**北京邮电大学软件学院**

**2019-2020学年第一学期实验报告**

**课程名称： 通信协议软件设计**

**项目名称： 实验一：基于SDL的通信协议描述**

**项目完成人：**

**姓名： 宋振铭 学号： 2017211861**

**姓名： 陈凌云 学号： 2017211868**

**姓名： 吴志镛 学号： 2017211869**

**姓名： 赵景煜 学号： 2017211880**

**姓名： 陆琦伟 学号： 2017211881**

**姓名： 薛子豪 学号： 2017522041**

**指导教师：**\_**\_\_\_\_\_\_\_\_ \_ \_雷友珣\_ \_\_\_\_ \_\_\_\_\_\_**

**日 期： 2019年 10月 28日**

1. **实验目的**

通过本实验使学生理解通信协议形式化描述的意义、掌握SDL描述语言及其开发工具。

1. **实验任务**

对选定的通信协议进行分析，基于SDL语言开发环境，使用SDL语言对所选定的通信协议进行描述和分析。

1. **实验内容** 
   1. 下载、安装SDL语言开发环境PragmaDev Studio。
   2. 选定一通信协议（如滑动窗口协议）进行分析。
   3. 使用PragmaDev Studio对选定的协议进行SDL描述，包括该协议的SDL系统图、功能块图、进程图。若所选定的协议提供单工通信服务功能，需要分别对协议发送方协议实体和接收方协议实体均给出SDL系统描述。
   4. 使用PragmaDev Studio对所描述的协议进行模拟分析。
2. **实验环境** 
   1. Windows 10系统主机；
   2. PragmaDev Studio软件
3. **实验分工**

**我们小组在开始之前首先进行集体讨论，首先我们明确了我们要实现两周滑动窗口协议（回退N协议和停等协议），我们先商讨了如何实现设计和实现协议，最后确定先画出协议的MSC设计图，然后再通过SDL用语言实现，详细分工如下：**

吴志镛:理想化停等协议的分析和实现，并设计了两钟滑动窗口协议的MSC图

陈凌云：基于理想化停等协议，进行选择重传部分的分析和实现。:

赵景煜: 负责停等协议中的模拟发送和接收数据，发现错误并对程序进行修改

宋振铭：承担回退N协议的进程模块中接受来自发送方的数据和有关ACK的处理

陆琪伟: 负责回退N协议计的时器信号的处理以及来自网络层数据包的处理

薛子豪：负责回退N协议的数据结构，信号的定义以及系统整体架构的组织

1. **协议描述**

**1、停等协议**

停等协议和ARQ协议 有一些相似性，都是停-等式的方法，但是二者最大的差别在于 停等协议是全双工协议，可收可发；而ARQ协议是单工协议只有一方进行发送另一方负责接收。

分析：

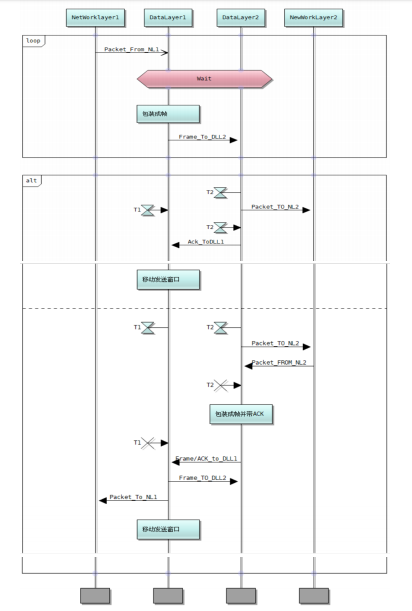
1.一般状态下两个链路层其中一个开始发送第一帧

2.接受方则要检查是否位重复帧，而对于一位滑动窗口而言，是否重复就是判断 ack seq 是否分别和next\_frame\_to\_send与 frame\_expected 都不同

3.而如果不是重复帧 则会发送其将要发送的下一帧数据，否则持续发送同一帧数据（即无论如何只要接收到一帧必定返回一帧）

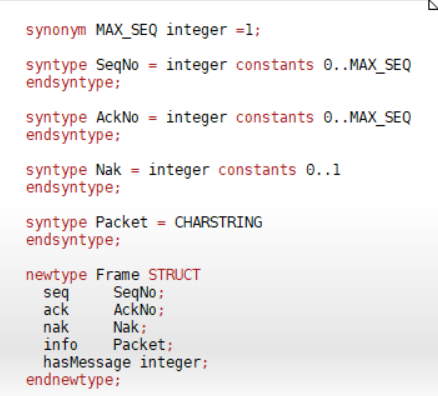
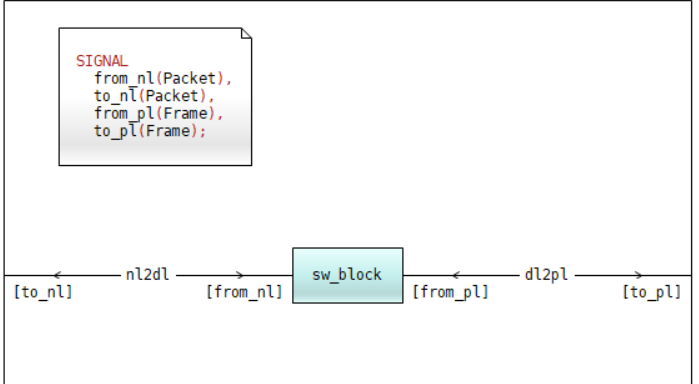
**2**、**GoBack N协议**

**详细的协议的设计MSC图如下：**

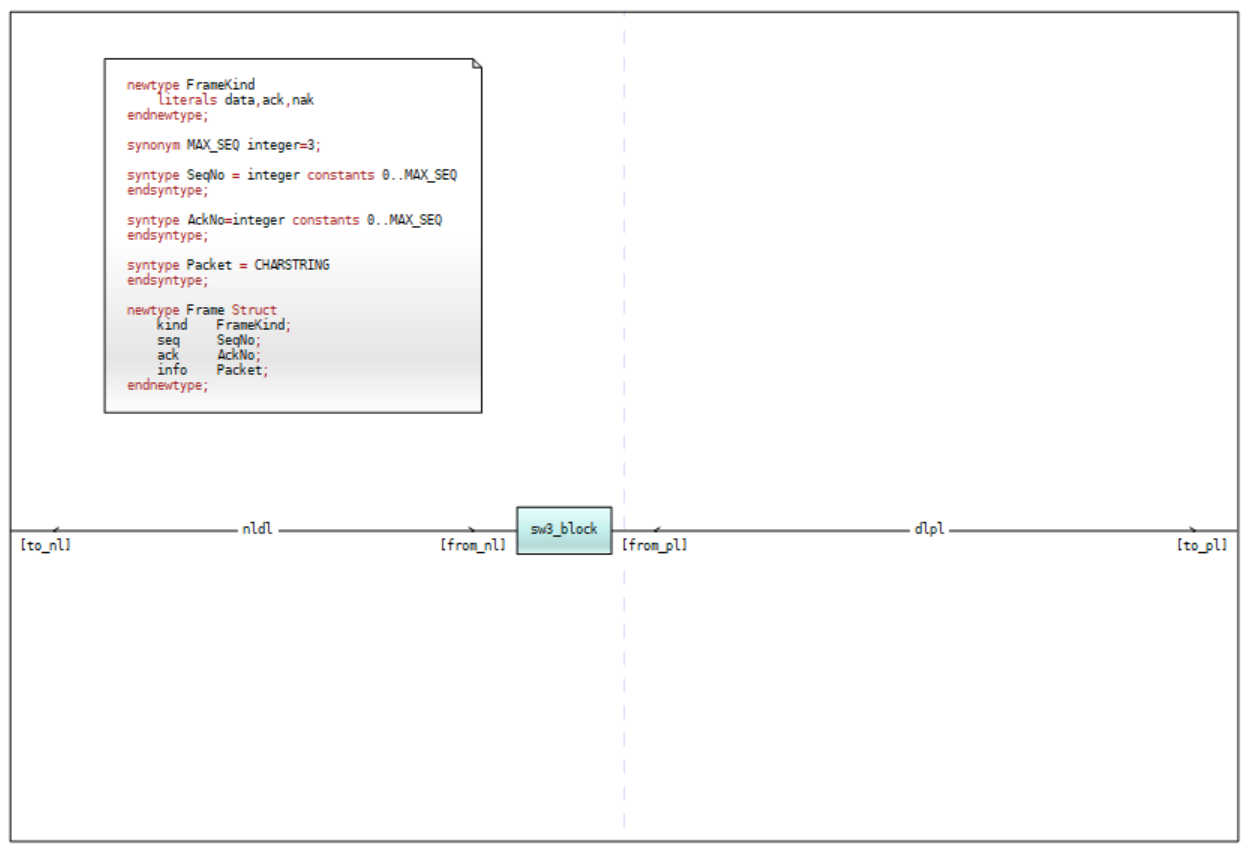


1. **SDL系统描述**
   1. **停等协议：**

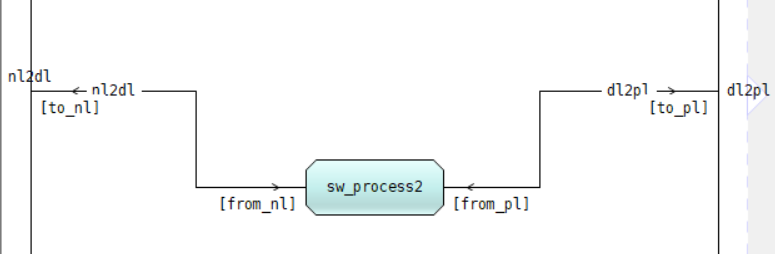
系统图：



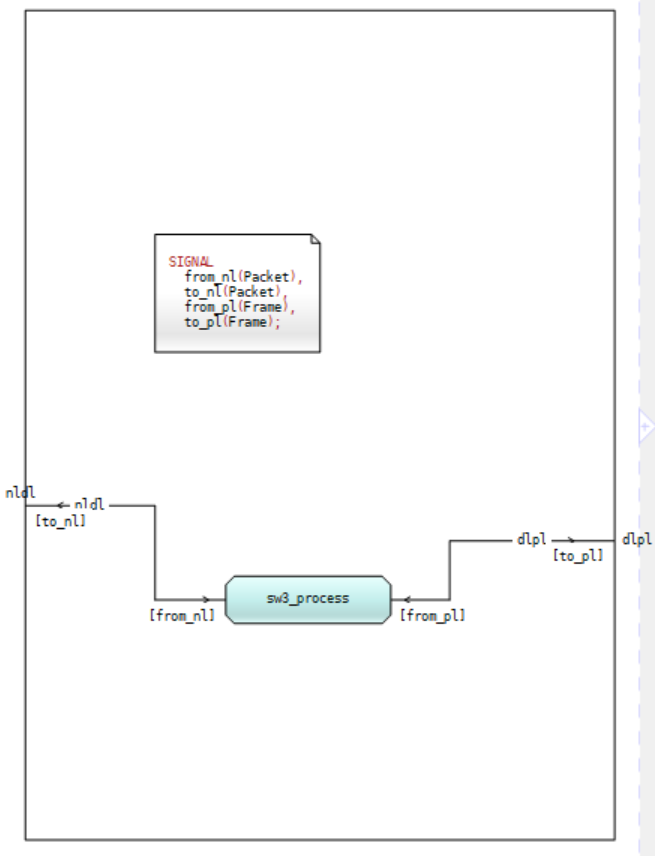
1. 整个系统图 即代表了数据链路层使用的数据传输协议
2. Signal 即为 来自物理层/网络层，通向物理层/网络层的四个信号，代表传输数据的流向
3. 只有两个信道 nl2dl(网络层到数据连路层) 和 dl2pl(数据链路层到物理层)。
4. Sw\_block 即为滑动窗口的协议功能块
5. 数据定义中，
   1. MAX\_SEQ 代表 滑动窗口的大小 由于是 one-bit 所以为 1
   2. SeqNo 为帧序列 ，AckNo 为确认序号,Nak为重传序号
   3. Packet即代表 网络层向数据链路层传递的包，使用的类型为 字符串
   4. Frame代表数据链路层向物理层传递的 帧
      1. 其中 hasMessage 代表此帧是否包含信息，主要用于应对网络层的数据尚未传递给链路层 而 接受方的确认数据已经达到要求下一帧的情况，用于确保只有帧到达了才进行传递
   5. **回退N协议**



1. **功能块图**
   1. **停等协议：**



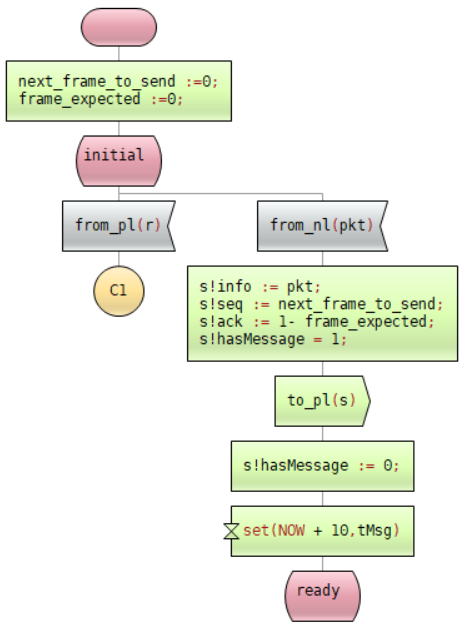
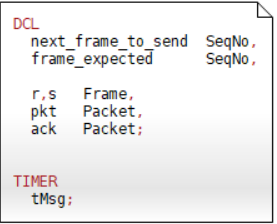
* 1. **回退N协议：**



1. **进程状态图**

**a)停等协议:**

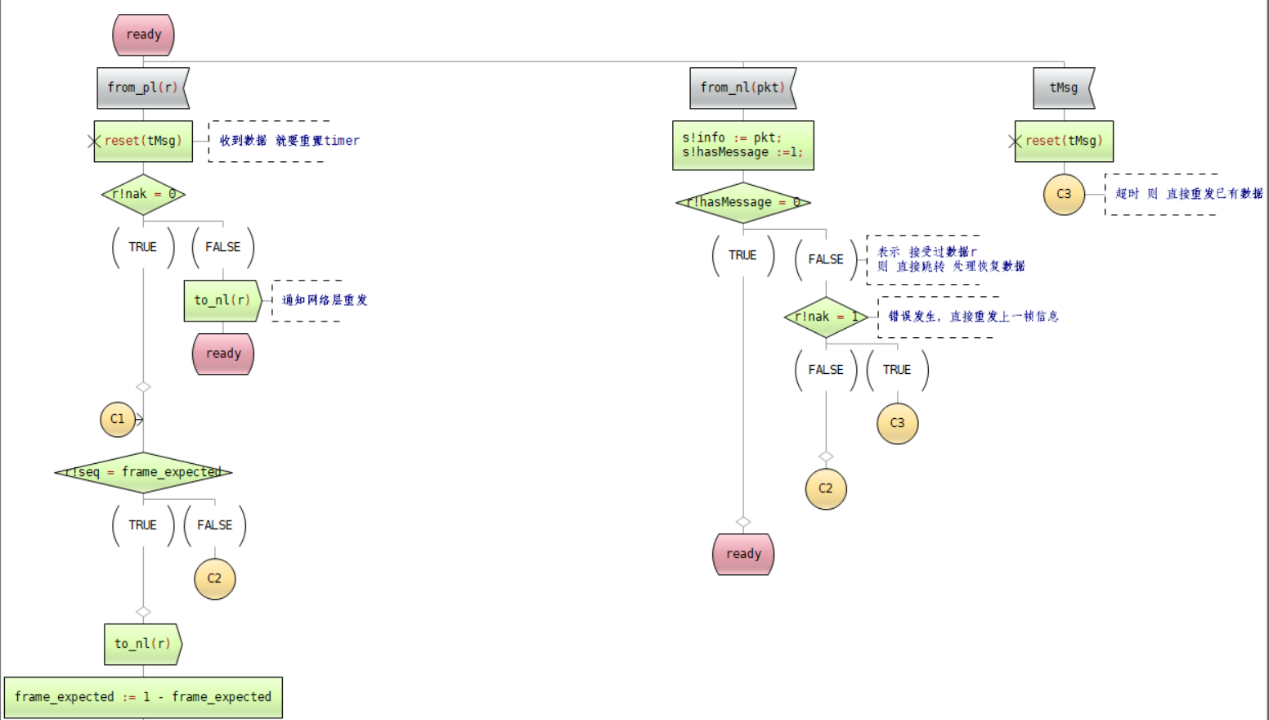
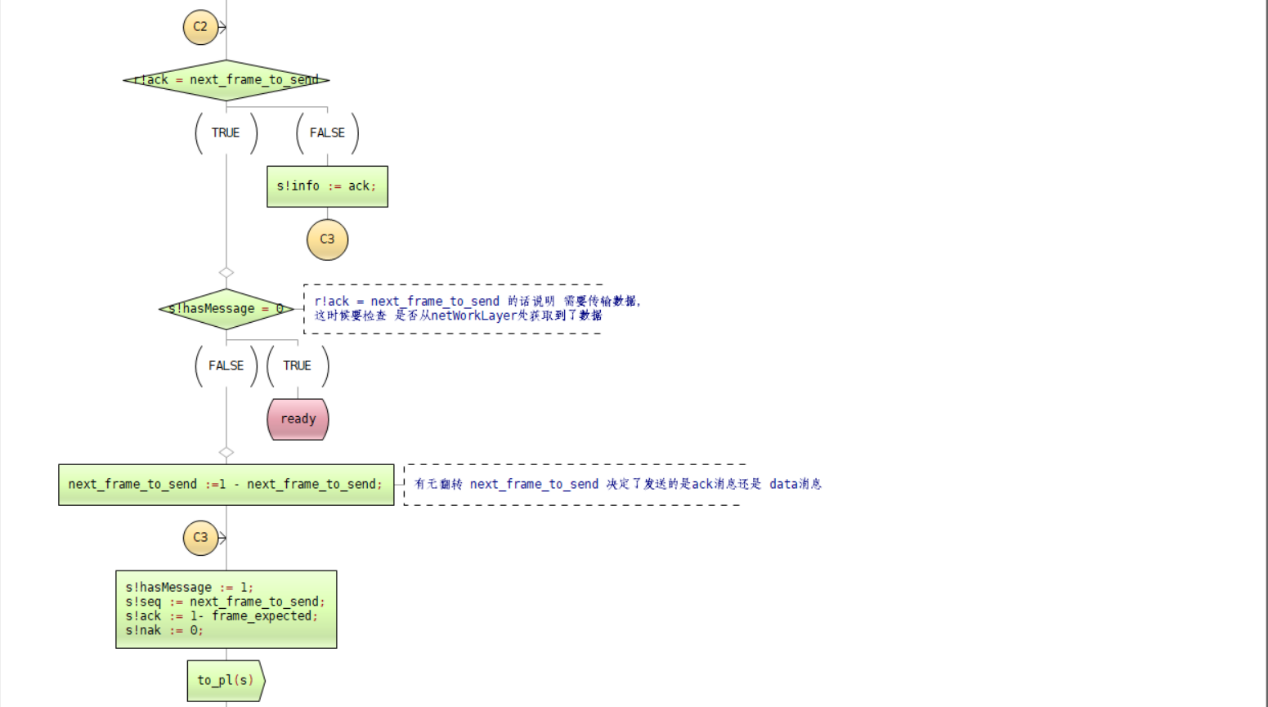
1. **声明**
   1. Next\_frame\_to\_send 为下一次要送的数据的帧号
   2. Frame\_expected 为 想要获取的帧号
   3. R为接受的帧 s 为发送的帧
   4. pkt 为从网络层接受到的包，ack 只是空字符串，用于代表数据为空
   5. tMsg为计时器



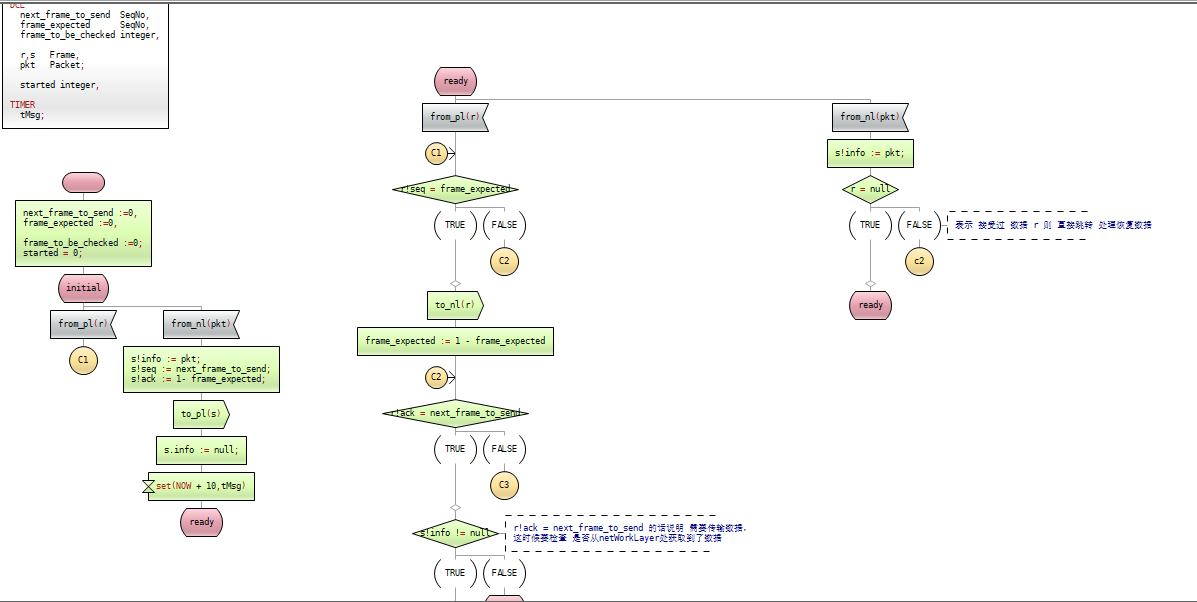
1. **程序**
   1. 输入 分为三种 —— from\_pl from\_nl tMsg

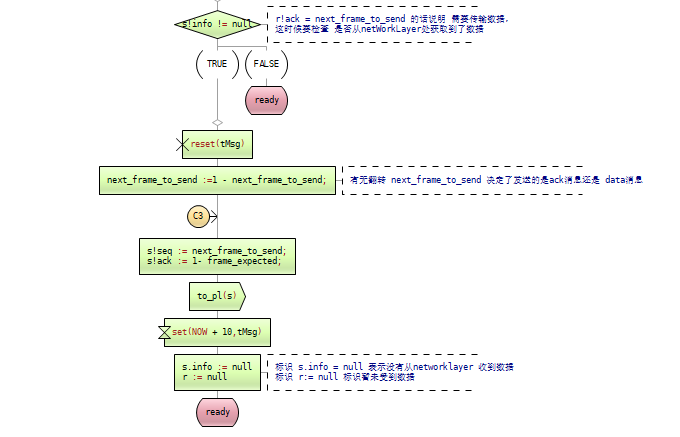
分别对应 物理层数据到来 网络层数到达 和 计时器超时

* 1. 由于 调试的时候 只有 一个程序在执行，所以实际上 数据的 传收 是同一个进程进行的。  
     至于如何判断谁是接收方，谁是发送方，主要是根据 帧的ack 是否等于要发送的帧号，如果不是则 代表为确认消息 否则就是 发送的消息



* + - 1. **回退N协议：**

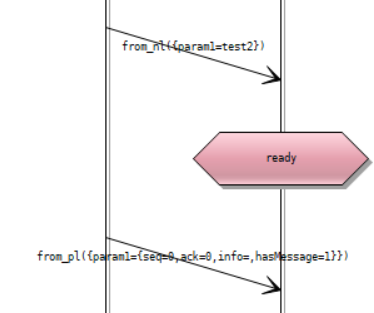
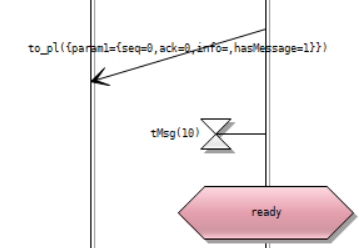
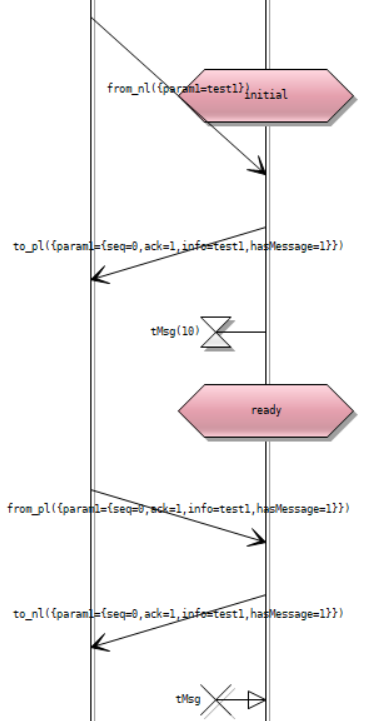
****

****

1. **主要场景的模拟执行过程**
   1. **停等协议：**

## 正常收发

**如左为一个正常的数据收发过程：**



from\_nl 从网络层收到信息 test1 ,表示要发送信息给接收方，于是链路层开始工作

链路层 将PDU 合并到网络层得到的package，构成一个frame并向下传递给物理层，

【】

设置计时器，当超时的时候重发消息。

进入准备状态 准备接受新信息

接受方的物理层收到信息将信息解析为帧传递给链路层

链路层得到帧，再将帧解析为包向上传递给网络层

取消自己的计时器【指接受方发送的确认消息的计时器】

同时 接受方要将确认信息发送给物理层表示收到了信息，要求获取下一帧数据

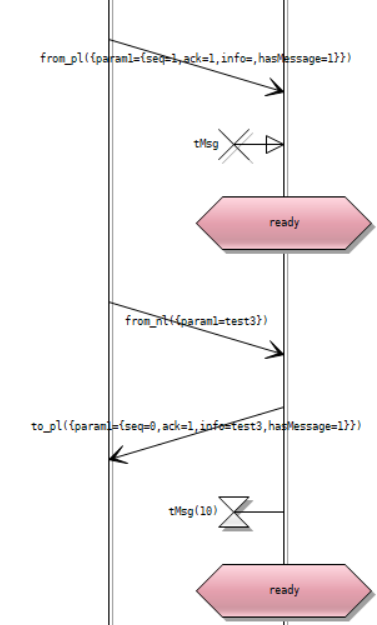
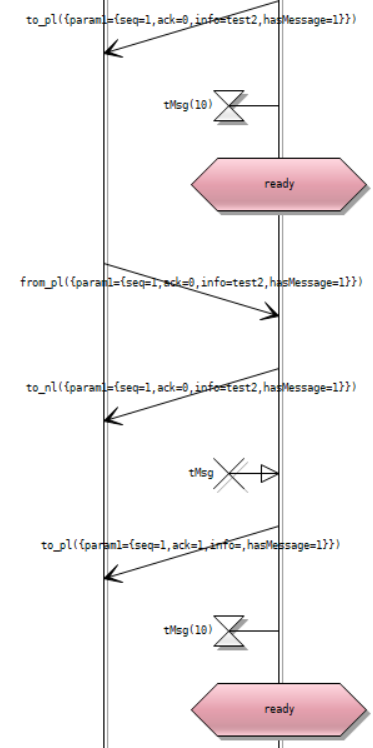
设置确认消息的计时器

此时发送方的网络层发来了下一个要发送的packet

接受方的确认消息到达，发送方将进行下一次的消息发送

## 异常一：网络层下一packet发送延迟，而接受方的确认信息已经到达

发送方 发送下一帧



设置计时器

接收方物理层将数据传递给链路层处理

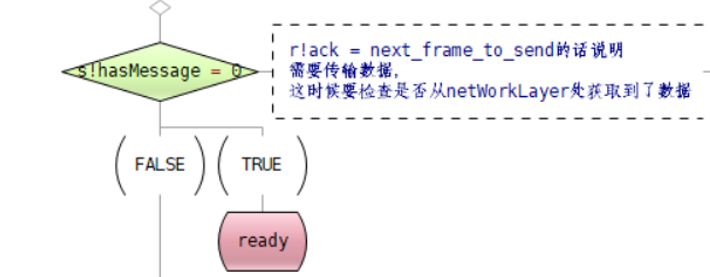
接收方链路层再将受到的数据传递给网络层

重置计时器

接受方 将确认消息 借由物理层传递给发送方

设置计时器

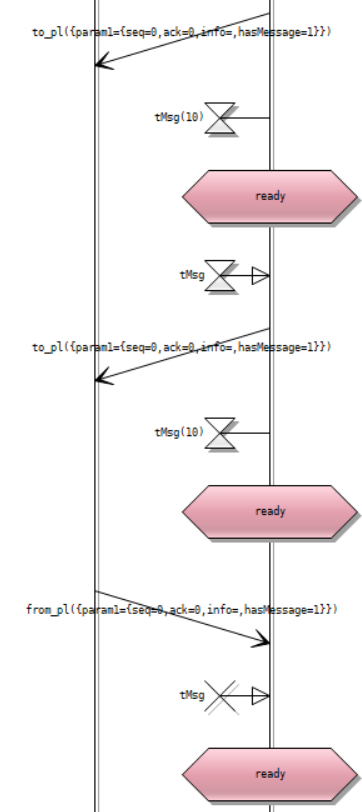
发送方接收到确认信息发现要发送下一帧，但是此时 发送方发现网络层的packet尚未传递下来(即s!hasMessage = 0 )所以进入ready  
【但是由于接受到了确认信息，所以需要重置计时器】:



在经过一定的延迟之后 网络层的数据到达，发送方继续数据的发送

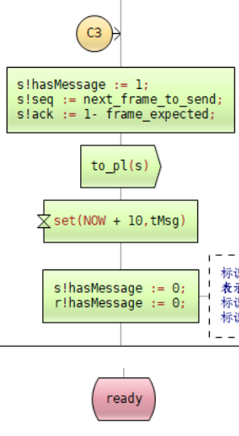
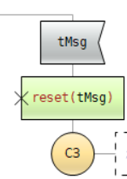
## 异常二：计时器超时，

### 接受方发送的确认信息超时：

接受方发送确认消息（info=空）

设置计时器

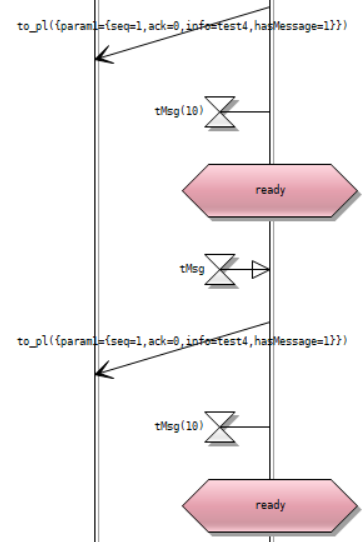
计时器超时重复发送相同的确认信息

  
再次设置计时器

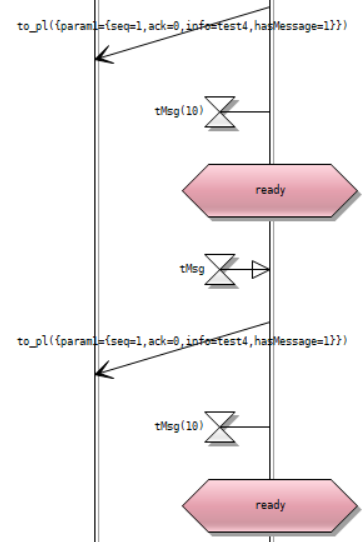
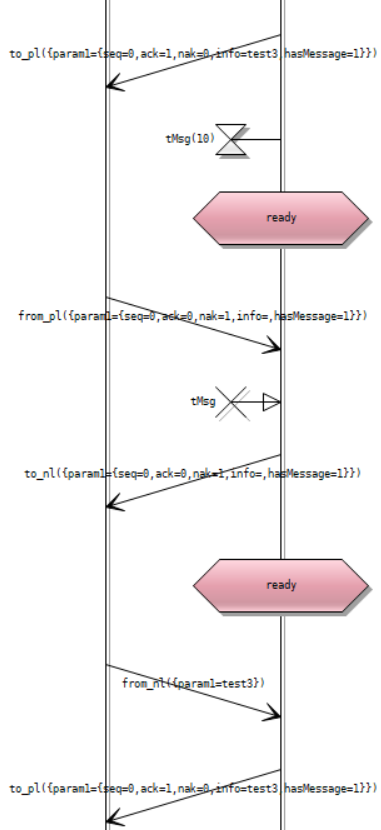
发送方物理层成功得到数据，解析成帧发送给链路层

链路层重置自身的计时器

### 发送方发送的信息超时：

同上，直接进行消息的重复发送操作

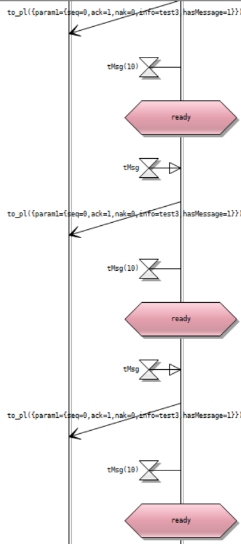
### 异常三：信息发送中产生错误（之后补充的功能故frame结中多加了一个 nak位 ）

发送方发送信息  
  
  
设置计时器

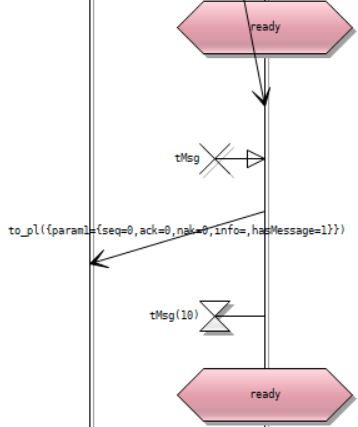
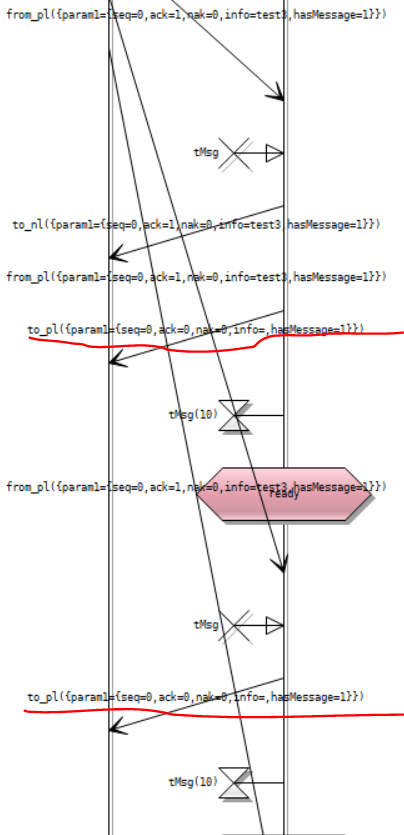
**数据经过校验程序后 发现错误，置nak = 1 清空其它信息**  
  
重置timer  
  
通知网络层重发上一个包  
  
  
  
  
网络层发送包

回到正常数据发送状态

## 异常四：发送方由于定时器定时过短，导致不断向接受方发送同一信息：

**对发送方：**

如左图，由于timer过短，导致重复发送了三条同样的数据 test3



**对接受方：**接收数据

返回确认信息

接收数据

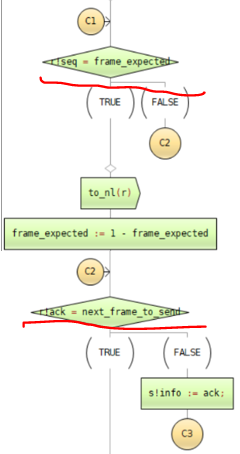
返回确认信息

接收数据

返回确认信息

如作图红线部分所示，三次接收返回的是同一个确认信息帧，代表多次相同信息的发送不会导致数据的错乱。

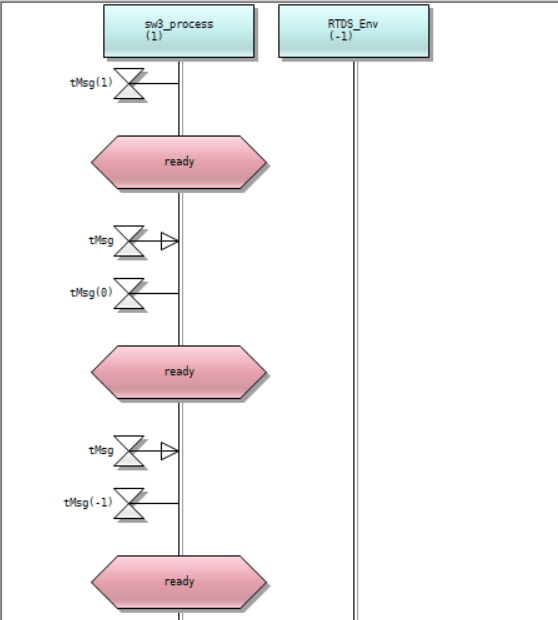
**这归功于算法中的：(见下图)**



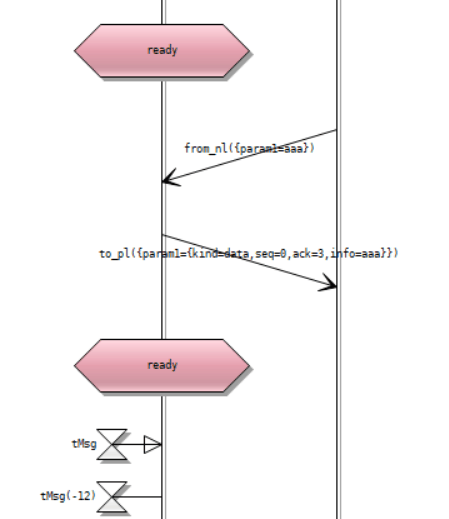
**对两个状态位的检查。**

**b)回退N协议:**

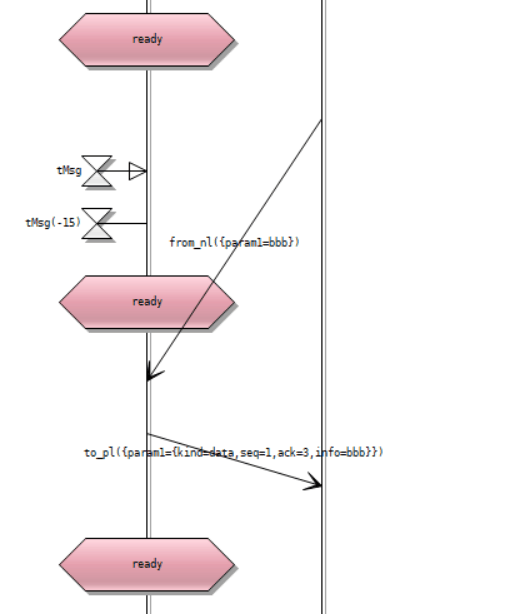
1. **运行系统，启动实时定时器，系统的定时器每隔一秒钟会收到一个时钟信号。**



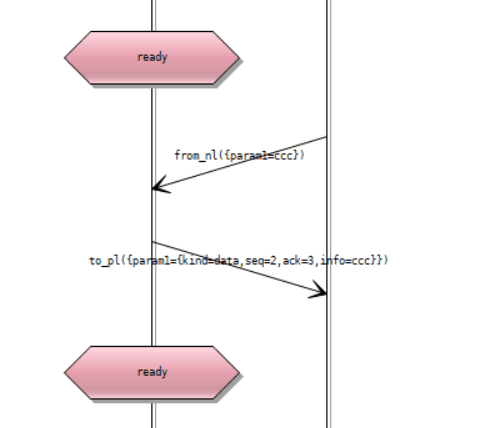
1. **系统接收到一个来自物理层的内容为aaa的数据包，将其向上传递给网络层。**



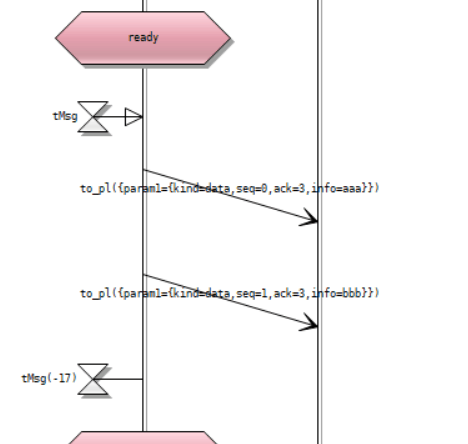
1. **系统接收到来自网络层的含有内容bbb的数据包，并将其向下送入物理层。**



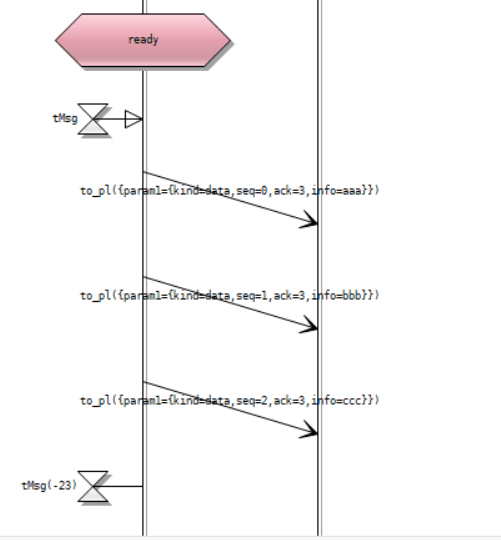
1. **系统接收到来自网络层的含有内容ccc的数据包，并将其向下送入物理层**。



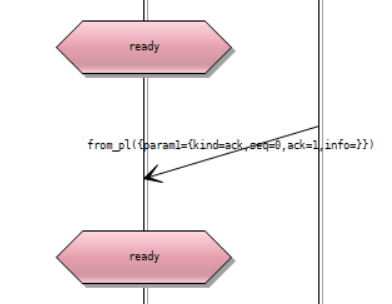
1. **因为之前所发送的两个包没有收到确认消息所以重发。含有ccc的包因未超时所以暂时没有重发。**



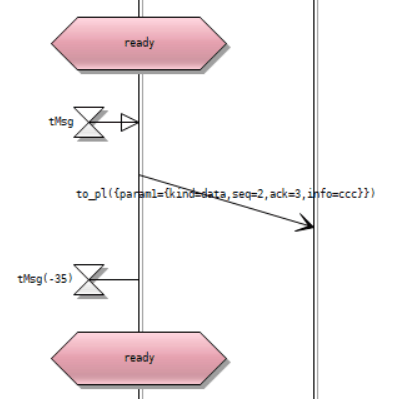
1. **因为之前发送的三个包都没有收到确认消息，所以重发这三个包。**



1. **系统收到了序列号为1的数据包的确认，即所发送的前两个数据包已经被接收方收到。**



1. **系统重发从未收到确认的数据包重发**。



**十一、 实验总结**

在本次实验中，我们学习到了如何使用PragmaDev软件进行协议的MSC图和SDL图的绘制，并进行通信协议的设计，以及对已建立的协议模型进行模拟运行和调试。通过自己分析设计并最终实现停等协议和回退N协议，我们对数据链路层中的通信有了更深刻的理解，能够将理论课学到的的知识转化为实践中的能力。