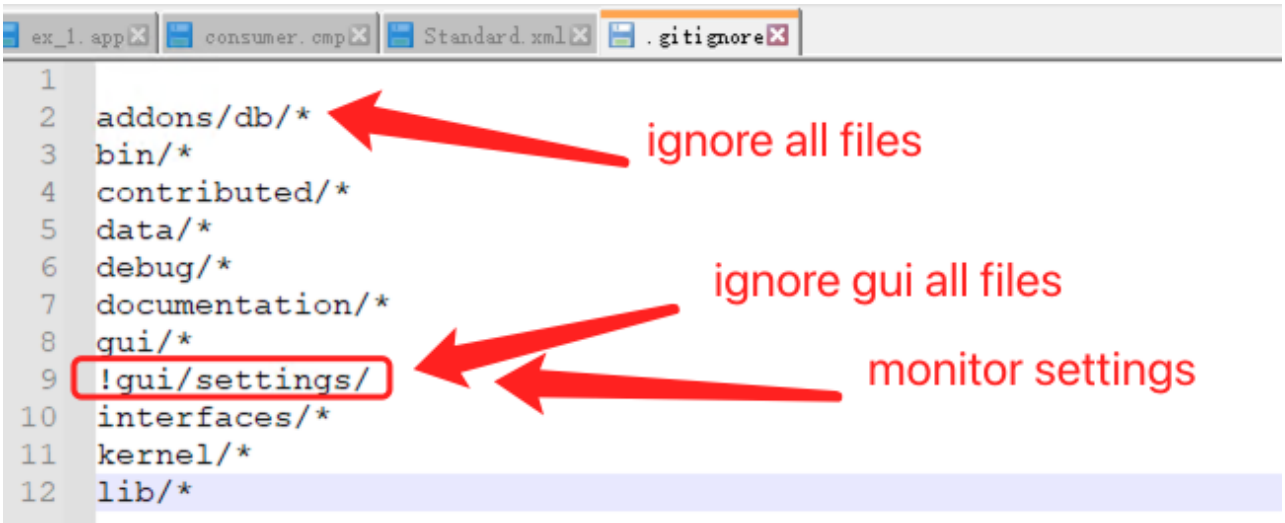
EXata 扩展（五）：添加单主机应用 Consumer

目标：添加一个Single Host Applicaiton：Consumer，实现初步的类似 NDN 中的 Consumer 的功能

0. 创建 Git 仓库：HyperNetExata51

参考：https://blog.csdn.net/qq_34803821/article/details/86648313

- 下载 Git 和 Git Desktop，安装。
- 资源管理器，右键“～/5.1”目录，“Git Bash here”，进入 Git 命令行下该文件夹；
- 执行“Git Init”，会提示“Initialized empty Git repository in D:/Scalable/exata/5.1/.git/”；
- 添加 include/ 文件夹：add include
- 提交：gti commit –m "Add include"
- 使用 Git Desktop：打开本地仓库，发布到 GitHub 上，输入远端仓库名称：HyperNetExata51.
- 创建完毕 OK
- Github Desktop 添加libraries/user_models/文件夹。
  - 左侧“Changed Files ”勾选 user_models/ 所有文件
  - 输入 summary，如“Add user-models firstly”--》Commit to master
  - 然后：Push to origin，更新到 GitHub 远端仓库：https://github.com/jiangtaoluo/HyperNetExata51.git
- 在 修改 .gitignore 文件减少 Github Desktop 显示的“Changes”



-

1. Consumer 的功能需求

1.1 Consumer 协议概述

1. **协议层次定位**：作为一种单主机应用协议：single host Application

2. 基本功能：
    a. 创建一个请求消息：Request message，可支持多个方法，先支持“GET”，以后支持Post、Put
    b. 填写 Name，以及其他需求（未来）
    c. 交下层（暂时 UDP，未来补充 HNP）发走，等待应答（XMIT_REQ_WAIT_RESPONSE状态）；
    d. 收到 Response，则打印结果，更新统计；
    e. 消息采用 HTTP 一样的文本编码方式。
3. 第二阶段实现：
    a. 多个请求消息一起发；
    b. 按一定统计分布连续发。
4. 对等服务端为 Producer 下一节实现。
5. 区分 Producer 和 Provider：
    a. Producer 作为生产者，Provider 为供应者，包括生产者、中间节点或者由 Producer 指定的其他节点。

1.2 Consumer 协议实现

1.2.1 创建文件

1. 创建 app_consumer.h 和 app_consumer.cpp，保存在 user_models/src 下。
2. 创建 hnp_common.h，存放共同的声明。
3. 在application.h【include目录下】中AppType 枚举类型中添加应用协议类型：APP_CONSUMER 和 APP_PRODUCER. 设定默认端口 6698【本人手机尾号！】

```
    // LuoJT: add CONSUMER for HNP
    APP_CONSUMER,
    APP_PRODUCER = 6698,

    APP_PLACEHOLDER
};
```

4. 在trace.h 文件：协议追踪列表中添加新协议：

```
    // Luojt: CONSUMER and PRODUCER
    TRACE_CONSUMER,
    TRACE_PRODUCER,

    // Must be last one!!!
    TRACE_ANY_PROTOCOL
};
```

5. 在applicaition.cpp 中添加新协议初始化方法
    a. 包含新协议头文件：#include "app_consumer.h"
    b. 在 APP_InitializeApplications 中添加新协议处理部分。“CONSUMER”对应场景应用配置文件（XXX.app）中的协议名称-CONSUMER。下面要确立Consumer应用初始化的参数，然后再进行初始化。

```
            }
        // LuoJT: Consumer
            else
            if (strcmp(appStr, "CONSUMER") == 0)
            {
                // to be continued
            }
#ifdef INTERFACE_JNE_VMF
            else
            if (strcmp(appStr, "VMF") == 0) {
                VMF::Wrapper::runTests();
            }
```

c. CONSUMER 命令行参数设计【待续】见 1.2.2 节，确定后继续；

d. 补充读参数部分的代码

1.2.2 Consumer 的命令行参数【后面已修改】

为首次简化，参考 CBR 的参数，但改造为 Single host Application 的参数，设计命令行参数

- source：即运行节点 ID
- item to send：
- item-size：in bytes
- interval：
- start-time
- end-time：

即 CONSUMER <source-Id> <items_to_send> <item-size> <interval> <start_time> <end_time>

如：CONSUMER 1 100 1024 0.1S 3S 20S

表示：CONSUMER 应用，节点=1，待发送 100 个，每个大小为 1024 bytes，时间间隔固定为 0.1 second，开始时间：3 second，结束时间为 20 second。

1.2.3 读入 Consumer 参数

接 1.2.1 节中，在APP_InitializeApplicatioins中补充 CONSUMER 协议补充参数读入部分，首先编译，确认参数读入准确。

注意：在VS2010 CMD 窗口进行 nmake 编译！【普通 CMD 窗口编译出现错误！！！】

在ex_1.app中添加一行：CONSUMER 1 100 1024 0.1S 3S 20S

在 VS2010 配置调试属性，命令行参数改为ex_1.config，进行调试。参数读入正确。

1.2.4 GUI 添加 Consumer 协议

1. 创建一个新的 CMP 文件：在～/gui/settins/components/ 下，复制 myprotocol.cmp 生成 consumer.cmp；
2. 编辑consumer.cmp，替换所有 MYPROTOCOL 为 CONSUMER，删除Priority MDNP 等暂时不用的属性。特别地，propertytype 设定为“CONSUMER-SINGLEHOST”，这与下面设定要一致。

3. 在 GUI 工具箱的 Single Host Applicaitons 中添加一个新按钮 <SC> (代表 Service Consumer)

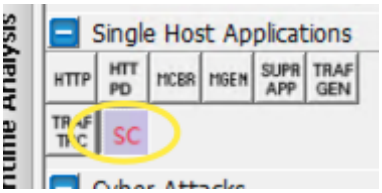
   a. 修改 standard.xml（位于~/gui/settings/toolsets/），在“Single Host Applications” Category 下的末尾，添加一行

```
<subcategory name="CONSUMER" icon="sc.png" tooltip="HNP service consumer" categorytype="Single Host Applications" type="App" propertytype="CONSUMER-SINGLEHOST" />
```

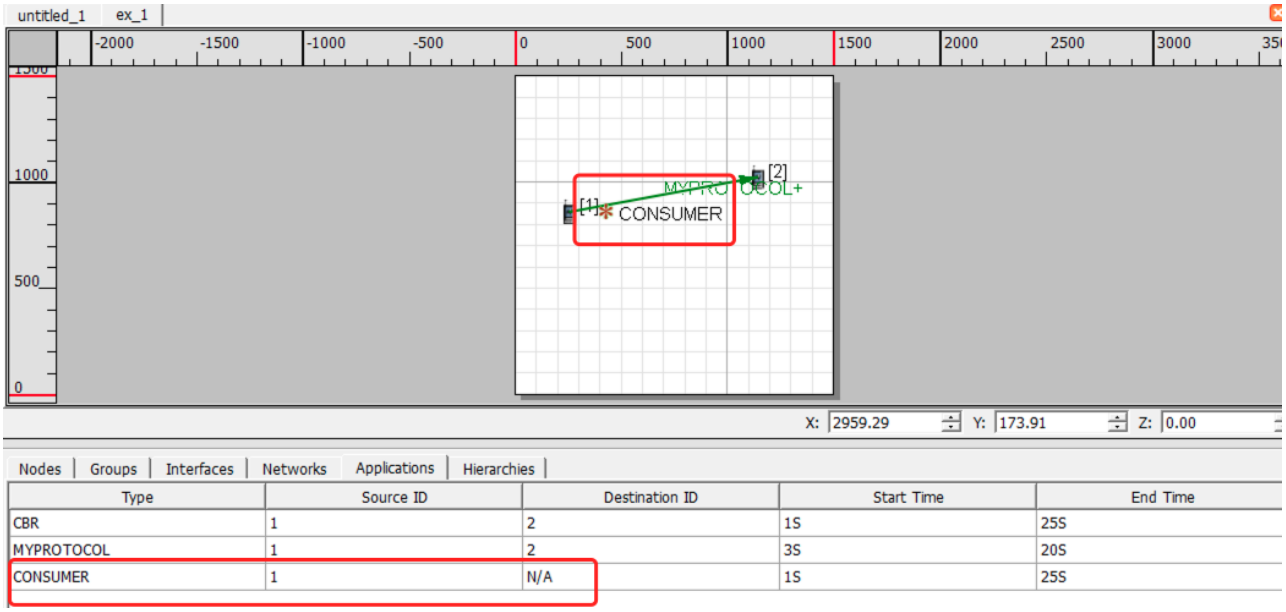
   b. 编辑按钮图标：在gui/icons/3Dvisualizer/icons/ 中复制场景图标文件 sc.png

4. 重启，EXata GUI，已实现：

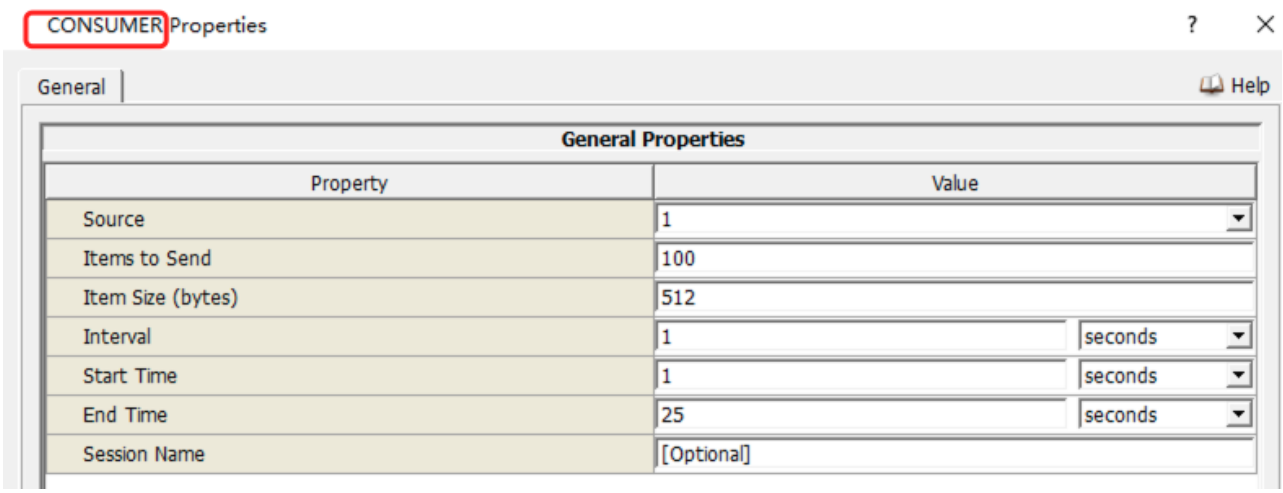
   a. 工具箱“Single Host applications”中已出现 “SC”按钮；



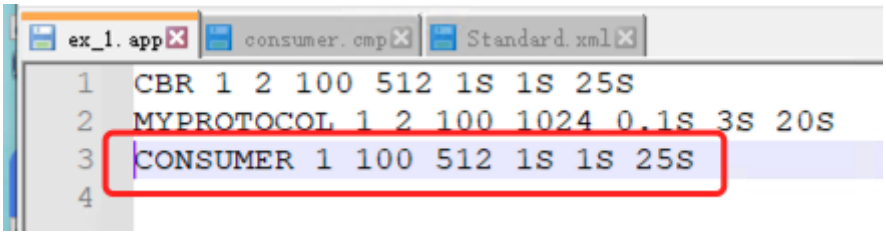
   b. 通过“SC”按钮，可以添加单主机应用



   c. 通过属性框，进行参数设置，按默认参数



   d. 保存后，查看ex_1.app 文件，发现参数已修改。

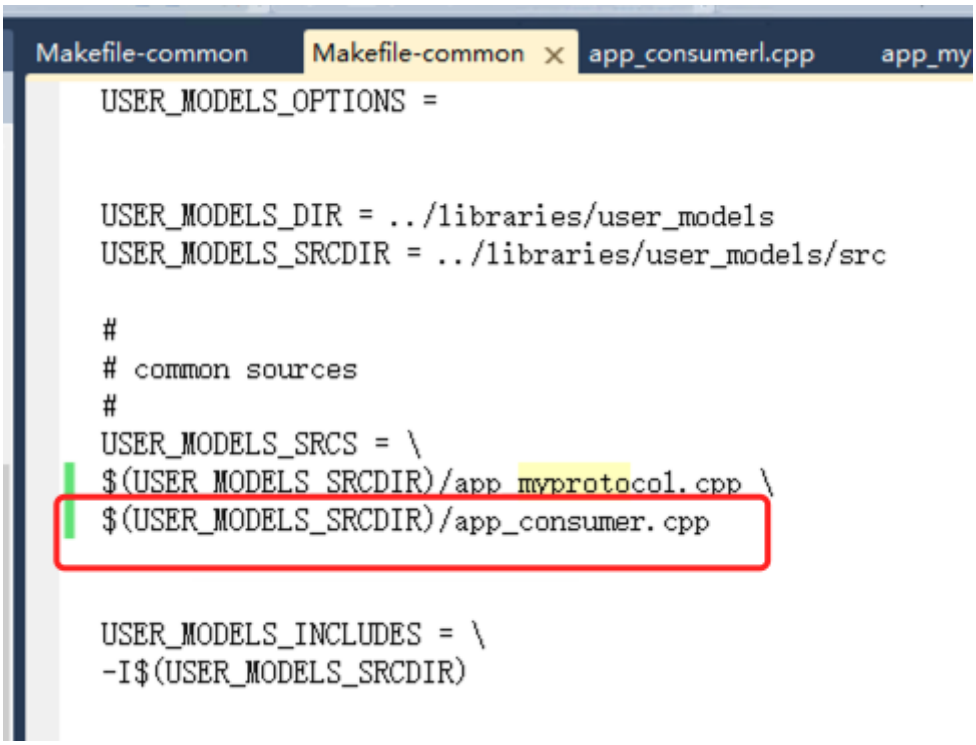


5. 至此，可以通过 GUI 进行初始化参数设置。后面进行代码实现 Consume 的初始化。

1.2.5 Consumer 初始化

继续进行 Consumer 初始化工作，包括Applicaiton layer 中的部分，还有Consumer 自身的实现。

- applicaiton.cpp中：在 CONSUMER 处理部分，进行参数读取和格式转换，调用 Consumer 初始化；
- app_consumer.h中：添加 AppConsumerInit 方法
- 添加app_consumer.cpp： 实现 AppConsumerInit 方法。
- 激活Consumer 协议：
  - user_models/Makefile-common：添加新建的源文件 app_consumer.app. 注意：上一行尾部要加换行符“\”

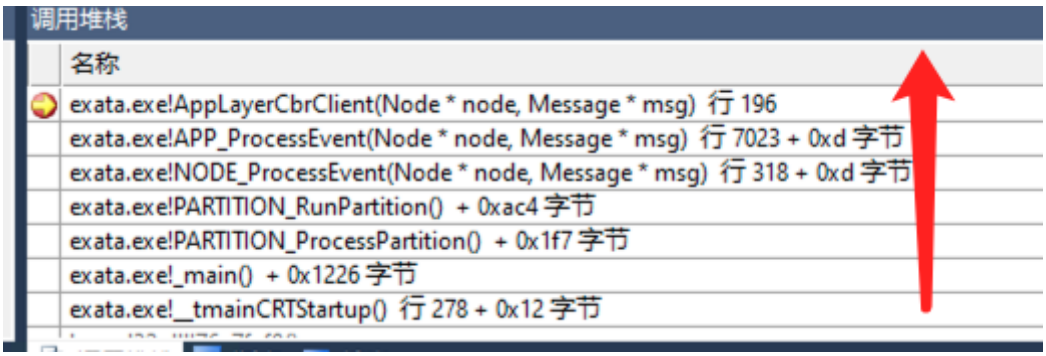


  - nmake 出错：U1073，不知道如何生成 app_consumer.objs！！！
    - 在 VS2010 IDE 中，将 cpp 文件添加到 Project 中，仍不行。
    - 完全重新编译再试：nmake clean；nmake；错误相同！！！这下严重了！
    - 低级错误：app_consumer.cpp文件名错写为app_consumerl.cpp，多了个“l”，更正过来，编译成功！
      ►►►
- app_consumer.h和 cpp 中，添加新建Consumer 实例的方法：AppConsumerNewConsumer，其中
  - 使用 MEM_malloc创建一个 Consumer 实例，类型为 AppDataConsumer；
  - 为各参数赋值
  - 将新实例在该节点：APP_RegisterNewApp

- 初始化继续：回到 AppConsumerInit 方法，创建新的 Consumer 之后，继续完成以下初始化工作：
  - 新建一个统计量：STAT_AppStatistics 新对象（EXata 的应用层统计是用类 STAT_AppStatistics 实现的）
  - 问题：该统计量对象初始化时要求有远端地址！解决办法有二
    - Consumer 参数中添加 remoteAddr；【✅】
    - 派生STAT_AppStatistics 子类。
    - 当下暂时不进行统计【后补】
  - **设定定时器，到期后发包**（第一个Request）
    - 最重要的类：Message。在 EXata 中 Message 既可以代表内部消息（事件），又代表网络中传送的 packet！最强大的功能之一！【后面看怎么充分利用 Message 类】。这里主要是设定定时器消息，在一段时间后发包，通常有以下几个步骤：
    - 分配一个新 Message：*MESSAGE_Alloc*
    - 为消息分配 Info 空间（动态分配的一片内存）：*MESSAGE_AllocInfo*
    - 取回Info 的指针进行赋值：*MESSAGE_ReturnInfo*
    - 设定事件：*MESSAGE_Sends.*
    - 至此，Consumer 初始化完成，下面等待事件到期进行事件处理。

1.2.6 Consumer 事件处理

Consumer 的事件处理行为，由 *AppLayerConsumer* 方法负责，调用流程由Partition-->Node-->AppLayer-->Consumer (这里以 CBR Client为例)



- 在 app_consumer.h 文件中声明该方法，并在 app_consumer.cpp中实现该方法。
- *AppLayerConsumer*(Node *nodePtr, *Message* *msg) 有两个参数：节点和消息
  - 根据msg -> eventType，消息中携带的事件类型分别处理，进入 MSG_APP_TIMER_EXPIRED事件中，
  - 首先，利用 msg 提取 timer = (AppTimer *)*MESSAGE_ReturnInfo*(msg);
  - 再利用 timer 提取 Consumer 实例： *AppConsumerGetConsumer；【补充声明和定义】*
  - 根据 timer 的类型进行处理，主要是 APP_TIMER_SEND_PKT 类型，包括生成下层消息交付下层发送，以及设定下一次发送等。

在进行发包处理实现之前，需要重新定义 Consumer 的配置接口。

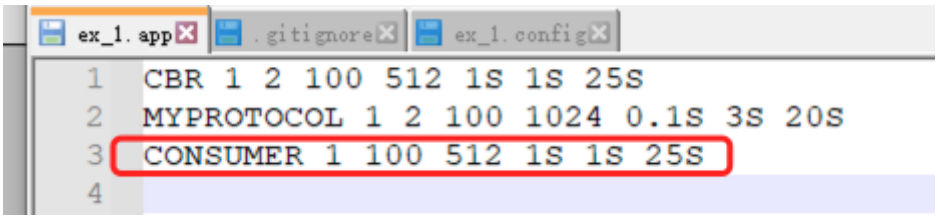
1.2.7 Consumer 配置接口修改

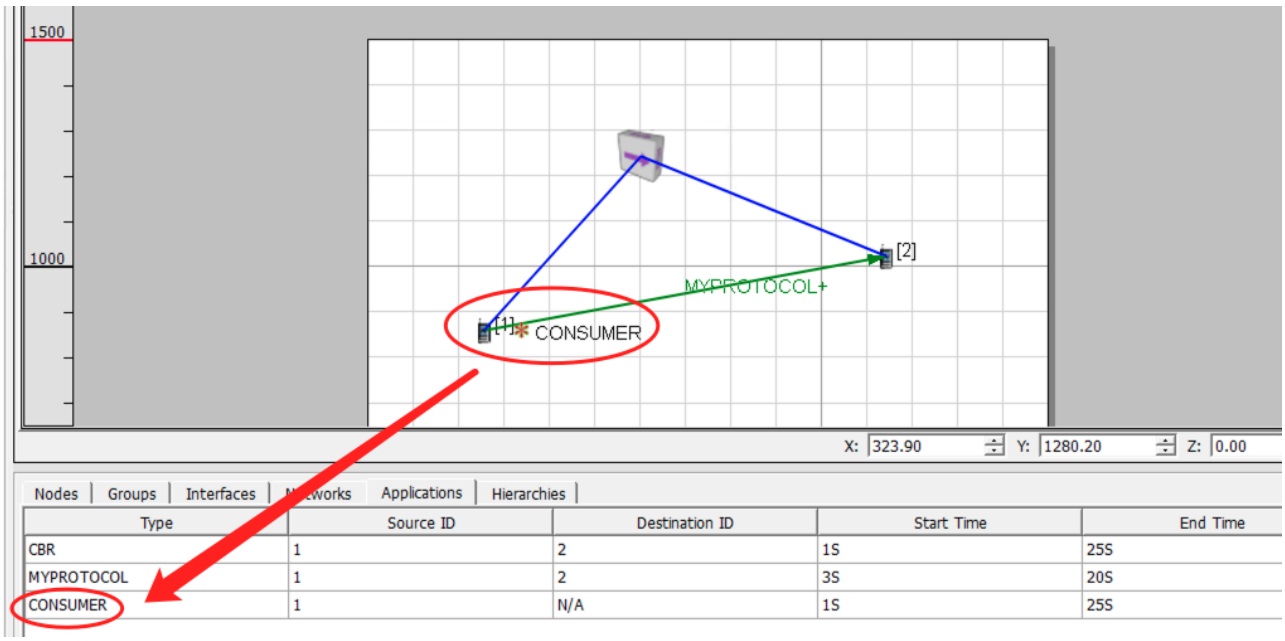
拟将 Consumer 设计成为只发一个 REQUEST 消息，而接收所请求的资源的一个应用，而不是一个可以发出不同分布的 Traffic generator。而traffic generator 可以设计为包括多个 Consumer 对象，分别发不同的请求，采用不同的启动时间来实现。

初步计划，REQUEST 消息的格式如下：

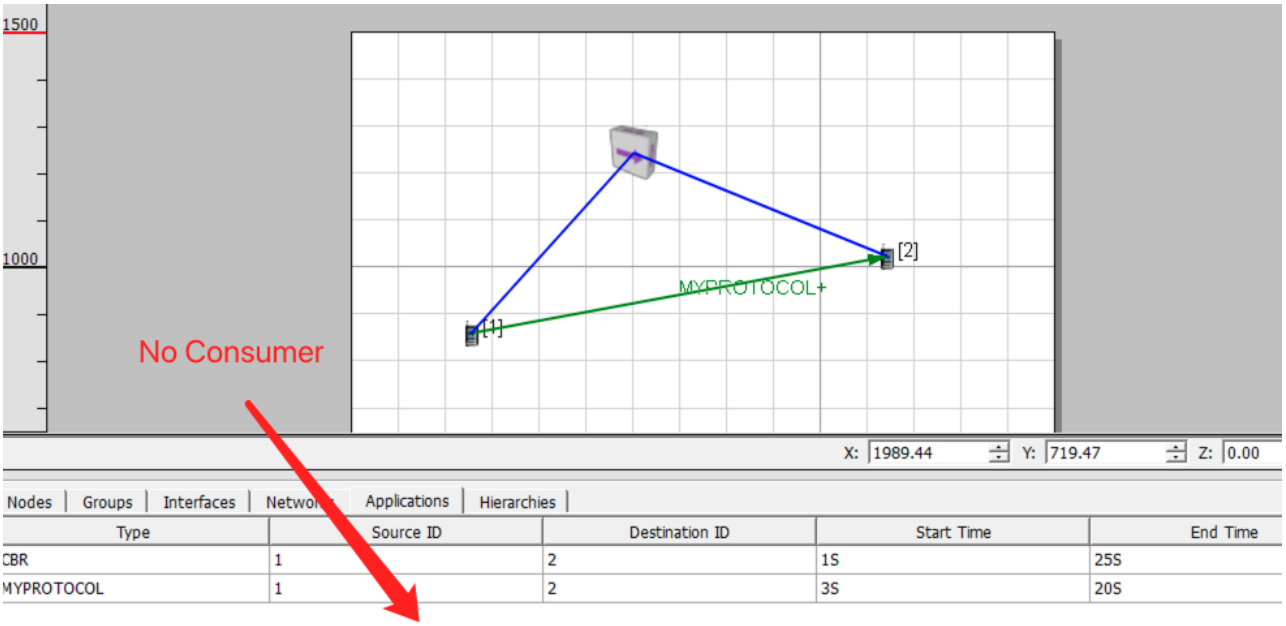
因此，Consumer 不再采用类似 CBR 的配置接口

【问题】EXata GUI 没有保存新添加的 Consumer 应用：添加时在 Application Table 中有，保存完，退出 EXata GUI，下次重新进来时，不显示，此时，ex_1.app 文件中有该配置项。



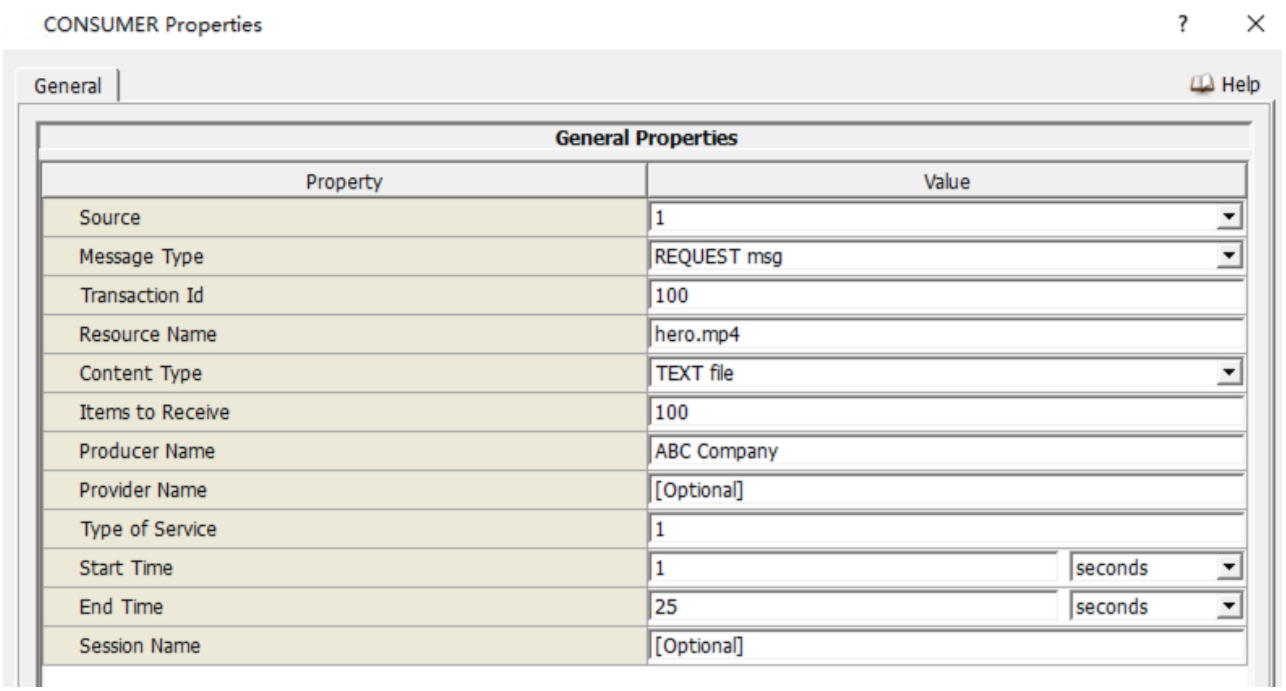


重启 GUI 后，Application Table 中没有 Consumer 的一行！！！ 【什么原因？？？】

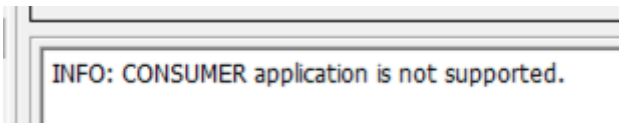


1. 先修改 Consumer 的属性框

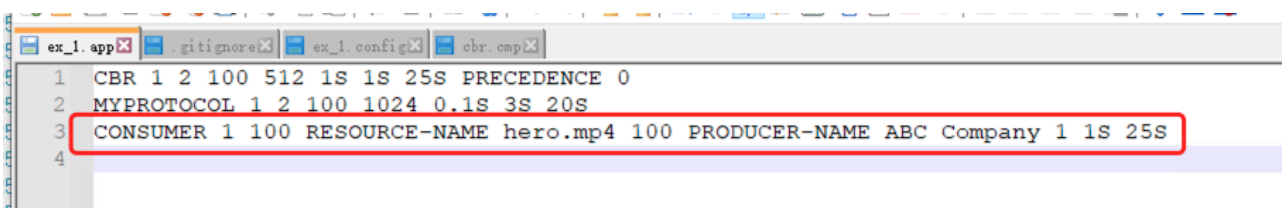
2. 修改 consumer.cmp 文件，使得 Consumer 应用的属性框达到以下效果（对照上面 REQUEST 消息格式）



3. 但此时，GUI 提示以下错误“CONSUMER application is not supported”，【下面修改完善】



4. 保存后ex_1.app内容如下：【问题：没有出现消息类型：REQUEST】【待修改】



5. 修改初始化部分

直接在命令行下运行会提示输入格式错误。该错误出现在 application.cpp 的 3050 行，即对 Consumer 输入行的参数个数初步判断：

```
F:\ex\ex_1>exata ex_1.config
EXata Developer Version 5.1
Kernel Version: 12.10
Build Number: 201310091
Build Date: Oct  9 2013, 18:55:48
EXATA_HOME = D:\Scalable\exata\5.1

Attempting license checkout (should take less than 2 seconds) ...Loading scenario ex_1.config
Error in file ..\main\application.cpp:3050
Wrong CONSUMER configuration format!
CONSUMER <src> <items to send> <item size> <interval> <start time> <end time>
```

6. 首先要搞清楚 COnsumer 属性框内各参数，为什么有些在 EX_1.app 应用配置文件中未出现，有些确实带着 Token出现的。比如当属性框内各参数如下时，

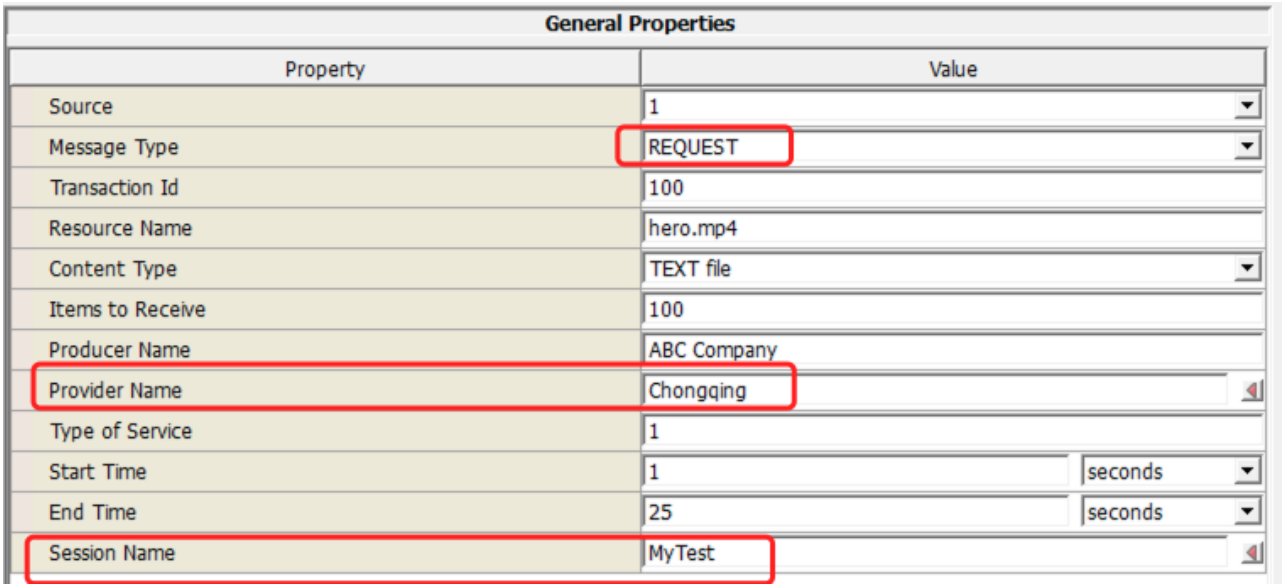
General Properties

Property	Value
Source	1
Message Type	PUSH
Transaction Id	100
Resource Name	hero.mp4
Content Type	TEXT file
Items to Receive	100
Producer Name	ABC Company
Provider Name	[Optional]
Type of Service	1
Start Time	1 seconds
End Time	25 seconds
Session Name	[Optional]

EX_1.app 中 CONSUMER 参数行如下：消息类型（Message Type）和数字类型的参数都不带 Token， 即参数的名称，但象“Resource Name”、“PRODUCER-NAME” 却先给 Token Name 然后是值?

```
3  CONSUMER 1 MSG-PUSH 100 RESOURCE-NAME hero.mp4 100 PRODUCER-NAME ABC Company 1 1S 25S
4
```

7. 如果消息类型取默认的“MSG-REQUEST”，原先为 Optional 的 Provider Name 和 “Session Name” 给出来又会怎么样呢?



General Properties	
Property	Value
Source	1
Message Type	REQUEST
Transaction Id	100
Resource Name	hero.mp4
Content Type	TEXT file
Items to Receive	100
Producer Name	ABC Company
Provider Name	Chongqing
Type of Service	1
Start Time	1 seconds
End Time	25 seconds
Session Name	MyTest

a. CONSUMER 1 MSG-REQUEST 100 RESOURCE-NAME hero.mp4 100 PRODUCER-NAME ABC Company PROVIDER-NAME Chongqing 1 1S 25S APPLICATION-NAME MyTest

b. 有没有 Token 的规则是什么?

i. 找到答案：全部由 CMP 文件中参数的属性控制，当【keyvisable】= true 时，该参数的 token 就会在 app 配置文件中出现；否则不出现。【optional】= true，表示该参数属于可选参数，处理原则：可选参数尽量后置；第二，可选参数，尽量令 【keyvisable】= true，即参数行中首先出现该参数的 Token，否则，不好识别。

c. 参数调整：Transaction Id：不需要，先拿掉。余下各参数：

i. **Source**：必选，可以不见： 【optional】=false， 【keyvisable】=false

ii. **Message Type**：必选，可以不见： 【optional】=false， 【keyvisable】=false

iii. **Resource Name**：必选，可以不见： 【optional】=false， 【keyvisable】=false， 【spaceAllowed】=false【不允许有空格】

iv. **Content Type**：必选，可以不见： 【optional】=false， 【keyvisable】=false

v. **Producer Name**：必选，可以不见： 【optional】=false， 【keyvisable】=false， 【spaceAllowed】=false【不允许有空格】

vi. **Total Items**：必选，可以不见： 【optional】=false， 【keyvisable】=false

vii. **Chunk size**：必选，可以不见： 【optional】=false， 【keyvisable】=false； 0-1024

viii. **Start time**：必选，可以不见： 【optional】=false， 【keyvisable】=false

ix. **End Time**：必选，可以不见： 【optional】=false， 【keyvisable】=false

x. **Type of ervice**：必选，可以不见： 【optional】=false， 【keyvisable】=false

1. 1: time-constraint reliable，Latency Bound （ms）：输入最大允许时限
2. 2: reliable：无进一步参数
3. 3: best-effort：无进一步参数

xi. 【Optional】Provider Name：可选，可见： 【optional】=true， 【keyvisable】=true 【spaceAllowed】=false【不允许有空格】

xii. 【optional】Session Name：可选，可见： 【optional】=true， 【keyvisable】=true

【spaceAllowed】=false【不允许有空格】

xiii. 为简便，所有字符串都不允许有空格！

xiv. 修改后，在3 个节点上分布加载 3 个 Consumer，设定属性如下：

```
ex_1.app   consumer.cmp
1  CBR 1 2 100 512 1S 1S 25S PRECEDENCE 0
2  MYPROTOCOL 1 2 100 1024 0.1S 3S 20S
3  CONSUMER 1 REQUEST hero.mp4 AUDIO/VIDEO ABC 100 1S 25S RELIABLE
4  CONSUMER 2 REQUEST hero2.mp4 AUDIO/VIDEO ABC 100 1S 25S TIME-CONSTRAINT 10 PROVIDER CQUPT APPLICATION-NAME MyHero
5  CONSUMER 3 POST hero3.png IMAGE N3 100 1S 25S BEST-EFFORT APPLICATION-NAME pushhero3
6
```

8. 【问题】EXata GUI 提示出错，“CONSUMER DYNAMICADDRESS is not supported”；这里跟前面的“CONSUMER applicaiton is not supported”不同，多了个“DYNAMICADDRESS”，Why?

   a. 分析问题：这个 INFO 类告警出在什么阶段？由于无法在 VS 中调试EXataGUI，不好判断。

   b. 在 ex_1.app中删掉 CONSUMER 一行，重新启动 GUI，打开ex_1.config，没有该警告信息！说明：肯定是在读入CONSUMER 行给出的信息！因此，判断应该是在 GUI 显示 Application 信息时给出的 Info。问题来了，GUI 是如何判断系统不支持 CONSUMER 协议的？哪里保存的有所支持的协议呢？而且“CONSUMER-DYNAMICAPPLICATIOINS”后面的 DYNAMICADDRESS 哪里来的？

   c. 会不会是因为协议参数格式Simulator 读入时出错，而在 GUI 给予提示？但是在格式修改前也有此提示。==》先把格式修改后 Consumer 协议参数 读入格式的问题解决了再说！

1.2.8 Consumer 初始化修改

由于输入参数格式进行了修改，Consumer 在初始化时需要重新进行参数读入的调整。修改文件为<applicaiton.cpp>。

- GUI 不支持 Consumer 的提示信息，通过字符串搜索，只存在于 EXataGUI.exe 中，但该程序无法修改和调试。方案：先把输入格式改好。

几个需要修改的地方：

1. Application.cpp 中：App_InitializeApplications
2. AppConsumerInit：修改了接口，未实现完
3. AppConsumerNewConsumer：修改了接口，尚未开始实现。

修改完毕，调试发现错误：在ex_1.app读入时参数个数不对，该有 13 个参数，但返回只有 10 个！！！ 【参数个数不对的问题】

```
CONSUMER 4 GET hero.mp4 AUDIO/VIDEO ABC 100 512 1S 25S TIME-CONSTRAINT 10MS APPLICATION-NAME MeFirstConsumer
```

13

4. 调试发现这里的处理裁剪了输入的参数行，添加跳过 CONSUMER 协议部分，即可保证输入字符串不被修改

```
if ((strcmp(appStr, "SUPER-APPLICATION") != 0)
    && (strcmp(appStr, "THREADED-APP") != 0)
    // LuoJT: skip my CONSUMER
    && (strcmp(appStr, "CONSUMER") != 0))
{
    while (appNamePtr != valuePtr2)
    {
        *appNamePtr = ' ';
        appNamePtr ++;
    }
}
```

5. 【Tips】在做字符串比较时，一定要牢记：字符串相等时返回值为“零”，而不是 BOOL 型！正确的做法是下面的写法：

```
{
    if (strcmp(tosStr, "RELIABLE") != 0
    && strcmp(tosStr, "BEST-EFFORT") != 0)
    {
        ReportErrorConsumerFormat();
    }
```

6. 至此，初始化修改完成，并且应用协议的参数行能正确输入！目前已
7. 下面需要完成消息处理部分，由于向系统发送了 APP_CONSUMER 协议的消息，但在应用层部分没有实现，此时，会提示收到未知协议的消息：

```
Initialization completed in 0.186 sec at 2023-01-03 21:09:43.807
Current Sim Time[s] =    0.324682905  Real Time[s] =    0  Completed  1%
Current Sim Time[s] =    0.613889680  Real Time[s] =    0  Completed  2%
Current Sim Time[s] =    0.997545544  Real Time[s] =    0  Completed  3%
Error in file ..\main\application.cpp:7690
Application layer receives a message with unidentified Protocol = 8520
```

1.2.9 CONSUMER 事件处理

还是先从应用层的事件处理开始。

1. 首先把 Consumer 的远端地址设为广播和任播地址：参照《Programmer Guide》 P.137
    a. 在 AppConsumerInit 中调用 AppConsumerNewConsumer 时设定 remoteAddress 为 ANY_DEST；

```
119
120     Address remoteAddr = ANY_DEST; // a broadcast remote address
121
122     // Create an instance of Consumer
123     consumerPtr = AppConsumerNewConsumer(
124         node,
125         addr,
126         remoteAddr,
127         msgType,
128         rscName,
129         startTime,
130         endTime,
131         //tos,
132         appName);
```

b. 上面的直接赋值格式有误，需要进行格式转换，要借助于fileio.h中的io 函数进行转换：

```
119
120     // Set the remote address to be a broadcast one
121     Address remoteAddr;
122     SetIPv4AddressInfo(&remoteAddr, ANY_ADDRESS);
123
```

c.

2. 应用层事件处理：APP_ProcessEvent

a. 在switch（protocolType）中增加一个 APP_CONSUMER 的 case

```
// LuoJT: CONSUMER
case APP_CONSUMER:
{
    AppLayerConsumer(node, msg);
    break;
}
```

3. 实现 AppLayerConsumer （Node*, Message*）方法，在app_consumer.h 中声明，在 cpp 文件中实现。

a. 首先要根据消息类型做不同的处理。

b. 第一个是 定时器超时事件：MSG_APP_TimerExpired：

i. 从msg 中提取该 timer，并转换成AppTimer；

ii. 然后，根据 timer 中的sourcePort 信息，提取 consumer 实例；

c. 创建 UDP 消息，并发送【待补充】。。。。。

4. 对比 CONSUMER 和 MYPROTOCOL，查找 GUI 不支持 CONSUMER 的原因【未定位！！！】

myprotocol.cmp	D:\Scalable\exata\5.1\gui\settings\components\	4.61 kB	2022/12/20 2...	2022/12/20 21:39:56	CMP 文件
application.cpp	D:\Scalable\exata\5.1\main\	306.12 kB	2023/1/3 21:...	2022/1/24 15:38:50	C++ Source
app_myprotocol.cpp	D:\Scalable\exata\5.1\libraries\user_models\src\	19.34 kB	2022/12/29 1...	2022/12/13 14:35:22	C++ Source
trace.h	D:\Scalable\exata\5.1\include\	15.90 kB	2022/12/28 2...	2022/1/24 15:38:46	C/C++ Header
application.h	D:\Scalable\exata\5.1\include\	20.79 kB	2022/12/28 2...	2022/1/24 15:38:46	C/C++ Header
app_myprotocol.h	D:\Scalable\exata\5.1\libraries\user_models\src\	3.67 kB	2022/12/18 2...	2022/12/13 14:33:07	C/C++ Header
api.h	D:\Scalable\exata\5.1\include\	63.66 kB	2022/12/15 1...	2022/1/24 15:38:45	C/C++ Header

1.2.10 终止化