一、overview

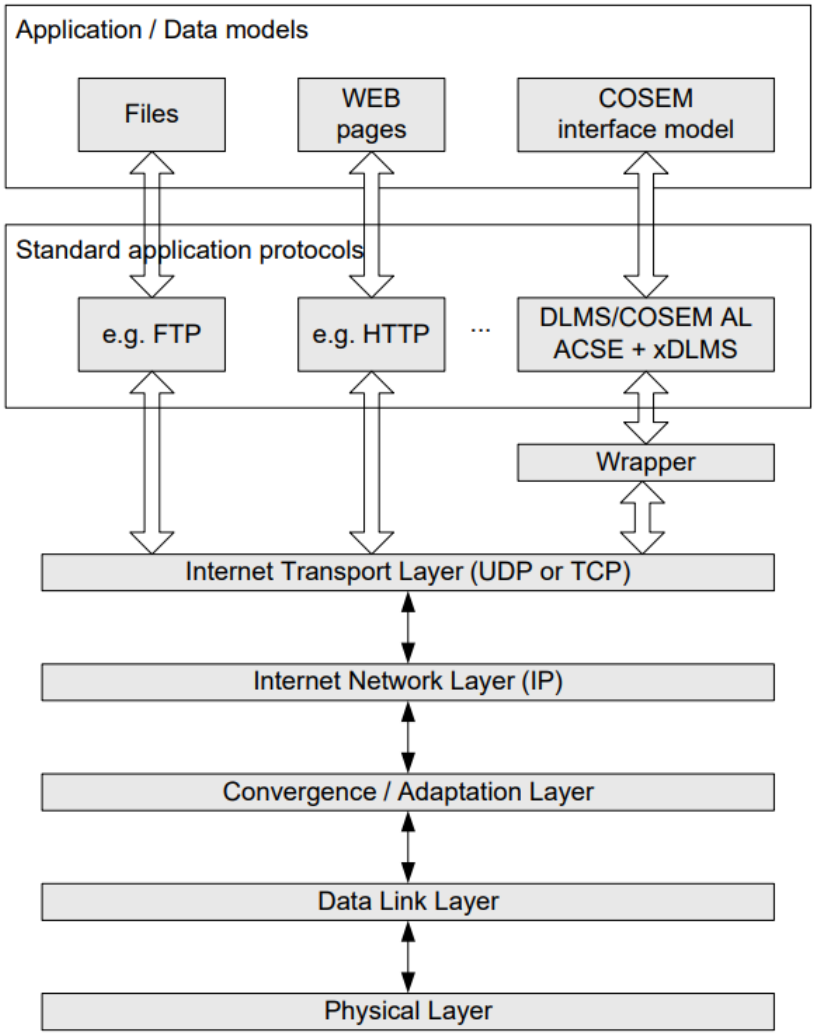


Figure 26 – DLMS/COSEM as a standard Internet application protocol

二、TCP、UDP区别

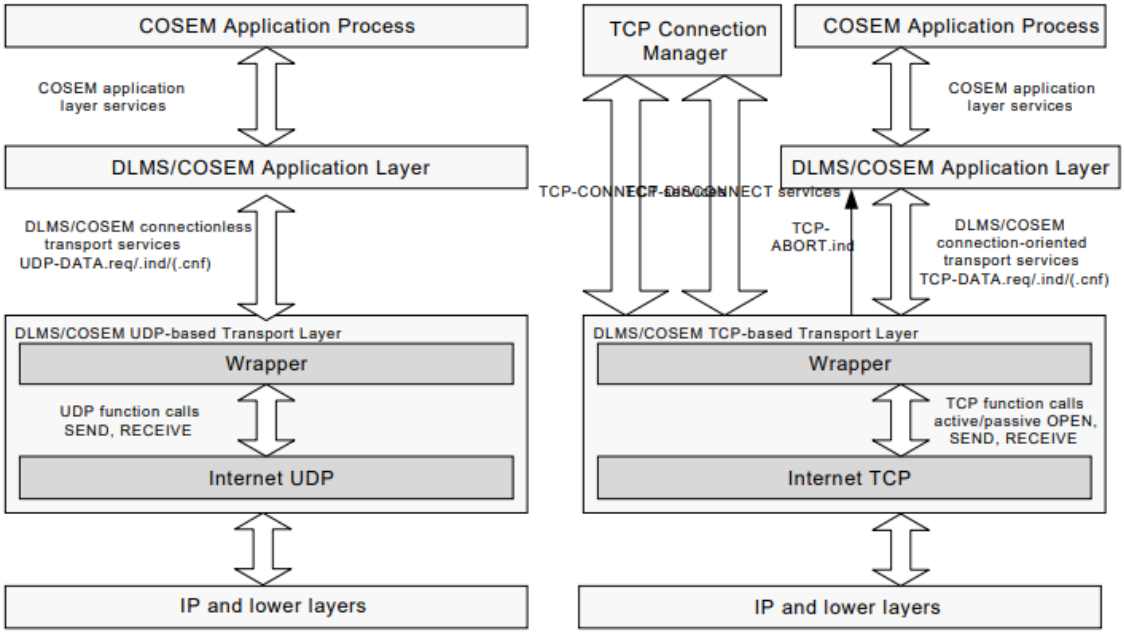
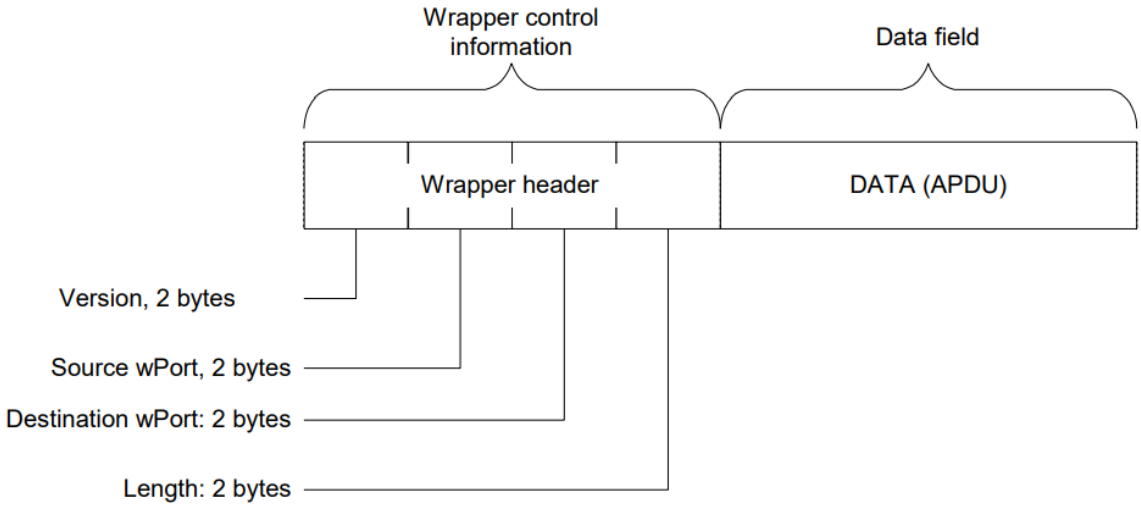


Figure 27 – Transport layers of the DLMS/COSEM_on_IP profile

区别:

	TCP	**UDP**
连接方面	面向连接	无连接，即发送数据之前不需要建立连接
安全方面	提供可靠的服务，通过TCP连接传送的数据，无差错，不丢失，不重复，且按序到达	尽最大努力交付，即不保证可靠交付
传输效率	传输效率相对较低	传输效率高，适用于对高速传输和实时性有较高的通信或广播通信
连接对象数量	连接只能是点到点，一对一的	支持一对一，一对多和多对多的交互通信

三、WPDU



`NOTE The maximum length of the APDU should be eight bytes less than the maximum length of the UDP datagram.`

- **Version**: carries the version of the wrapper. Its value is controlled by the DLMS UA. The current value is 0x0001. Note, that in later versions the wrapper header may have a different structure;
- **Source wPort**: carries the wPort number identifying the sending DLMS/COSEM AE;
- **Destination wPort**: carries the wPort number identifying the receiving DLMS/COSEM AE;
- **Data length**: indicates the length of the DATA field of the WPDU (the xDLMS APDU transported).

	Destination wPort	Source wPort
p2P通信	固定为1	客户端id
DCU透传连表	sapId（大于1）	客户端id
DCU连接	固定为1	客户端id

四、UDP-PDU

- **报文结构图**

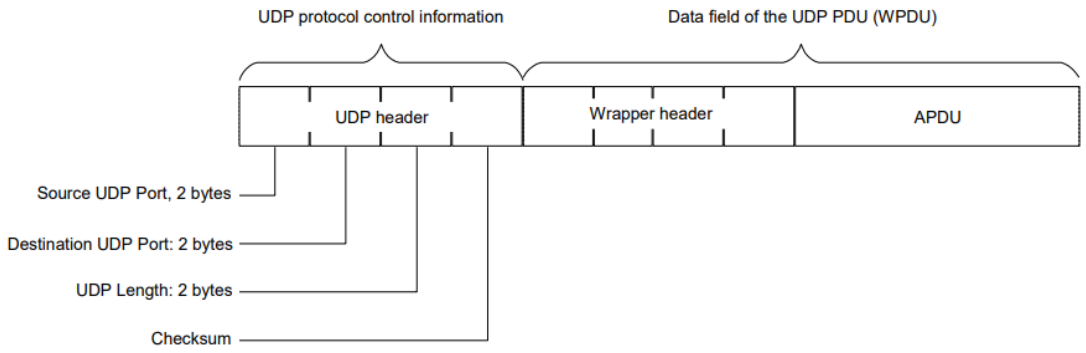


Figure 30 – The DLMS/COSEM connection-less, UDP-based transport layer PDU (UDP-PDU)

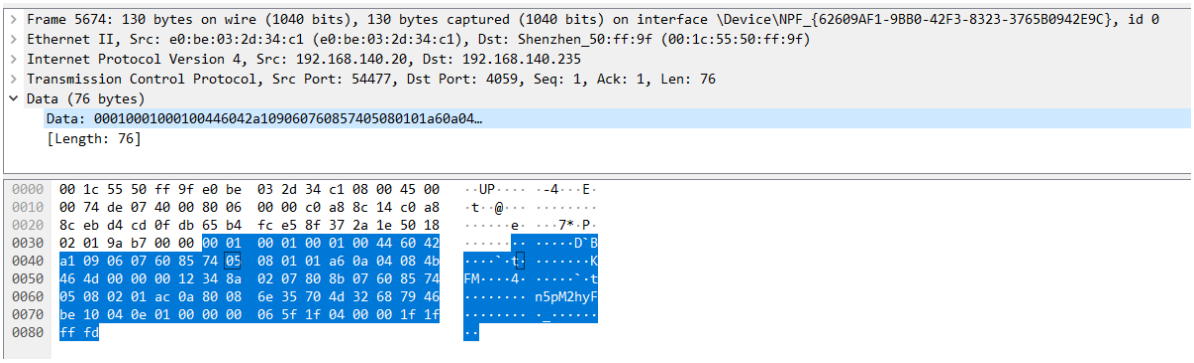
注：UDP项目使用较少，暂无实际报文，后续遇到再补。

五、TCP-PDU

- **报文结构图**

0	4	8	10	16	24	31
Source TCP Port				Destination TCP Port		
Sequence Number						
Acknowledgement Number						
Data Offset	Reserved		Flags	Window		
Checksum				Urgent Pointer		
TCP Options					Padding	
Data (part of the WPDU)						
...Data (part of the WPDU)...						

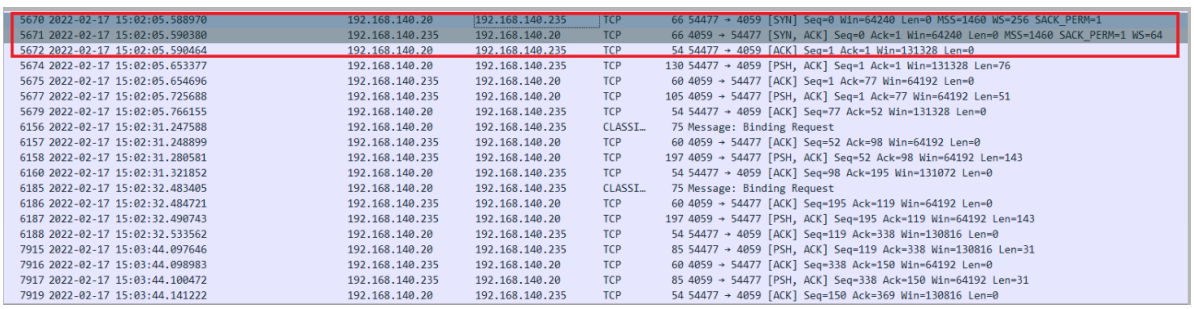
Figure 32 – The TCP packet format



- **TCP连接**

三次握手：

第一次握手：
客户端将TCP报文标志位SYN置为1，随机产生一个序号值seq=J，保存在TCP首部的序列号(Sequence Number)字段里，指明客户端打算连接的服务器的端口，并将该数据包发送给服务器端，发送完毕后，客户端进入SYN_SENT状态，等待服务器端确认。
第二次握手：
服务器端收到数据包后由标志位SYN=1知道客户端请求建立连接，服务器端将TCP报文标志位SYN和ACK都置为1，ack=J+1，随机产生一个序号值seq=K，并将该数据包发送给客户端以确认连接请求，服务器端进入SYN_RCVD状态。
第三次握手：
客户端收到确认后，检查ack是否为J+1，ACK是否为1，如果正确则将标志位ACK置为1，ack=K+1，并将该数据包发送给服务器端，服务器端检查ack是否为K+1，ACK是否为1，如果正确则连接建立成功，客户端和服务器端进入ESTABLISHED状态，完成三次握手，随后客户端与服务器端之间可以开始传输数据了。



四次挥手：

第一次挥手： Client端发起挥手请求，向Server端发送标志位是FIN报文段，设置序列号seq，此时，Client端进入FIN_WAIT_1状态，这表示Client端没有数据要发送给Server端了。
第二次挥手：Server端收到了Client端发送的FIN报文段，向Client端返回一个标志位是ACK的报文段，ack设为seq加1，Client端进入FIN_WAIT_2状态，Server端告诉Client端，我确认并同意你的关闭请求。
第三次挥手： Server端向Client端发送标志位是FIN的报文段，请求关闭连接，同时Client端进入LAST_ACK状态。
第四次挥手： Client端收到Server端发送的FIN报文段，向Server端发送标志位是ACK的报文段，然后Client端进入TIME_WAIT状态。Server端收到Client端的ACK报文段以后，就关闭连接。此时，Client端等待2MSL的时间后依然没有收到回复，则证明Server端已正常关闭，那好，Client端也可以关闭连接了。

> **因为开启了延时ack机制，导致收到第一个fin之后，发送ack的条件不能满足立即发送ack的条件，导致ack的发送被延时了，在延时的过程中，应用如果确认没数据要发，并且也要关闭此连接的情况下，会触发发送fin，这个fin就会和之前的ack合并被发出**