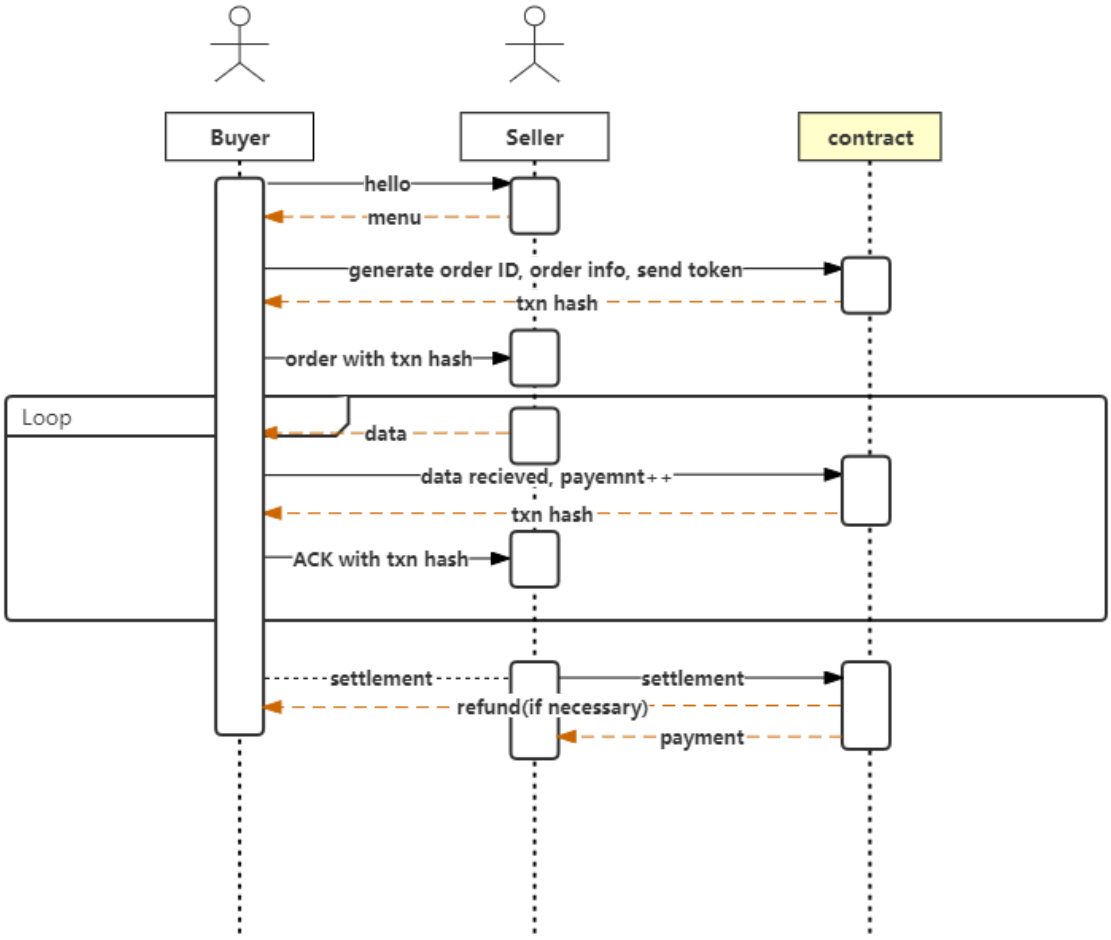
时序图UML



该图使用processon绘制，https://www.processon.com/view/link/637241637d9c0806b802689b

交互步骤

1. 买家通过registry获取卖家信息后，通过信息中的IP:port与卖家建立连接
2. 买家发送hello消息
3. 卖家收到hello，向买家发送menu
4. 买家选定商品后，生成订单ID，记录订单生成时间，deadline，数据规模，双方公钥后，向合约质押充足的代币
5. 买家将链上hash作为凭证发给卖方，卖方验证后按照粒度向买方发送数据
6. 买方收到数据，向合约更新支付信息，获取hash作为凭证发给卖方
7. 卖方验证买方是否付款后继续发送数据，重复步骤5-7
8. 买家或者卖家均有权力终止交易，当其中一方调用合约结算时，合约根据已发送数据/数据总量安比例支付给卖方/退款给买方。
9. 到达deadline后，未完成的订单可以被检测到，执行结算需要发送transaction。

合约

合约用于记录整个平台的订单状态。合约主要由一个map构成，key为订单的全局唯一ID（该ID由买家和卖家的公钥结合当时unix时间hash生成，如果碰撞，则重新生成，直到全局唯一为止），value是一个结构体，记录订单信息：

```
{
    "buyer",                                    // 买家公钥
    "seller",                                   // 卖家公钥
    "total_data",                               // 数据总量
    "data_transfered",                          // 已经传输的数据量
    "is_finished",                              // 订单是否结束
    “started_from",                             // 订单生成时间
    "deadline"                                  // 截止时间
}
```

应用层信息接口

通用消息模板

消息模板包括四个部分：消息类型（hello，menu，order，data，ACK，exit），数据载荷（根据不同的消息类型有所不同），签名信息（0x开头的十六进制65字节hash，以太坊签名格式，保证信息的真实性），买家的支付凭证，具体支付的transaction hash。

```
{
    "message_type": "",                 // 消息类型
    "payload": "",                      // 数据payload
    "signature": "",                    // 签名
    "verification": ""                  // 凭证
}
```

hello

```
{
    "message_type": "hello",
    "payload": {
        "public_key_of_the_buyer"               // 买家公钥(用于签名验证)
    },
    "signature": "",
    "verification": ""
}
```

menu

```
{
    "message_type": "menu",
    "payload": {
        "list_of_available_data": [{                    // 所有数据
            "data_ID": "",                              // 商品ID(唯一)
            "data_size": ""                             // 数据大小
            ”data_info": ""                             // 数据描述信息
        }],
        "granularity": "",                              // 传输粒度
        "list_of_available_payment": [                  // 支付方式
```

```
            "alipay",                                   // 支付宝
            "token"                                     // 虚拟货币
        ],
    },
    "signature": "signature of the seller",
    "verification": ""
}
```

order

```
{
    "message_type": "order",
    "payload": {
        ”order_hash": "",                       // 订单哈希，由买家生成并上链
        "data_ID": "",                          // 商品ID
        "payment": "",                          // 支付方式
    },
    "signature": "signature of the buyer",
    "verification": "transaction hash where the buyer has posted the order
details" //买家生成链上订单并的凭证
}
```

买家如果选择token的支付方式，还需要额外向合约里打钱，支付宝则不需要。买家将此transaction hash作为凭证发给卖家

data

```
{
    "message_type": "data",
    "payload": "data",
    "signature": "signature of the seller",
    "verification": ""
}
```

卖家按照粒度，传输数据并签名。

ACK

```
{
    "message_type": "ACK",
    "payload": "ack",
    "signature": "signature of the buyer",
    "verification": "transaction hash where the payment has been made"

}
```

买家收到数据后，将支付信息上链，更新合约状态，将transaction hash作为凭证发给卖方。

exit

数据传输结束后，双方向对方发送exit消息，附带打分和评价，用于rating上链（TODO）。

```
{
    "message_type": "exit",
    "payload": "rating each other",
    "signature": "",
    "verification": ""
}
```

异常处理

如果传输过程中某一方掉线，买家可以重新向卖方发送上一次的ACK信息，卖方验证后，继续传输数据，实现断点续传。

如果其中一方永久失联，任意一方均有权力调用结算，合约根据已发送数据/数据总量按比例支付给卖方/退款给买方。

如果双方均失联，在下一次某一方与合约交互时，合约可以根据公钥检测到未完成并超时的订单，此时可发送transaction进行结算。