敗
4. 請OS記住，當Queue有空位就通知sender重傳

Windows有提供多個Os環境 (subsystem)

可以把應用程式視為client，那些環境視為server

**APLC(advanced local procedure call)(不是API，使用者看不見!)，用於在同台機器上溝通，類似RPC，但最佳化於window**

**Windows類似mach使用OO方式看待port，有兩種port，connect與communication**

**1.所有server會公開connect port供程式送訊息**

**2.接下來會建立一對私人channel(兩邊的communication port)**

**3.接下來有三種方式傳訊息**

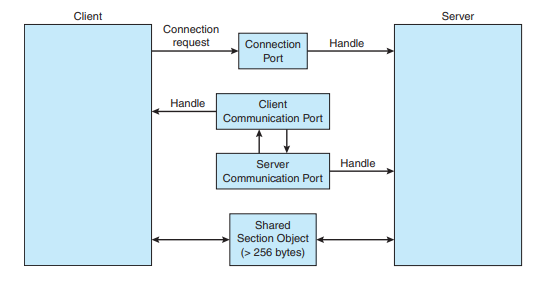
**1.小訊息(256 byte以下): 使用兩邊的queue**

**2.大訊息: 會開一塊專門share的區域傳(share section object)**

**(傳此位址與size)**

**3.極大訊息: 有API可以直接讓server讀取client端記憶體**

\*會由雙方決定要不要開channel，訊息大小決定要不要開section object



P151 android RPC

RPC用於client-server分散式系統，也可以支持同一系統的IPC

使用binder framework來開發RPC

Android 提供 application component，應用程式可以使用複數個application component來完成任務(功能相當於service)