

DDS应用开发指导

南京臻融科技有限公司

黄志勇

概述

- 软件开发流程
- 二次开发库流程

DDS应用开发

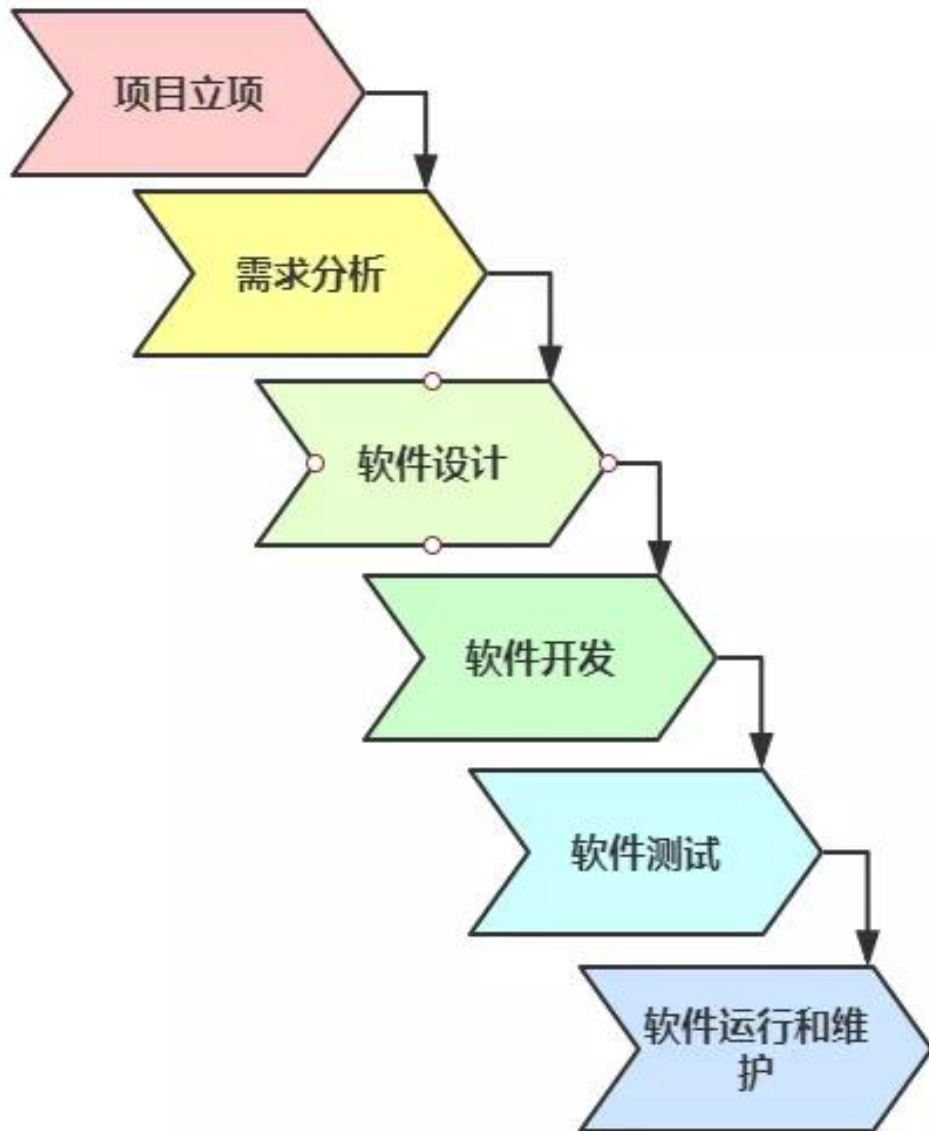
- DDS文档
- DDS概念理解
- DDS接口

开发实例

- 需求
- 设计
- 编码
- 调试

应用开发概述

- 瀑布模型

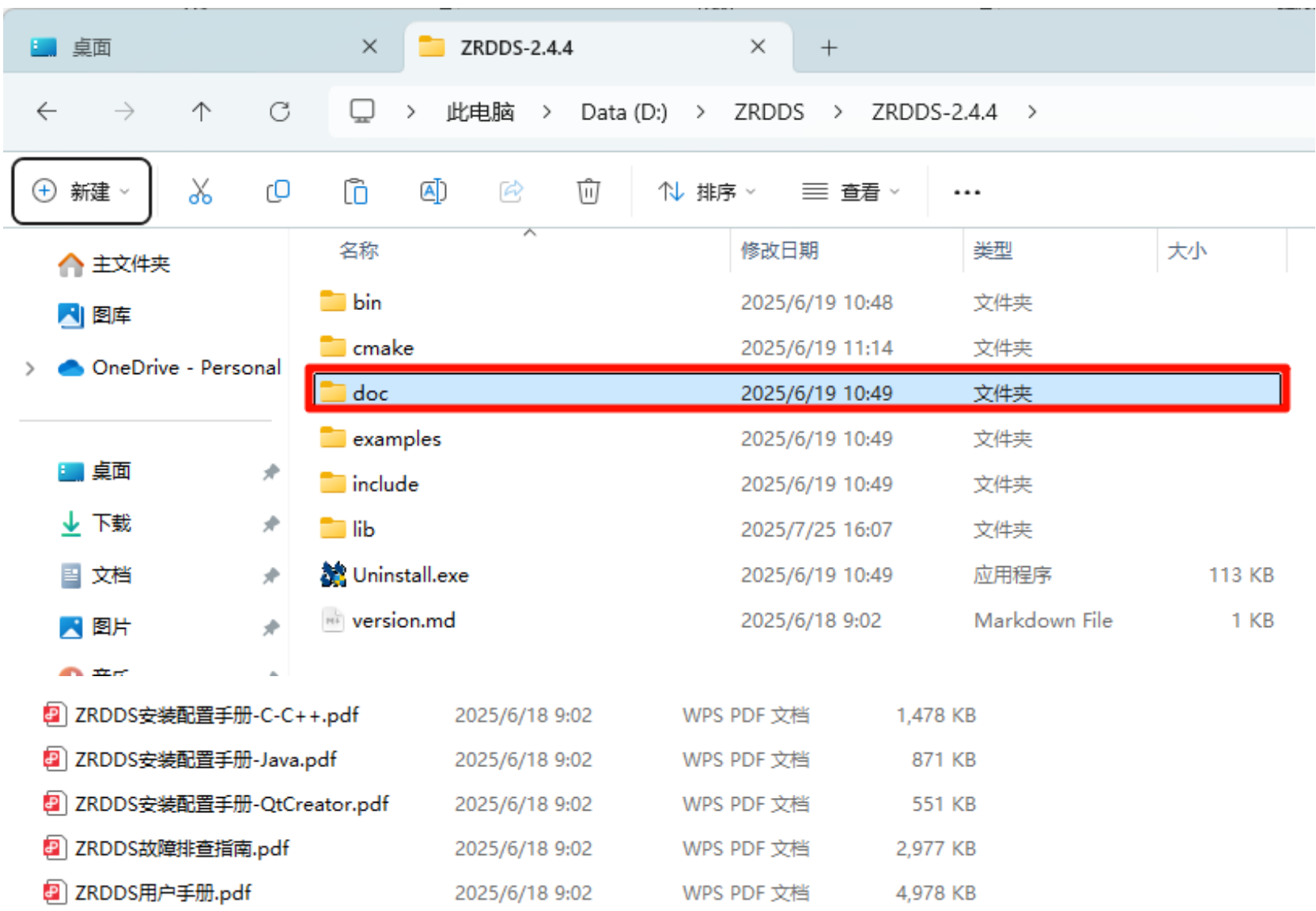


- 敏捷模型

- 快速迭代
- 程序员的主观能动性，以及程序员之间的互动，优于既定流程和工具。
- 软件能够运行，优于详尽的文档。
- 跟客户的密切协作，优于合同和谈判。
- 能够响应变化，优于遵循计划。

- 文档
 - 阅读 “快速入门”
 - 试跑 “示例代码”
 - 理解 “概念定义”
 - 详看 “函数接口”

- 阅读文档
 - Windows开发：
<https://learn.microsoft.com/zh-cn/windows/apps/>
 - Linux开发：man 手册
<https://man7.org/linux/man-pages/index.html>
- 第三方库：User Manual（用户手册）、[开发者手册](#)。



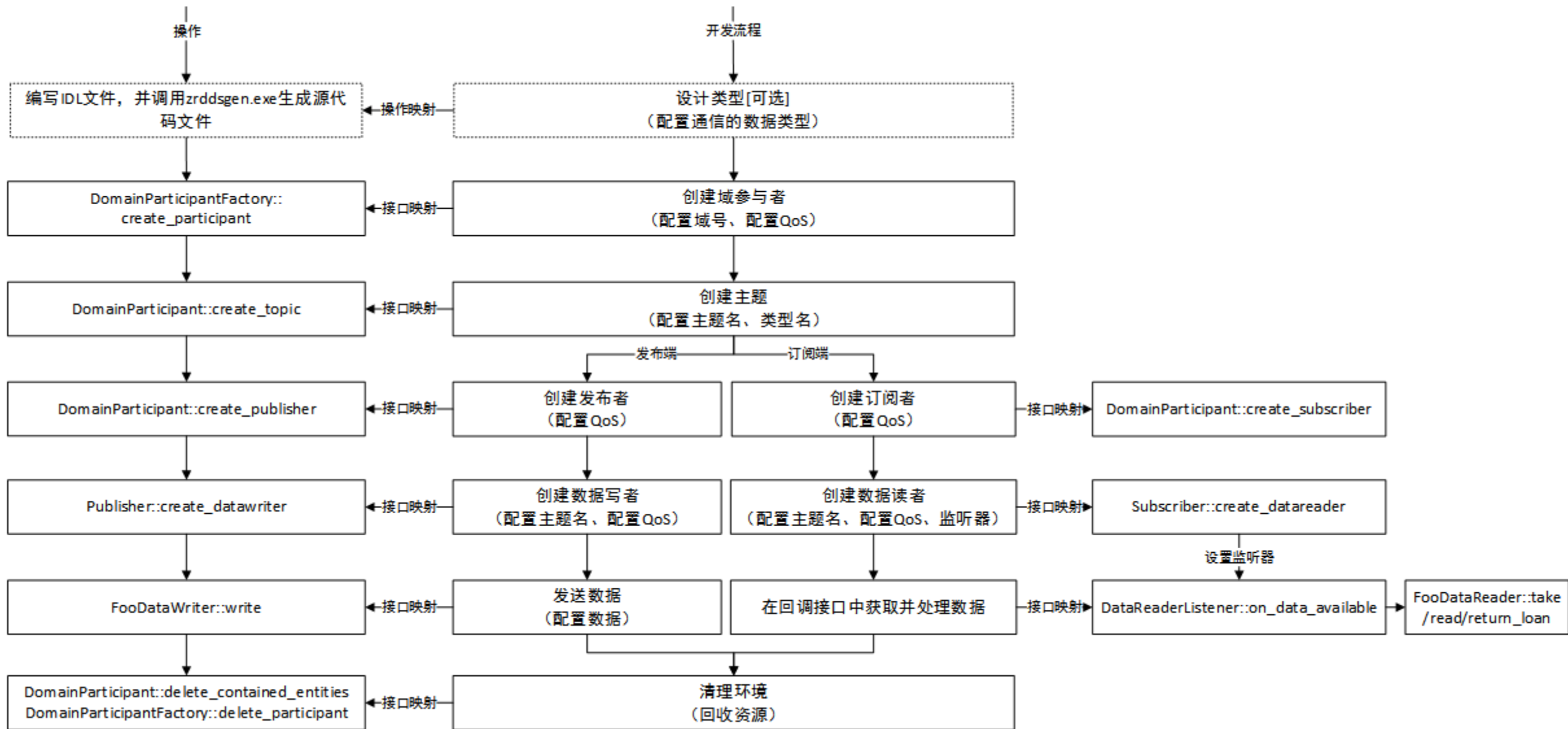
The screenshot shows a Windows File Explorer window with the address bar set to 'Data (D:) > ZRDDS > ZRDDS-2.4.4'. The left sidebar shows the '桌面' (Desktop) folder selected. The main pane displays a list of files and folders. The 'doc' folder is highlighted with a red rectangle.

名称	修改日期	类型	大小
bin	2025/6/19 10:48	文件夹	
cmake	2025/6/19 11:14	文件夹	
doc	2025/6/19 10:49	文件夹	
examples	2025/6/19 10:49	文件夹	
include	2025/6/19 10:49	文件夹	
lib	2025/7/25 16:07	文件夹	
Uninstall.exe	2025/6/19 10:49	应用程序	113 KB
version.md	2025/6/18 9:02	Markdown File	1 KB

名称	修改日期	类型	大小
ZRDDS安装配置手册-C-C++.pdf	2025/6/18 9:02	WPS PDF 文档	1,478 KB
ZRDDS安装配置手册-Java.pdf	2025/6/18 9:02	WPS PDF 文档	871 KB
ZRDDS安装配置手册-QtCreator.pdf	2025/6/18 9:02	WPS PDF 文档	551 KB
ZRDDS故障排查指南.pdf	2025/6/18 9:02	WPS PDF 文档	2,977 KB
ZRDDS用户手册.pdf	2025/6/18 9:02	WPS PDF 文档	4,978 KB

DDS应用开发

- 快速入门



- 示例工程（\$ZRDDS_HOME/examples/cpp/ide/x64Win64VS2015.sln）

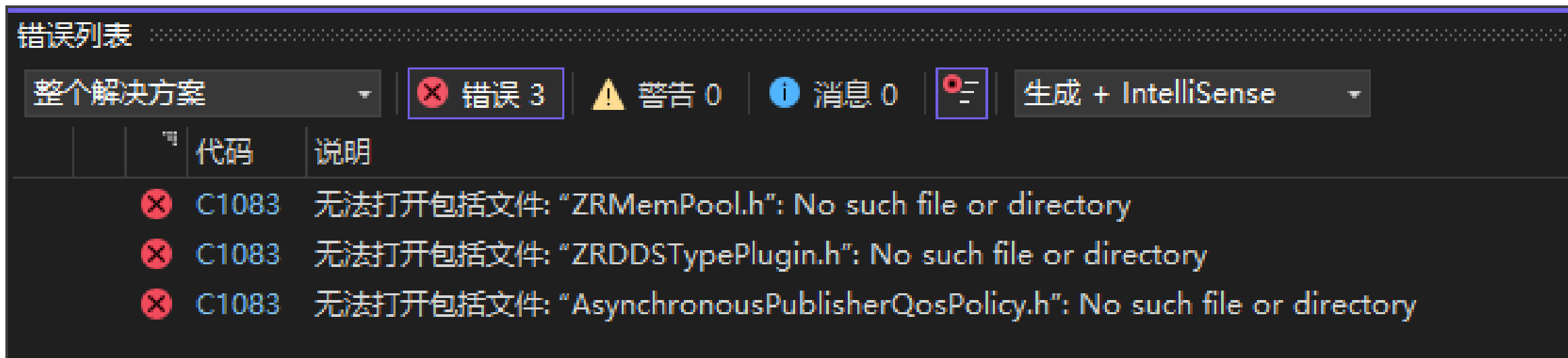
```
20 int main(int argc, char** argv)
21 {
22     ...// 域号
23     ...const int domain_id = 80;
24     ...ReturnCode_t rtn;
25
26     ...if(TheParticipantFactory == NULL)
27     {
28         ...printf("get instance failed\n");
29         ...return -1;
30     }
31
32     ...// 创建域参与者
33     ...DomainParticipant* dp = TheParticipantFactory->create_participant(
34         ...DomainId_t(domain_id), DOMAINPARTICIPANT_QOS_DEFAULT, NULL, STATUS_MASK_NONE);
35     ...if (dp == NULL)
36     {
37         ...printf("create dp failed\n");
38         ...return -1;
39     }
40
41     ...// 注册数据类型
42     ...rtn = ShapeTypeTypeSupport::get_instance()->register_type(dp, NULL);
43     ...if (rtn != RETCODE_OK)
44     {
45         ...printf("register type failed\n");
46         ...return -1;
47     }
48
49     ...// 创建主题
50     ...Topic* tp = dp->create_topic(
51         ..."DATA RECEIVE BY LISTENER",
52         ...ShapeTypeTypeSupport::get_instance()->get_type_name(),
53         ...TOPIC_QOS_DEFAULT,
```

```
D:\ZRDDS\ZRDDS-2.4.4\exam
***** 2025-08-13 17:01:02:045 *****
Current ZRDDS version(2.4.4-rlc954a9) for 528-demo was compiled at Jul 29 2025 13:47:44
*****

D:\ZRDDS\ZRDDS-2.4.4\exam
sample->z(24): Data Receive by Listener
sample->x: 30
sample->y: 0
sample->z(24): Data Receive by Listener
wait for receive data
sample->x: 31
sample->y: 0
sample->z(24): Data Receive by Listener
sample->x: 32
sample->y: 0
sample->z(24): Data Receive by Listener
wait for receive data
```

- 找不到 Windows SDK 版本 8.1。请安装所需版本的 Windows SDK，或者在项目属性页中或通过右键单击解决方案并选择“重定解决方案目标”来更改 SDK 版本。
 - 选择自己计算机安装了的SDK版本；



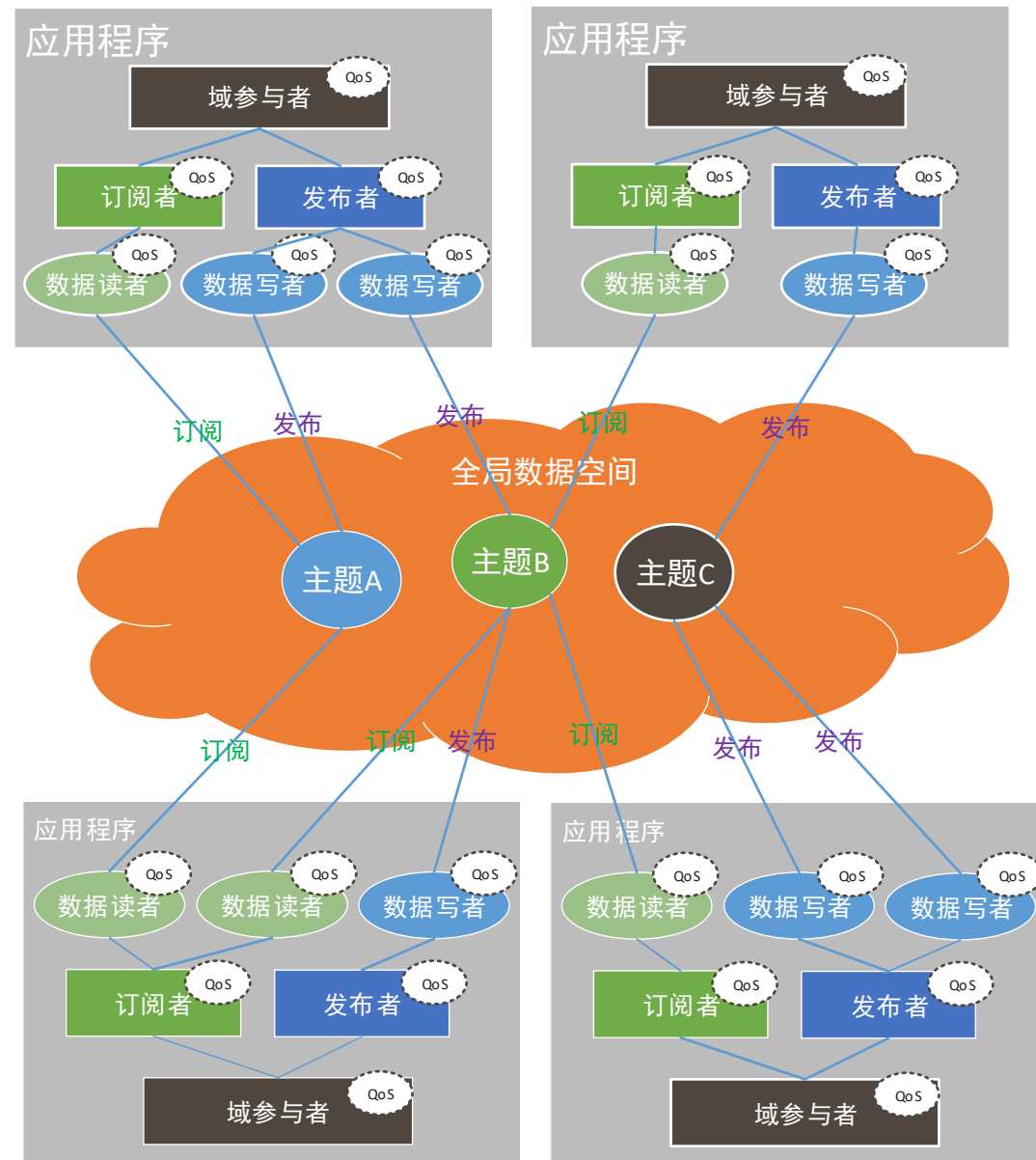


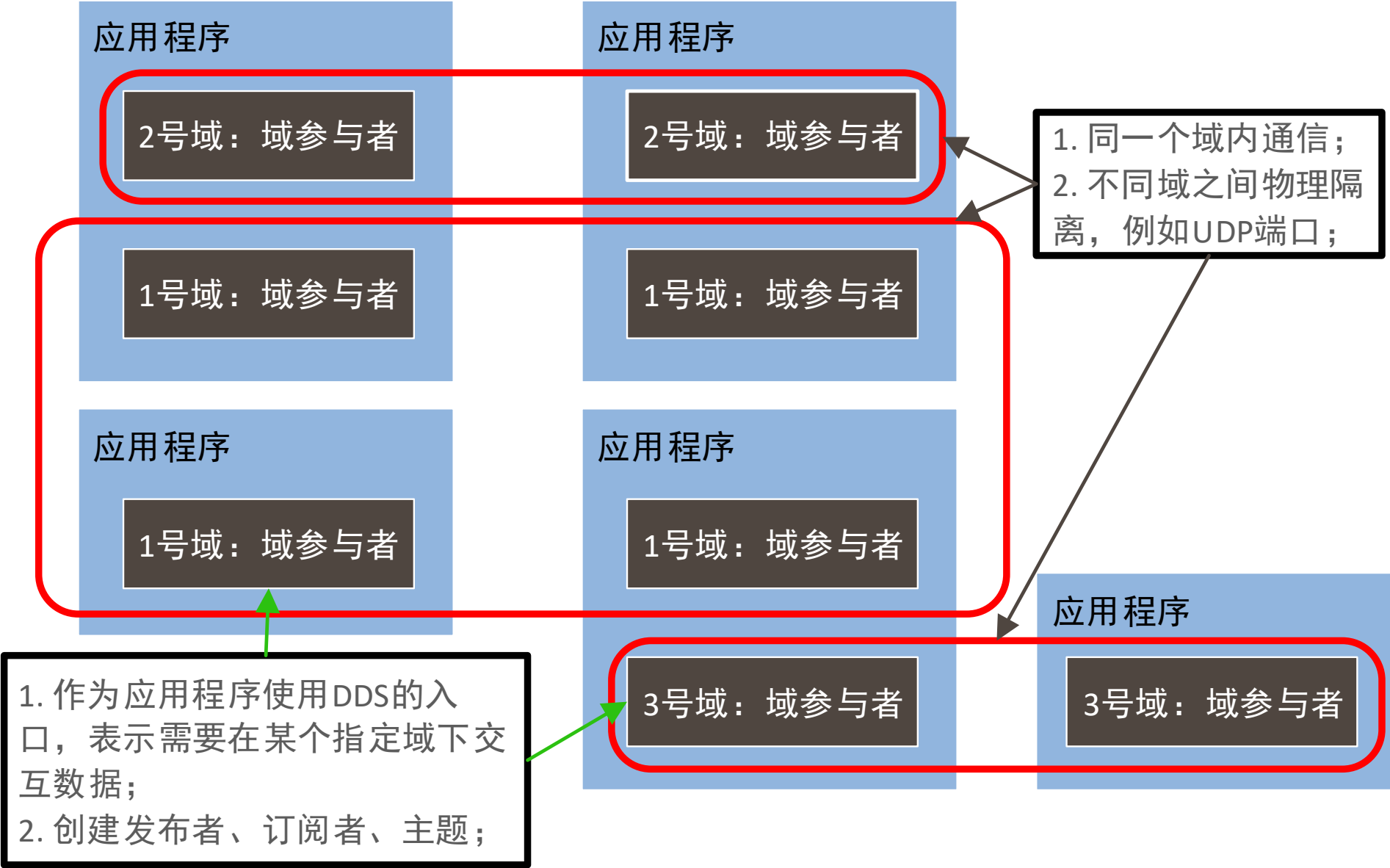
- 检查环境变量ZRDDS_HOME是否有效，或者直接把包含目录修改为绝对路径
- 低于VS2019版本的Visual Studio，无法链接VS2019版本的库

- 无法找到ZRDDSCppzd_VS2015.lib
 - 示例原本的工程使用的是VS2015版本，所以需要修改为我们本次提供的VS2019版本的库，从VS2015开始的lib库，都是前向兼容的
- 运行zrddsgen.exe缺少dll
 - 为保持最大兼容性，zrddsgen.exe是通过VS2013生成的，所以需要安装VS2013的运行环境；

概念理解——全局数据空间

- 按主题组织系统中的所有数据，“存储”在全局数据空间中；
- 发布需要分享的主题数据；
- 订阅自己感兴趣的主体数据；
- 无需其他应用程序的存在性以及位置信息，实现空间、时间上解耦合；
- “域参与者” - “发布者” - “数据写者” 实体层次；





```
DomainParticipant* dp = TheParticipantFactory.create_participant(  
    DomainId_t(0),  
    DOMAINPARTICIPANT_QOS_DEFAULT,  
    NULL, STATUS_MASK_NONE);  
if (dp == NULL)  
{  
    printf("Create DomainParticipant failed\n");  
    return -1;  
}
```

域号

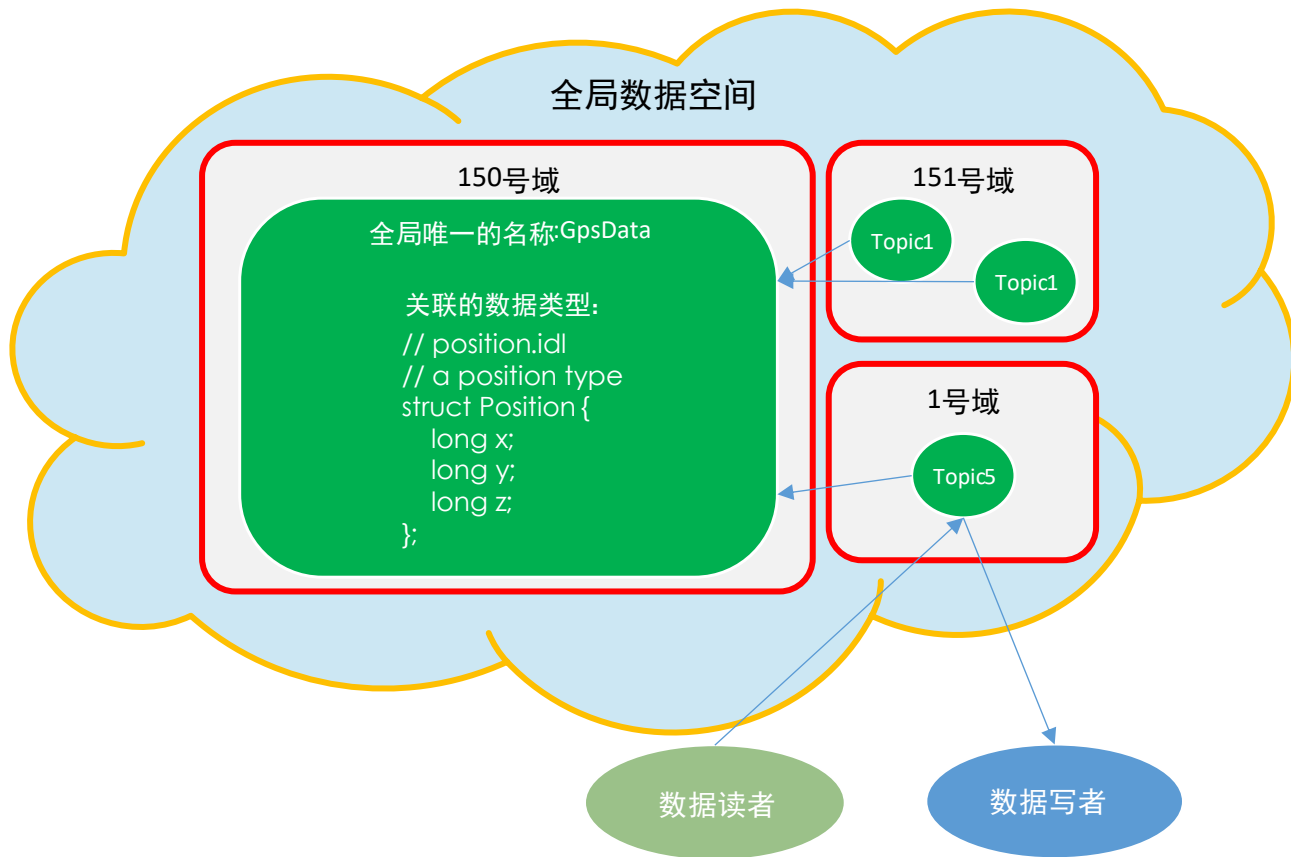
- 使用DDS之前应为准备共享的/感兴趣的数
据定义类型；
 - 定义IDL文件；
- 优点
 - 用户与DDS交互都是通过带类型的接口，使程
序清晰易于使用；
 - 将序列化、反序列化的工作从用户转移到DDS
（涉及到不同平台时会变得复杂），简化用户
编程难度以及工作量；
- 缺点
 - 不够灵活，系统需预先设计数据类型；
 - 数据类型修改比较复杂；
 - ZRDDS提供“动态数据类型”避免该问题；

```
// position.idl  
// a position type  
struct Position {  
    long x;  
    long y;  
    long z;  
};
```

```
// 发布数据  
Position sample;  
PositionDataWriter* writer = ...;  
writer->write(sample, HANDLE_NIL_NATIVE);
```

```
// 订阅数据  
Position sample;  
SampleInfo info;  
PositionDataReader* reader = ...;  
reader->read_next_sample(sample, info);
```


- “连接”发布者与订阅者；
- 域内唯一的名称；
- 关联的数据类型；



用户定义
数据类型
必须一致

```
if (CharSeqTypeTypeSupport::get_instance()->register_type(
    dp, CharSeqTypeTypeSupport::get_instance()->get_type_name()) != RETCODE_OK)
{
    printf("Register data type failed\n");
    dp->delete_contained_entities();
    TheParticipantFactory.delete_participant(dp);
    return -1;
}
```

```
Topic* topic = dp->create_topic(POSITIVE_TOPIC_NAME,
    CharSeqTypeTypeSupport::get_instance()->get_type_name(),
    TOPIC_QOS_DEFAULT, NULL, STATUS_MASK_NONE);
if (topic == NULL)
{
    printf("Create topic failed\n");
    dp->delete_contained_entities();
    TheParticipantFactory.delete_participant(dp);
    return -1;
}
```

- 应用创建发布者表示自身想向系统共享数据；
- 应用通过发布者创建数据写者；
 - 数据写者向系统的全局数据空间“写”数据；
 - 发布者可以创建多个数据写者；
 - 每个数据写者关联一个主题；

- 应用创建订阅者表示自身想从系统获取共享数据；
- 应用通过订阅者创建数据读者；
 - 数据读者从系统的全局数据空间“读”数据；
 - 订阅者可以创建多个数据读者；
 - 每个数据读者关联一个主题；

- 创建发布者 (Publisher)

```
Publisher* pub = dp->create_publisher(PUBLISHER_QOS_DEFAULT, NULL, STATUS_MASK_NONE);  
if (pub == NULL)  
{  
    printf("Create Publisher failed\n");  
    TheParticipantFactory.delete_participant(dp);  
    return -1;  
}
```

- 创建订阅者

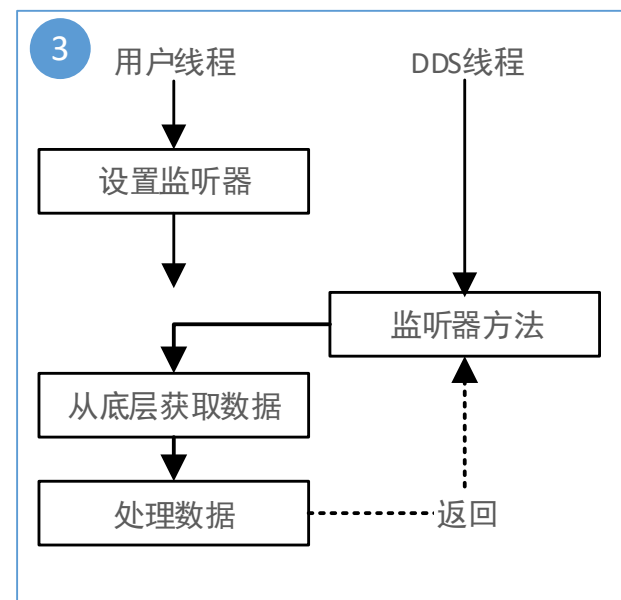
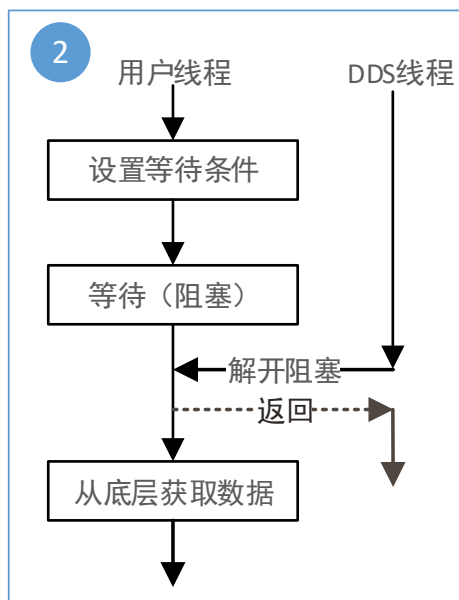
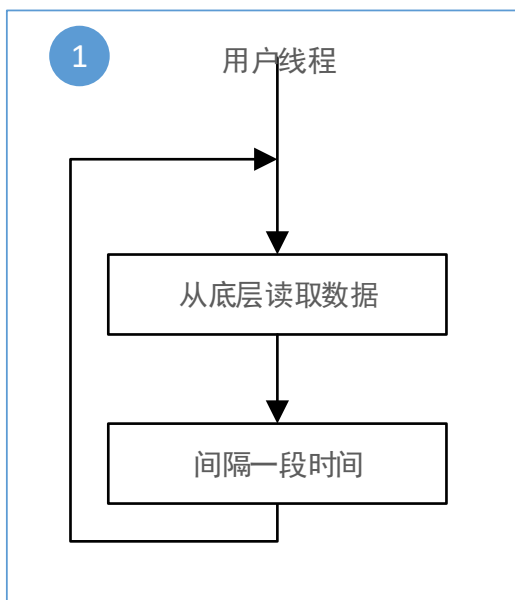
```
Subscriber* sub = dp->create_subscriber(SUBSCRIBER_QOS_DEFAULT, NULL, STATUS_MASK_NONE);  
if (sub == NULL)  
{  
    printf("Create Subscriber failed\n");  
    dp->delete_contained_entities();  
    TheParticipantFactory.delete_participant(dp);  
    return -1;  
}
```

- 创建数据写者 (DataWriter)

用户定义
数据类型

```
DataWriter* rawDw = pub->create_datawriter(  
    topic, DATAWRITER_QOS_DEFAULT, NULL, STATUS_MASK_NONE);  
if (rawDw == NULL)  
{  
    printf("Create DataWriter failed\n");  
    dp->delete_contained_entities();  
    TheParticipantFactory.delete_participant(dp);  
    return -1;  
}  
CharSeqTypeDataWriter* dw = dynamic_cast<CharSeqTypeDataWriter*>(rawDw);  
if (dw == NULL)  
{  
    printf("Cast DataWriter failed\n");  
    pub->delete_contained_entities();  
    dp->delete_contained_entities();  
    TheParticipantFactory.delete_participant(dp);  
    return -1;  
}
```

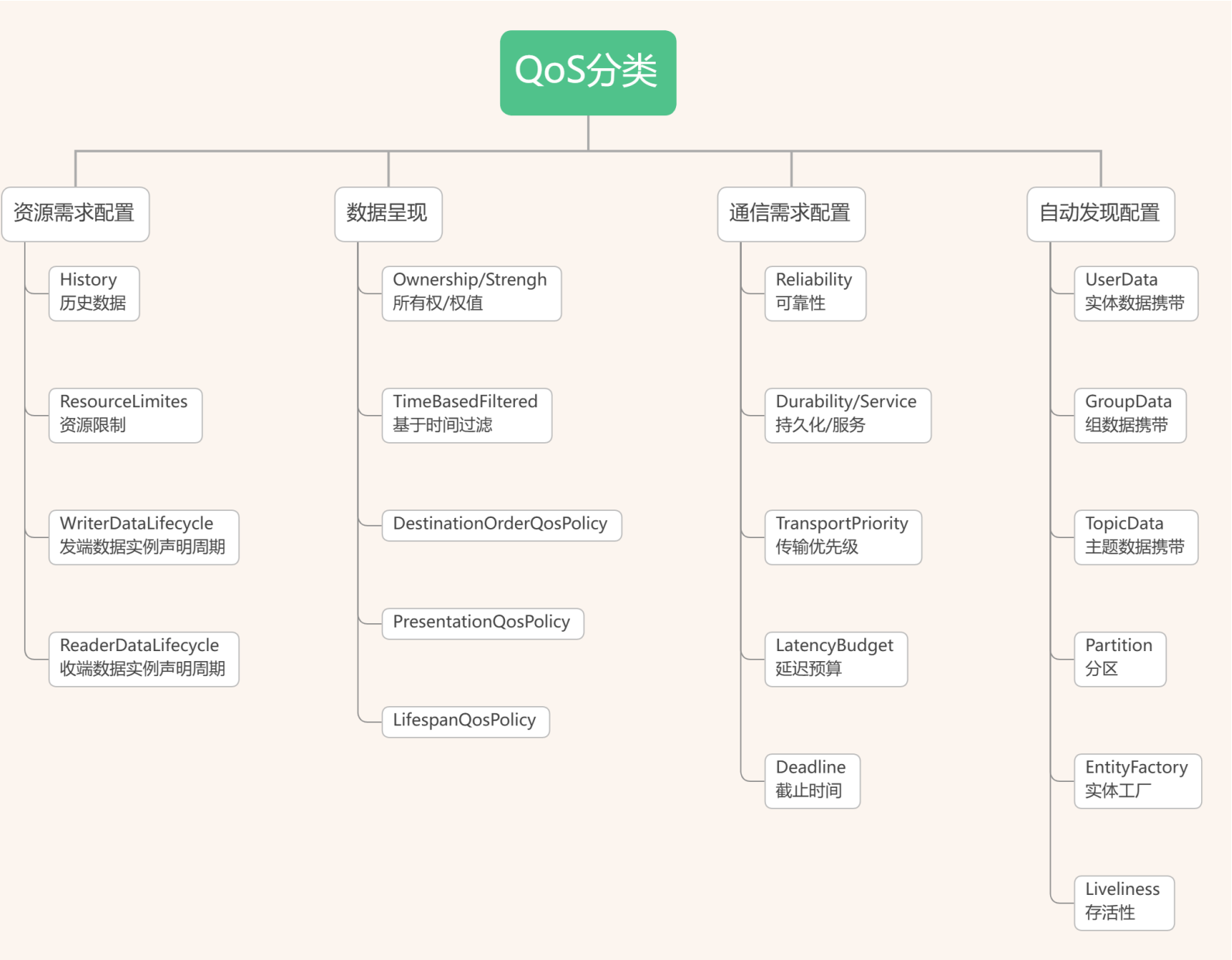
- 同步查询
 - 在用户线程中轮询尝试从数据读者中读取数据；
- 同步等待
 - 设置条件等待（阻塞），当有满足条件的数据到达时，DDS会解开阻塞，再调用数据读者接口读取数据；
- 异步监听
 - 非阻塞，为数据读者设置监听器，当有数据到达时，DDS会调用监听器的方法，用户在方法中去从数据读者中读取数据；



```
Mylistener* drListener = new Mylistener;
DataReader* dr = sub->create_datareader(
    topic, DATAREADER_QOS_DEFAULT, drListener, STATUS_MASK_ALL);
if (dr == NULL)
{
    printf("Create DataReader failed\n");
    sub->delete_contained_entities();
    dp->delete_contained_entities();
    TheParticipantFactory.delete_participant(dp);
    return -1;
}
```



```
class Mylistener :public DataReaderListener
{
    void on_data_available(DataReader *the_reader)
    {
        printf("received data : \n");
        CharSeqTypeDataReader *_reader = (CharSeqTypeDataReader*)the_reader;
        CharSeqTypeSeq data_values;
        SampleInfoSeq sample_infos;
        ReturnCode_t rtn;
        if (NULL == the_reader)
        {
            printf("datareader error\n");
            return;
        }
        rtn = _reader->take(data_values, sample_infos,
            MAX_INT32_VALUE, ANY_SAMPLE_STATE, ANY_VIEW_STATE, ANY_INSTANCE_STATE);
        if (DDS_RETCODE_OK != rtn)
        {
            return;
        }
        for (int i = 0; i < sample_infos.length(); i++)
        {
            CharSeqTypePrintData(&data_values[i]);
        }
        _reader->return_loan(data_values, sample_infos);
        return;
    }
};
```

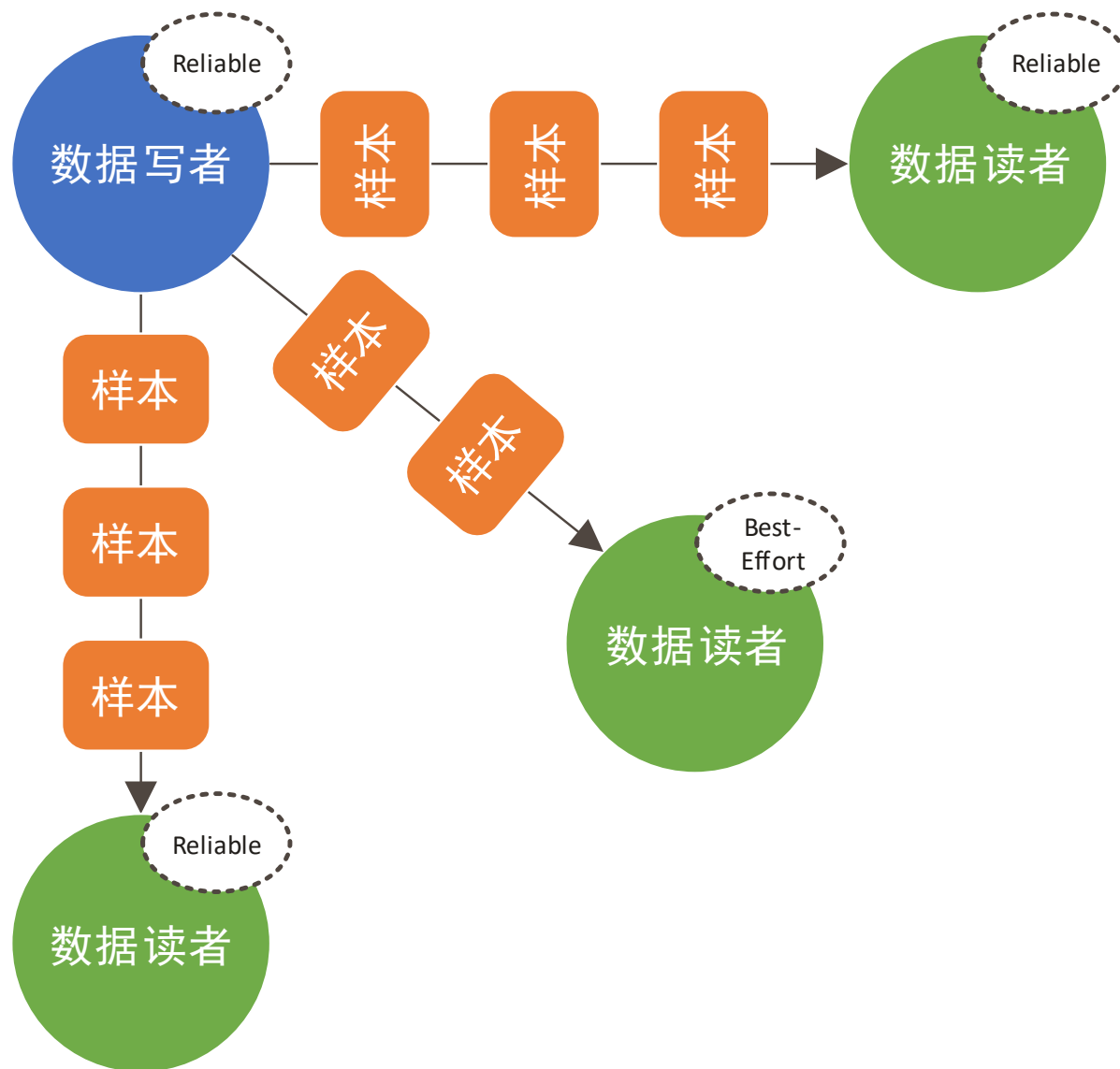


- BEST-EFFORT

- 低延时；
- 适合传输周期性等数据量较大且可靠性不高的数据；

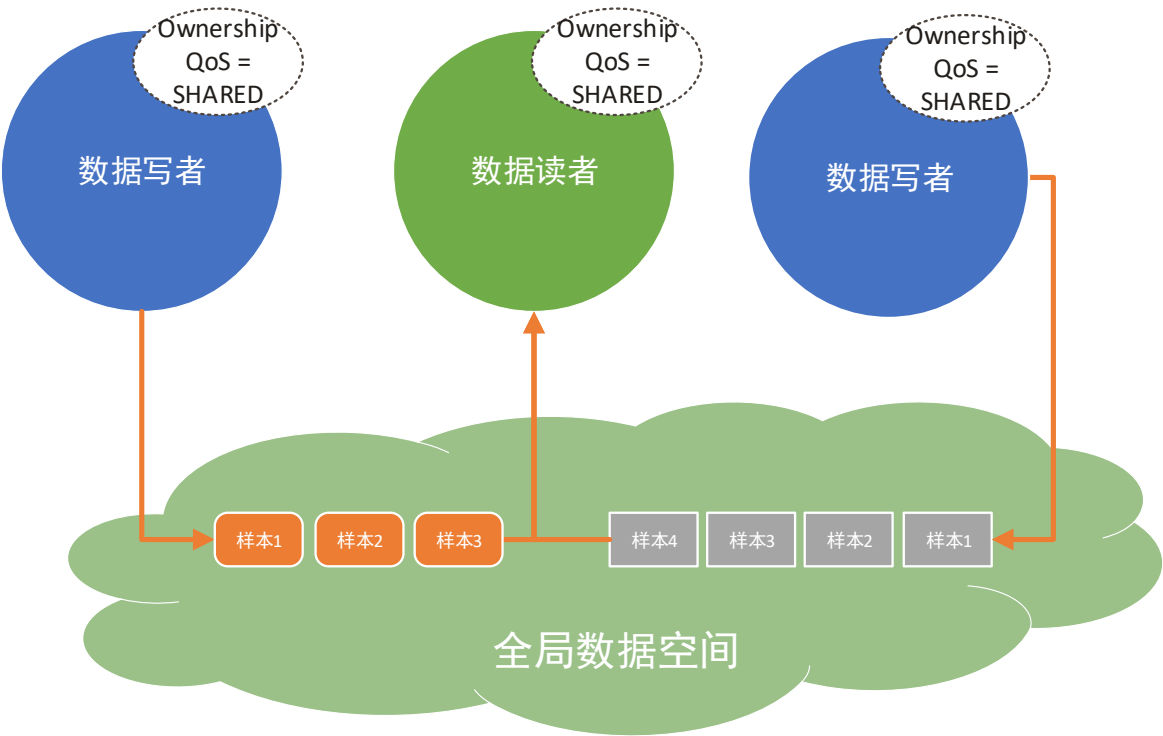
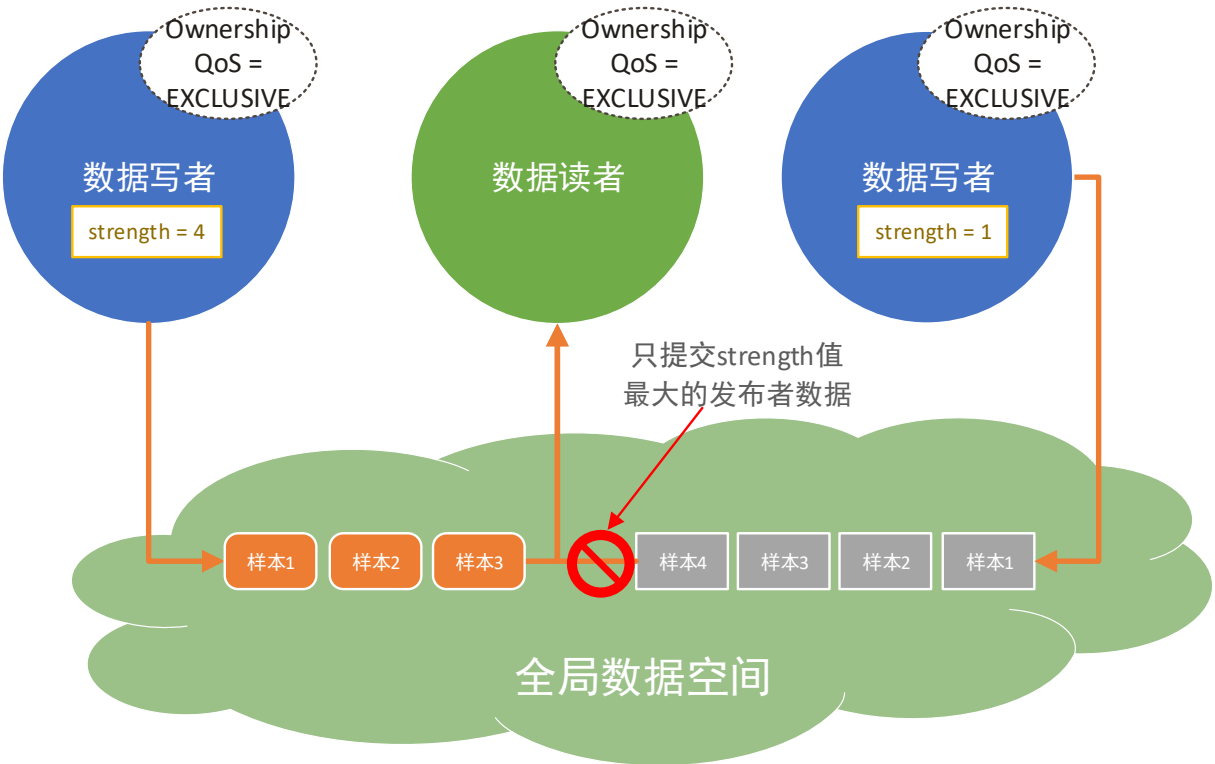
- RELIABLE

- 保证订阅端保序接收发布端发布的所有报文；
- 适合传输控制指令等可靠性需求高的数据；



- SHARED

- 订阅端可以收到所有匹配发布端的数据；



- EXCLUSIVE

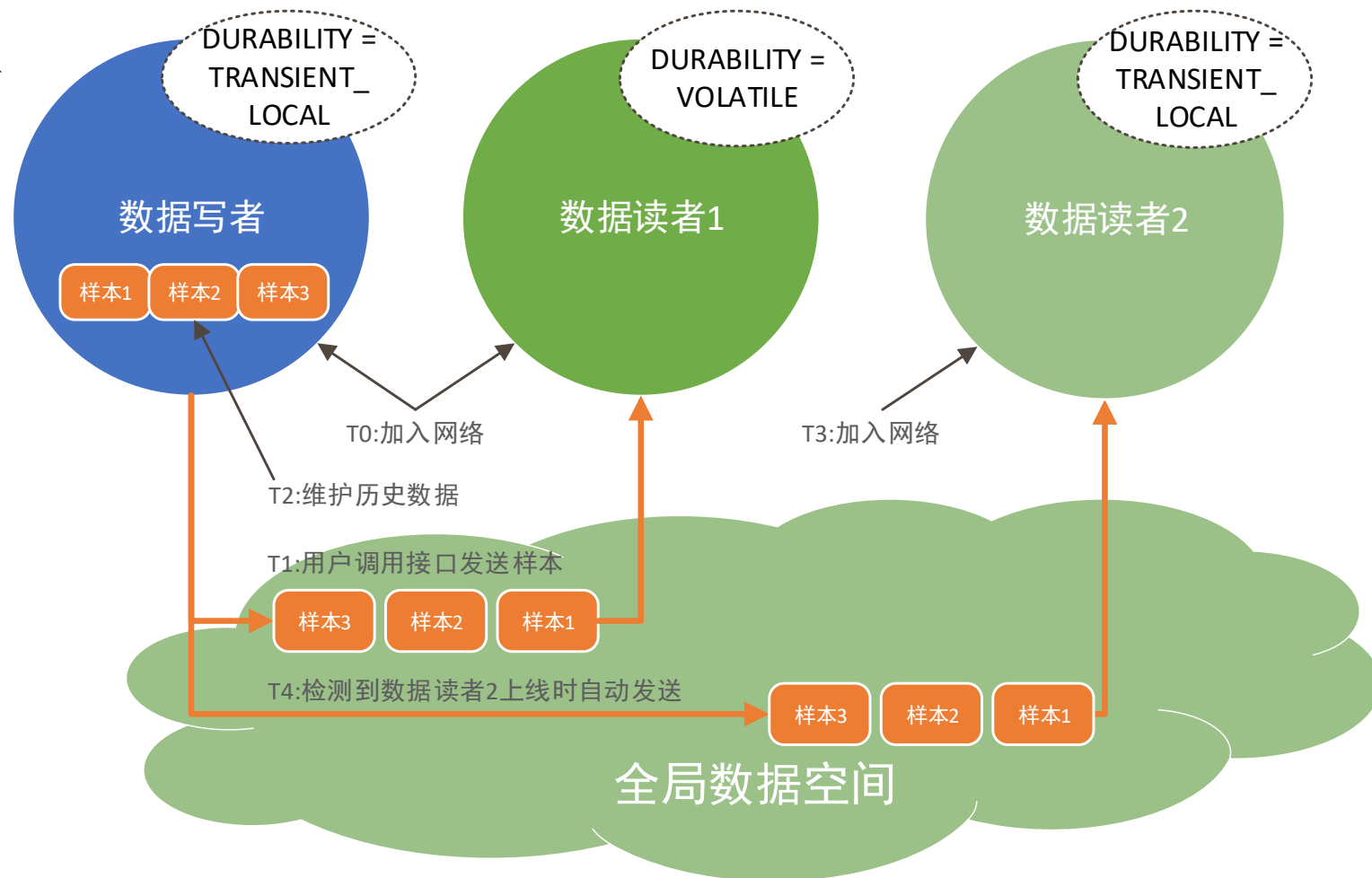
- 可用于实现系统热备份；
- 订阅端只接收当前系统中所有匹配发布端中权值（Strength）最大的发布端发送的数据；

- TRANSIENT_LOCAL

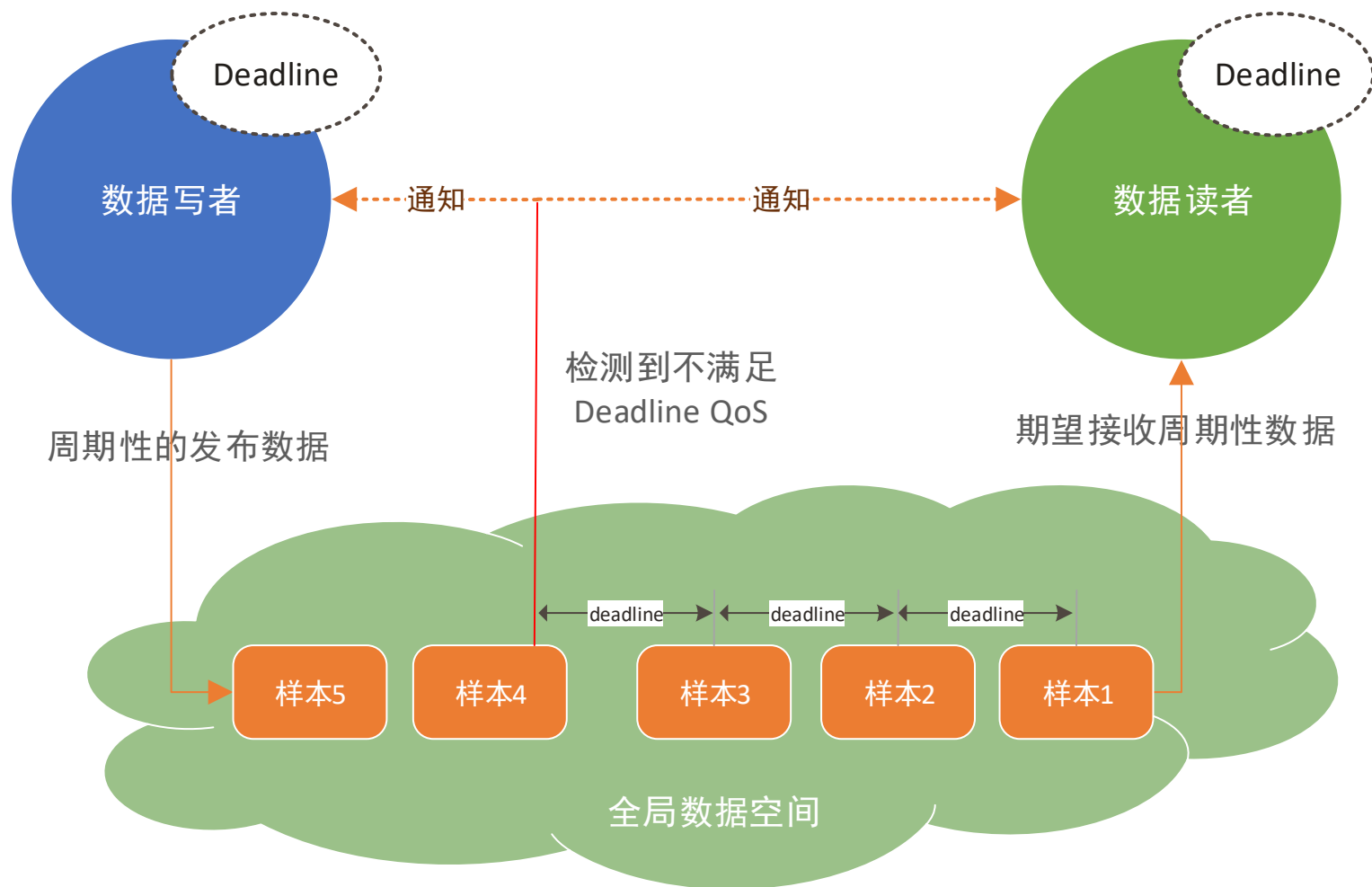
- 保存一定数量的历史数据，并发送给“将来”新创建的订阅者；

- VOLATILE

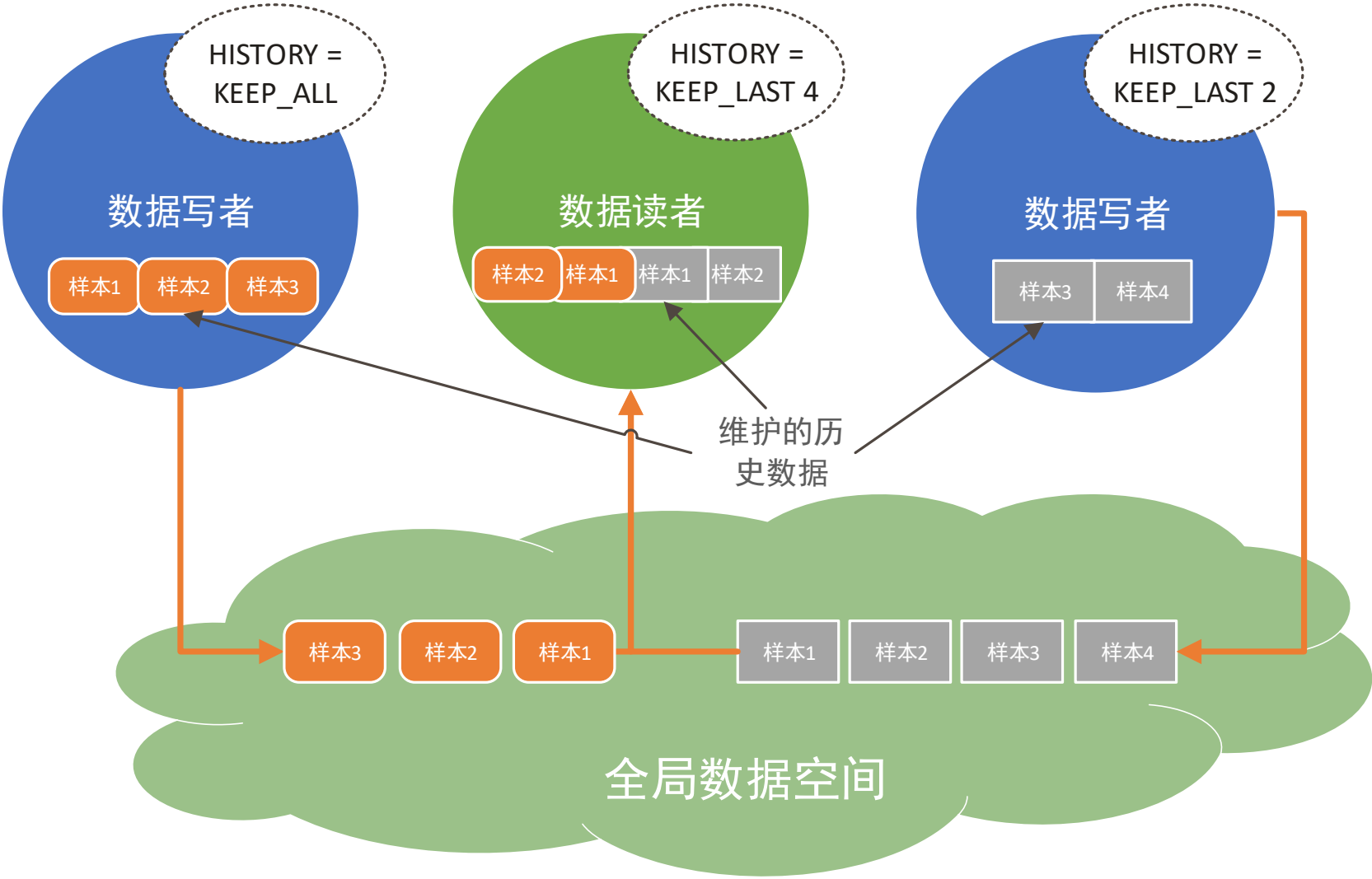
- 不发送历史数据



- 可用于检测发送端是否被周期性调用，检查程序或者运行时是否和设计一致；
- 可用于检测接收端是否在指定的时间内收到数据，避免过期数据；



- 数据写者端表示可靠队列；
- 数据读者端表示接收到的历史数据；
- KEEP_ALL表示保存所有的样本；
- KEEP_LAST表示保留最新的DEPTH个；



- 函数签名
 - 函数名称
 - 函数参数
 - 函数返回值
 - 函数的功能
 - 注意事项
 - 相关接口

```
DomainParticipant * DDS::DomainParticipantFactory::create_participant(const DomainId_t & domainId,  
                                                                       const DomainParticipantQos & qoslist,  
                                                                       DomainParticipantListener * a_listener,  
                                                                       const StatusKindMask & mask  
                                                                       )
```

创建一个新的域参与者实体，并设置QoS以及监听器，域参与者的创建表明应用程序打算加入`domainId`指定的域中进行通信。

参数

domainId 表明需要加入的域号，取值范围为[0-232]。

qoslist 表示为该域参与者设置的QoS，`::DOMAINPARTICIPANT_QOS_DEFAULT` 表明使用域参与者工厂中保存的默认的QoS。

[in,out] **a_listener** 为该域参与者设置的监听器。

mask 设置应用程序感兴趣的状态，只有应用程序感兴趣的状态发生变化时，才会通知应用程序。

返回

创建成功指向创建成功的域参与者实体对象，否则返回NULL，失败的原因可能为：

- 分配空间失败或者初始化资源失败，具体的错误信息参见日志；
- `qoslist` 含有无效值或者含有不一致的QoS。

接口	说明
DDS::DDSIF::Init()	初始化ZRDDS，并获取域参与者工厂。
DDS::DDSIF::CreateDP()	创建域参与者，并配置域、QoS、监听器。
DDS::DDSIF::PubTopic()	创建与指定主题名称、类型名称关联的数据写者，并设置QoS、监听器。
DDS::DDSIF::UnPubTopic()	取消发布，通过指定精确的数据写者指针。
DDS::DDSIF::UnPubTopicWTopicName()	取消发布，通过指定域以及主题名称。
DDS::DDSIF::SubTopic()	创建与指定主题名称、类型名称关联的数据读者，并设置QoS、监听器。
DDS::DDSIF::UnSubTopic()	取消订阅，通过指定精确的数据读者指针。
DDS::DDSIF::UnSubTopicWTopicName()	取消发布，通过指定域以及主题名称。
DDS::ZRDDSDataWriter::write()	向DDS系统发布数据，由于DDS的强类型安全，应转化为特定类型的数据写者调用write函数，如零拷贝的数据写者，应调用 ZeroCopyBytesDataWriter::write() 接口，对于非零拷贝（缓冲区类型）应调用 BytesDataWriter::write() 。
DDS::SimpleDataReaderListener	默认简单的数据到达监听器，用户继承该监听器并在 DDS::SimpleDataReaderListener< T, TSeq, TDataReader >::on_process_sample 中处理到达的样本。
DDS::DDSIF::Finalize()	回收DDS中间件所有资源。

```
int ZRDDS DemoModulesInitial()
{
    // 获取入口, 并指明QoS配置
    DomainParticipantFactory* factory = DDSIF::Init(NULL, "non_rio");
    if (factory == NULL) { printf("DDSIF::Initialize failed.\n"); return -1; }
    // 初始化使用udp的域参与者
    DomainParticipant* udpDp = DDSIF::CreateDP(150, "udp_dp");
    if (NULL == udpDp) { printf("DDSIF::CreateDP failed.\n"); return -2; }
    // 创建非零拷贝 (缓冲区类型) 数据读者, 并设置监听器
    g_zrddsModule.m_listener = new Mylistener();
    DataReader* airTrackDr = DDSIF::SubTopic(
        udpDp, "AirTrack", BytesTypeSupport::get_instance(), "best_effort_reader", g_zrddsModule.m_listener);
    if (NULL == airTrackDr) { printf("DDSIF::SubscribeTopic failed.\n"); return -4; }
    return 0;
}
```

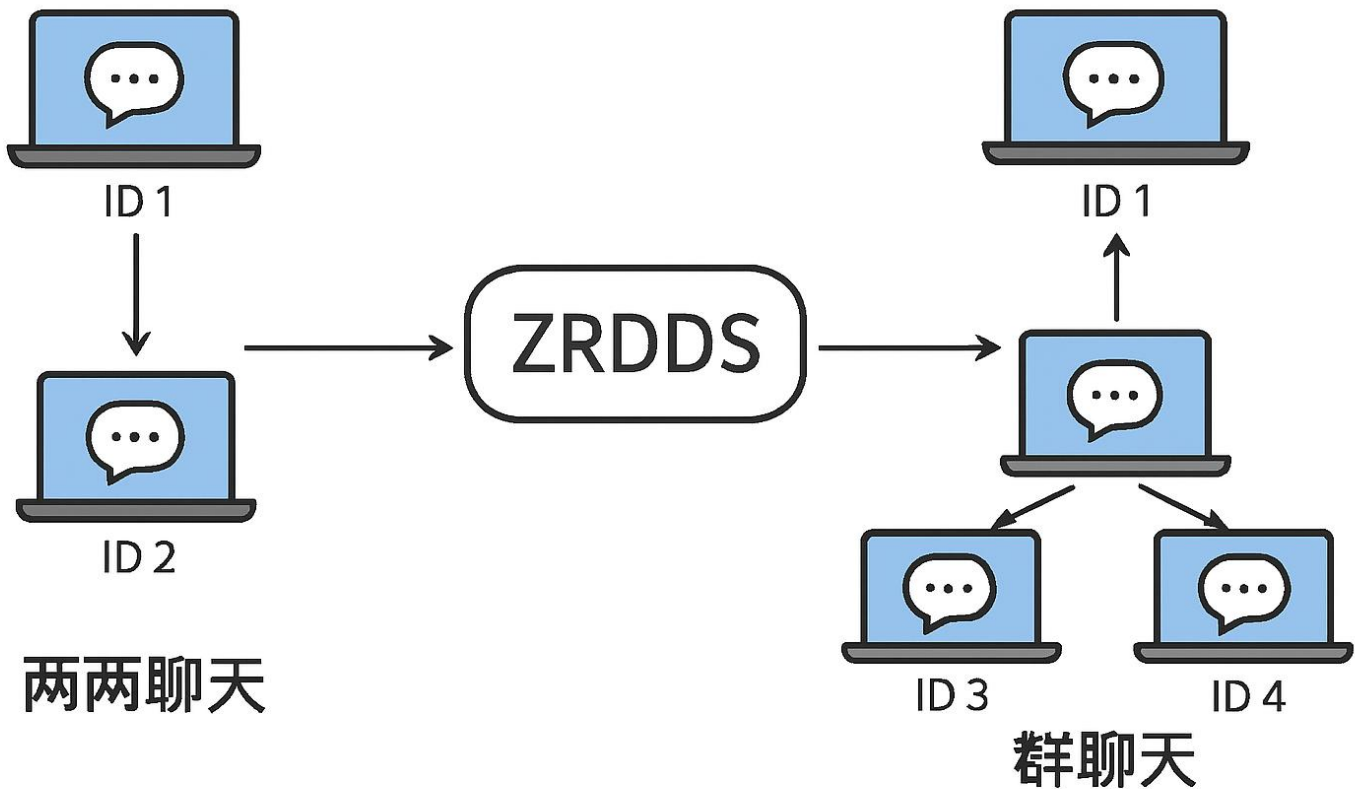
```
ReturnCode_t retCode = DDSIF::BytesWrite(150, "AirTrack", buffer, length);
if (retCode != RETCODE_OK)
{
    printf("send failed(%d).\n", retCode);
}
```

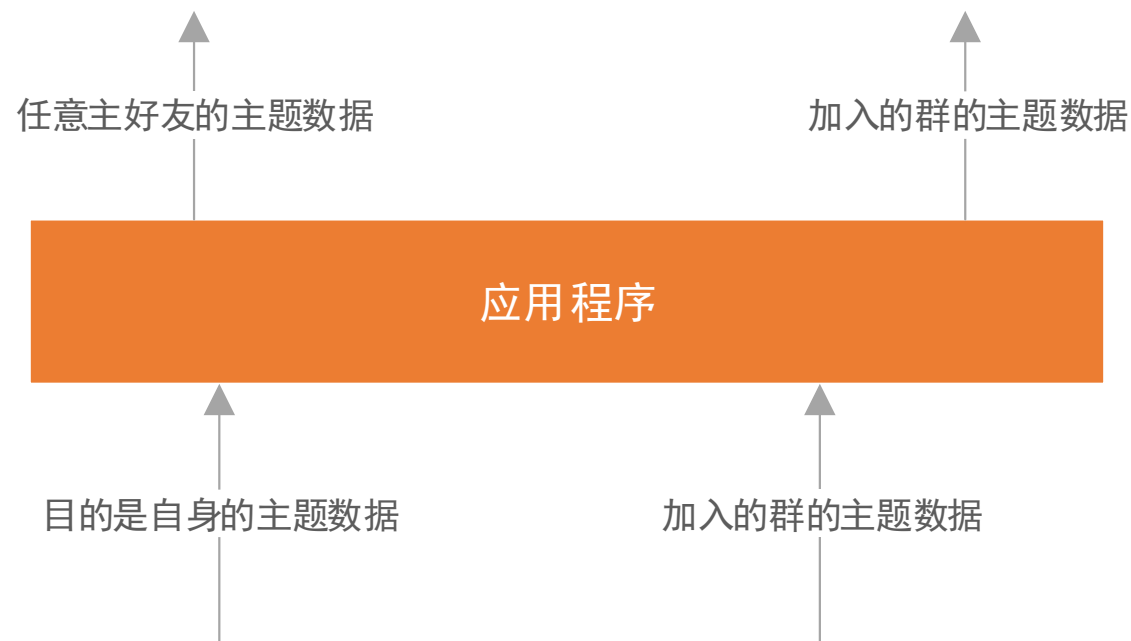
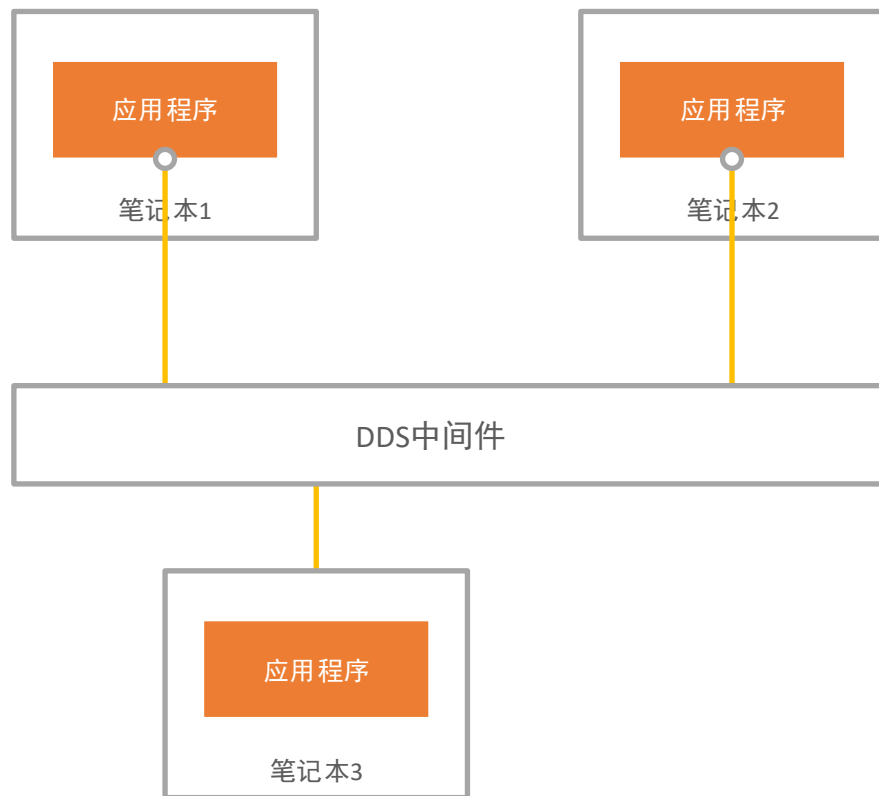
```
/* 定义非零拷贝 (缓冲区数据类型) 回调接口 */
class Mylistener : public DDS::DataReaderListener
{
public:
    void on_data_arrived(DataReader* reader, void* curSample, const SampleInfo& curInfo)
    {
        Bytes* sample = (Bytes*)curSample;
        printf("received data(%d) length(%u) from topic(%s)\n", *(int*)buffer, length, topicName);
    }
};
```

DDS应用开发实例

- 使用ZRDDS通信中间件实现即时聊天系统
 - 两两聊天：发送数据给指定的ID
 - 群聊天：群成员都能收到发送的数据

使用ZRDD通信中间件实现即时聊天系统





- 主题设计
 - 主题名称：好友ID或者群ID
 - 数据类型：字符串或者二进制缓冲区
- QoS设计
 - 不能丢包->可靠性（RELIABLE）
 - 上线后能收到别人发送的历史数据->持久化（TRANSIT_LOCAL）
 - 下线检测->（自动发现、Liveliness）

- 发布订阅关系
 - 框图
 - 创建订阅发布时机
 - 自身主题订阅（启动时）
 - 好友主题发布（添加好友时）
 - 群消息订阅发布（加入群时）
- 数据发送时机
 - 按需（用户触发）
 - 内容（元数据+用户提供）
- 数据处理逻辑
 - 异步回调方式
 - 输出数据

- 集成开发环境（Visual Studio/Visual Studio Code/Eclipse/JetBrains
- 编码规范
- 版本管理（Git）

Leaderboard

Token usage across models

Top this week		
1.	Anthropic: Claude Sonnet 4 > Claude Sonnet 4 significantly enhances the capabilities of its predec...	161B tokens ↑ 2%
2.	Anthropic: Claude 3.7 Sonnet > Claude 3.7 Sonnet is an advanced large language model with improv...	90.8B tokens ↓ 16%
3.	Google: Gemini 2.5 Pro Preview 06-05 > Gemini 2.5 Pro is Google's state-of-the-art AI model designed for ad...	63B tokens new
4.	Google: Gemini 2.5 Flash Preview 05-20 > Gemini 2.5 Flash May 20th Checkpoint is Google's state-of-the-art ...	55.1B tokens ↓ 12%
5.	Google: Gemini 2.0 Flash > Gemini Flash 2.0 offers a significantly faster time to first token (TTFT)...	29B tokens ↑ 305%
6.	Google: Gemini 2.5 Pro Preview 05-06 > Gemini 2.5 Pro is Google's state-of-the-art AI model designed for ad...	27.3B tokens ↓ 62%
7.	OpenAI: GPT-4o-mini > GPT-4o mini is OpenAI's newest model after [GPT-4 Omni] (/models/...	16.4B tokens ↓ 12%
8.	Google: Gemini 2.5 Flash Preview 04-17 > Gemini 2.5 Flash is Google's state-of-the-art workhorse model, spe...	15.7B tokens ↓ 30%
9.	OpenAI: GPT-4.1 > GPT-4.1 is a flagship large language model optimized for advanced i...	15.4B tokens ↑ 5%
10.	Anthropic: Claude Opus 4 > Claude Opus 4 is benchmarked as the world's best coding model, at t...	11.6B tokens ↑ 21%

Cursor

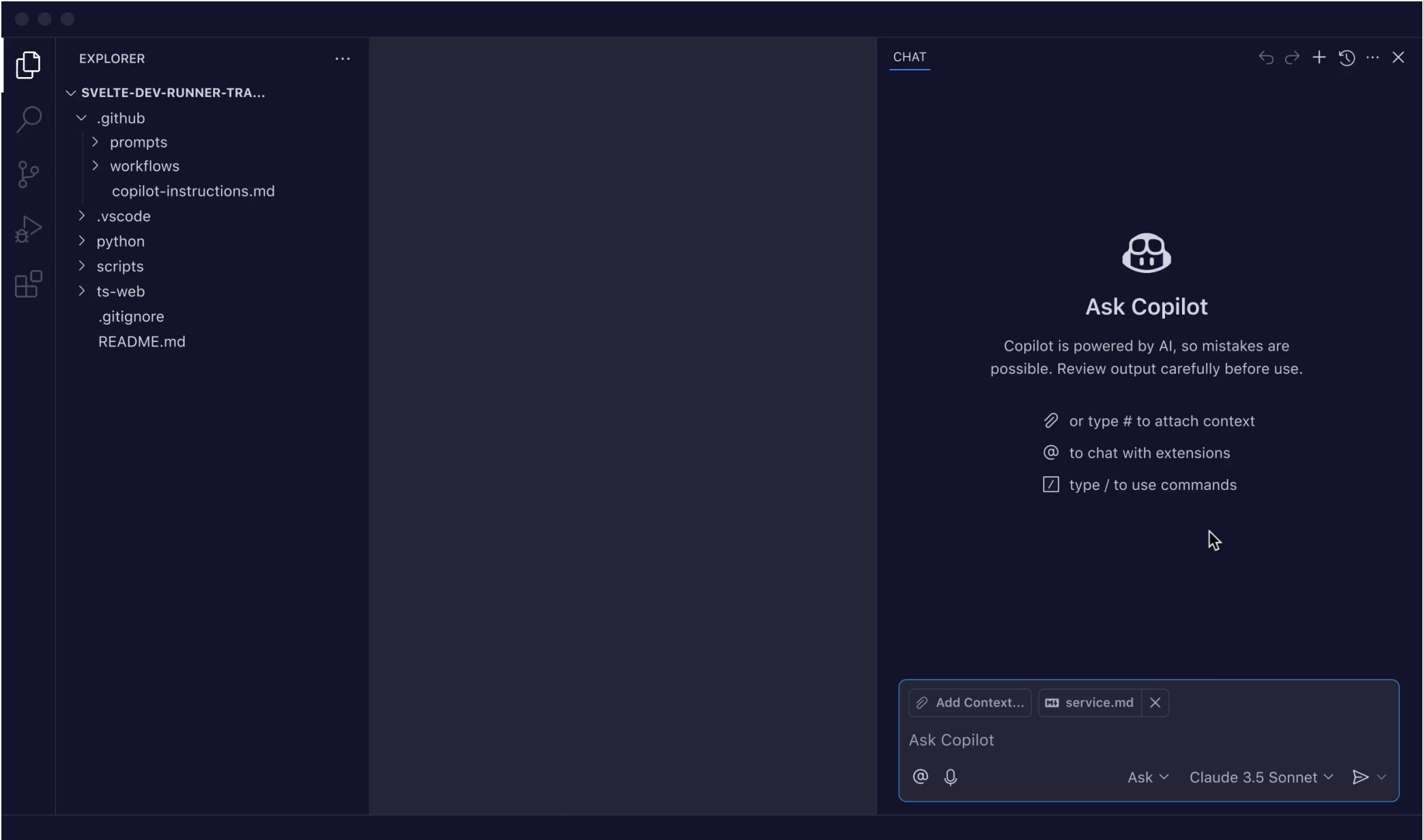
- The Best way to code with AI。

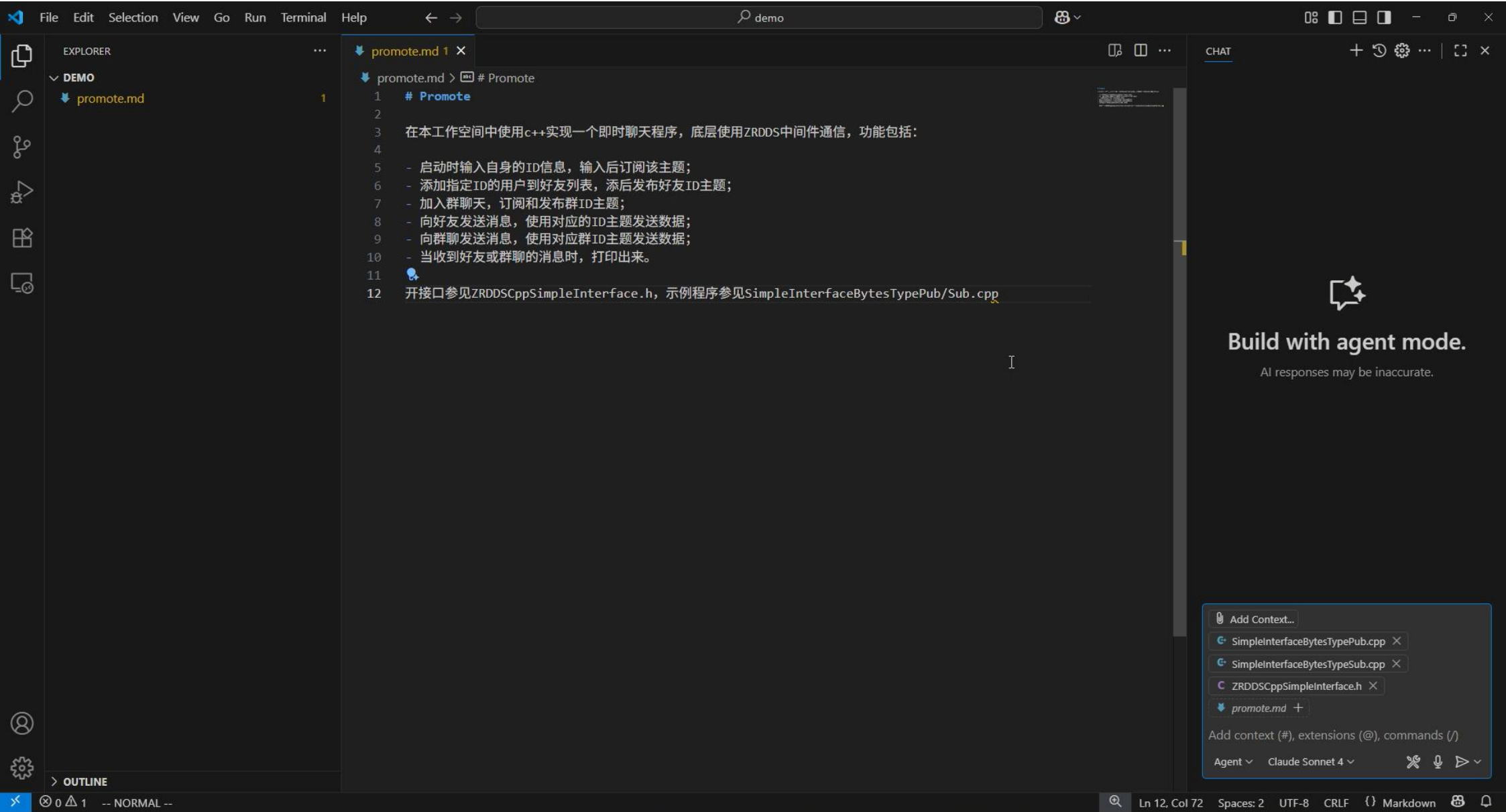
Github Copilot

- VSCode完美

Trae

- 国内领先





文件(F) 编辑(E) 视图(V) Git(G) 项目(P) 生成(B) 调试(D) 测试(S) 分析(N) 工具(T) 扩展(X) 窗口(W) 帮助(H) 搜索 本地 Windows 调试器

main.cpp ChatSystem.cpp

ChatSystem (全局范围) printWelcome()

```
1 #include "ChatSystem.h"
2 #include <iostream>
3 #include <string>
4 #include <sstream>
5 #include <thread>
6 #include <chrono>
7
8 void printHelp() {
9     std::cout << "\n=== Chat System Commands ===" << std::endl;
10    std::cout << "add <friendId> <friendId> Add a friend" << std::endl;
11    std::cout << "join <groupId> <friendId> Join a group chat" << std::endl;
12    std::cout << "msg <friendId> <text> Send private message to friend" << std::endl;
13    std::cout << "group <groupId> <text> Send message to group" << std::endl;
14    std::cout << "friends Show friends list" << std::endl;
15    std::cout << "groups Show groups list" << std::endl;
16    std::cout << "help Show this help" << std::endl;
17    std::cout << "quit Exit the program" << std::endl;
18    std::cout << "=====" << std::endl;
19 }
20
21 void printWelcome() {
22     std::cout << "=====" << std::endl;
23     std::cout << "Welcome to ZRDDS Chat System!" << std::endl;
24     std::cout << "=====" << std::endl;
25     std::cout << "This chat system uses ZRDDS middleware for communication." << std::endl;
26 }
```

81% 未找到相关问题 行: 25 字符: 91 空格 CRLF

输出 显示输出来源(S): 生成

就绪

解决方案资源管理器

搜索解决方案资源管理器(Ctrl+;) 解决方案 'ZRDDSChatSystem' (4 个项目, 共 4 个)

ALL_BUILD

ChatSystem

引用

外部依赖项

Source Files

ChatSystem.cpp

main.cpp

CMakeLists.txt

INSTALL

ZERO_CHECK

GitHub Copilot 聊天 解决方案资源管理器 Git 更改 类视图

添加到源代码管理 选择仓库

AI生成——根据文档修改

omote.md

ChatSystem.h

ChatSystem.cpp

main.cpp 2

README.md

ZRDDS_QOS_PROFILES.xml

CMakeLists.txt

...

CMakeLists.txt

```
1  # ZRDDS聊天系统编译配置
2  cmake_minimum_required(VERSION 3.10)
3  project(ZRDDSChatSystem)
4
5  # 设置C++标准
6  set(CMAKE_CXX_STANDARD 11)
7  set(CMAKE_CXX_STANDARD_REQUIRED ON)
8
9  # 如果是Debug模式，添加调试信息
10 if(CMAKE_BUILD_TYPE STREQUAL "Debug")
11     set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -g -O0")
12 endif()
13
14 # ZRDDS库路径设置（请根据实际安装路径修改）
15 set(ZRDDS_ROOT "D:/ZRDDS/ZRDDS-2.4.4")
16 set(ZRDDS_INCLUDE_DIR "${ZRDDS_ROOT}/include/CPlusPlusInterface")
17 set(ZRDDS_LIB_DIR "${ZRDDS_ROOT}/lib")
18
19 # 包含目录
20 include_directories(${ZRDDS_INCLUDE_DIR})
21 include_directories(${CMAKE_CURRENT_SOURCE_DIR})
22
23 # 查找ZRDDS库文件
24 find_library(ZRDDS_LIBRARY
25     NAMES zrdds ZRDDSCpp libzrdds libZRDDSCpp
```


AI生成——根据文档修改



文件(F) 编辑(E) 视图(V) Git(G) 项目(P) 生成(B) 调试(D) 测试(S) 分析(N) 工具(T) 扩展(X) 窗口(W) 帮助(H)

搜索 ZRDDSChatSystem

Debug x64 本地 Windows 调试器

ChatSystem.h ZRDDSCppSim...Interface.h main.cpp ChatSystem.cpp

ChatSystem

```
387     ...}
388 }
389
390 void ChatSystem::showFriends() const {
391     safeOutput("\n=== Friends List ===");
392     if (friends.empty()) {
393         safeOutput("No friends yet. Use 'add <friendId>' to add friends.");
394     } else {
395         for (size_t i = 0; i < friends.size(); ++i) {
396             safeOutput(std::to_string(i + 1) + ". " + friends[i]);
397         }
398     }
399 }
400
401 void ChatSystem::showGroups() const {
402     safeOutput("\n=== Groups List ===");
403     if (groups.empty()) {
404         safeOutput("No groups joined yet. Use 'join <groupId>' to join groups.");
405     } else {
406         for (size_t i = 0; i < groups.size(); ++i) {
407             safeOutput(std::to_string(i + 1) + ". " + groups[i]);
408         }
409     }
410 }
411 }
```

81% 6 0

行: 393 字符: 1 空格 CRLF

问题详细信息

C2662 "void ChatSystem::safeOutput(const std::string &): 不能将"this"指针从"const ChatSystem"转换为"ChatSystem &" ChatSystem.cpp (行 393)

转换丢失限定符 ChatSystem.cpp (行 393)

参见"ChatSystem::safeOutput"的声明 ChatSystem.h (行 96)

尝试匹配参数列表"(const char [53])"时 ChatSystem.cpp (行 393)

解决方案资源管理器

搜索解决方案资源管理器(Ctrl+;)

解决方案 'ZRDDSChatSystem' (4 个项目, 共 4 个)

ALL_BUILD

ChatSystem

引用

外部依赖项

Source Files

ChatSystem.cpp

main.cpp

CMakeLists.txt

INSTALL

ZERO_CHECK

GitHub Copilot 聊天

解决方案资源管理器

Git 更改

类视图

就绪

添加到源代码管理 选择仓库

AI生成——根据文档修改



文件(F) 编辑(E) 视图(V) Git(G) 项目(P) 生成(B) 调试(D) 测试(S) 分析(N) 工具(T) 扩展(X) 窗口(W) 帮助(H) 搜索 本地 Windows 调试器

ChatSystem.h ZRDDSCppSim...Interface.h main.cpp ChatSystem.cpp

ChatSystem

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

bool initialize(const std::string& userId);

...

// 添加好友

bool addFriend(const std::string& friendId);

...

// 加入群聊

bool joinGroup(const std::string& groupId);

...

// 发送私聊消息

bool sendPrivateMessage(const std::string& friendId, const std::string& message);

...

// 发送群聊消息

bool sendGroupMessage(const std::string& groupId, const std::string& message);

...

// 处理接收到的消息

void handleReceivedMessage(const std::string& data);

...

// 显示好友列表

void showFriends() const;

...

// 显示群组列表

void showGroups() const;

...

// 安全的控制台输出

void safeOutput(const std::string& message);

解决方案资源管理器

搜索解决方案资源管理器(Ctrl+)

解决方案 'ZRDDSCppSystem' (4 个项目, 共 4 个)

ALL_BUILD

ChatSystem

引用

外部依赖项

Source Files

ChatSystem.cpp

main.cpp

CMakeLists.txt

INSTALL

ZERO_CHECK

81 % 未找到相关问题

行: 96 字符: 48 空格 CRLF

输出

显示输出来源(S): 生成

1>ZRDDSCppzd_VS2015.lib(ZRBuiltInTypeCPlusPlusSequence.obj) : 找到 MSIL .netmodule 或使用 /GL 编译的模块; 正在使用 /LTCG 重新启动

1>LINK : warning LNK4075: 忽略 "/INCREMENTAL" (由于 "/LTCG" 规范)

1>正在创建库 C:/Users/zhenrong/Desktop/春华行动/demo/build/Debug/ChatSystem.lib 和对象 C:/Users/zhenrong/Desktop/春华行动/demo/t

1>正在生成代码

1>已完成代码的生成

1>ChatSystem.vcxproj -> C:\Users\zhenrong\Desktop\春华行动\demo\build\Debug\ChatSystem.exe

1>已完成生成项目 "ChatSystem.vcxproj" 的操作。

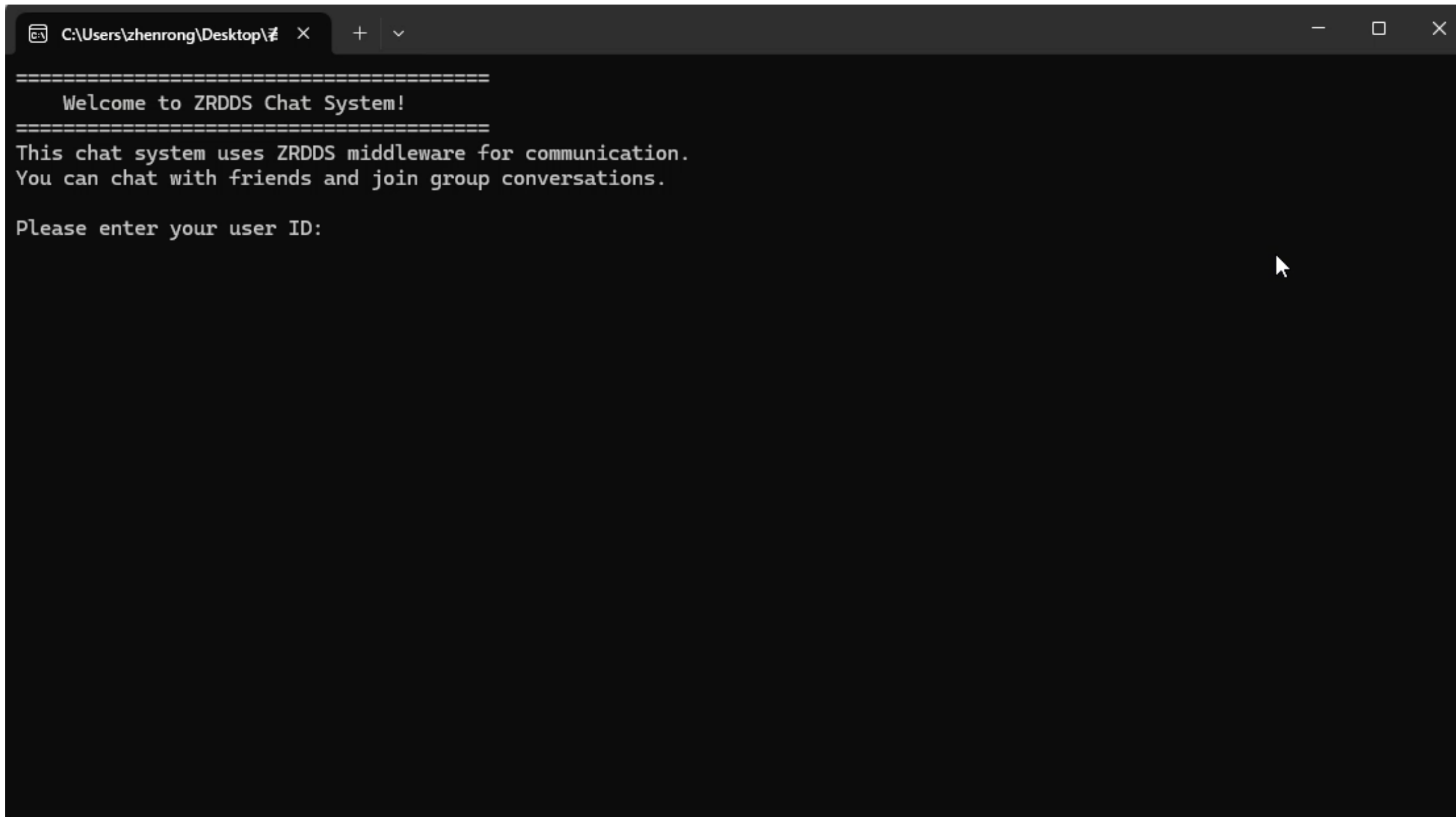
===== 生成: 1 成功, 0 失败, 1 最新, 0 已跳过 =====

===== 生成 于 9:10 完成, 耗时 04.152 秒 =====

已保存的项

GitHub Copilot 聊天 解决方案资源管理器 Git 更改 类视图

添加到源代码管理 选择仓库



```
C:\Users\zhenrong\Desktop\臻融科技 > .\ZRDDS.exe

=====
Welcome to ZRDDS Chat System!
=====
This chat system uses ZRDDS middleware for communication.
You can chat with friends and join group conversations.

Please enter your user ID:
```


首页

相关页面

模块

类

示例

▼ ZRDDS

▶ 臻融数据分发服务 (ZRDDSV2.4.0) 在线

ZRDDS下载

▶ ZRDDS常见问题

ZRDDS版本记录

ZRDDS调试日志信息表

▶ ZRDDS多种接口风格及开发流程

ZRDDS日志调试使用说明

▶ XML配置实体QoS

TCP多核传输 (>=v2.4.0)

大包零拷贝 (>=v2.4.0)

▶ 模块

▶ 类

▶ 示例

XML示例

1 <zrdds_qos>

2 <qos_library name="1"

3 <qos_profile nam

4 </qos_profile>

5 </qos_library>

6 <qos_library name="1"

7 <qos_profile nam

8 <participant

9 <entity_

10 <aut

11 </entity

12 </participan

13 <participant

14 <usertra

C ChatSystem.h M

G+ ChatSystem.cpp M

G+ main.cpp 2, M

ZRDDS_QOS_PROFILES.xml

1 <?xml version="1.0" encoding="utf-8"?>

2 <dds xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

3 xsi:noNamespaceSchemaLocation="http://community.rti

4 version="6.0.0">

5

6 <!-- 默认库配置 -->

7 <qos_library name="default_lib">

8

9 <!-- 默认配置文件 -->

10 <qos_profile name="default_profile">

11

12 <!-- 域参与者QoS配置 -->

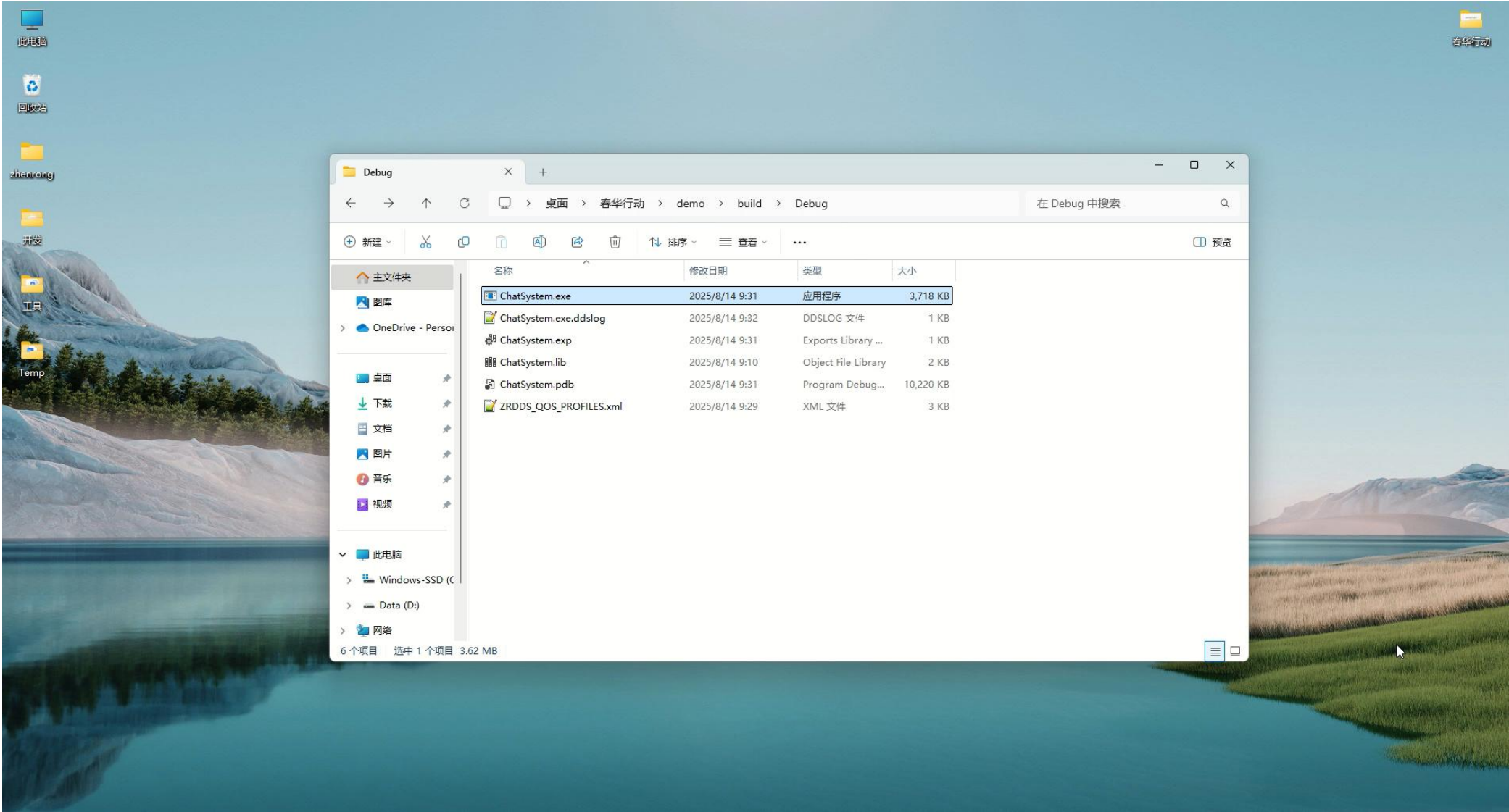
13 <domain_participant_qos>

14 <participant_name>

15 <name>ChatSystemParticipant</name>

16 </participant_name>

17 </domain participant qos>



ZRDDS管理监控器

文件 窗口 关于

系统逻辑视图

type filter text

域/主题

系统逻辑视图

150号域

hzy

wy

seu

N/A

N/A

N/A

N/A

N/A

N/A

系统物理视图

type filter text

系统物理视图

150号域 : wy主题详细信息

150号域 : seu主题详细信息

DW: 192.168.91.1: ChatSystem.exe: 1408

DW: 192.168.91.1: ChatSystem.exe: 19764

150号域 : seu主题详细信息

DR: 192.168.91.1: ChatSystem.exe: 1408

DR: 192.168.91.1: ChatSystem.exe: 19764

QoS信息

QoS名称

QoS值

hex_value

empty

durability

kind

VOLATILE_DURAB

durability_service

serice_cleanup_delay

{0, 0}

history_kind

KEEP_LAST_HISTC

history_depth

1

max_samples

-1

max_instances

-1

max_sample_per_instance

-1

deadline

period

{2147483647

latency_budget

duration

{0, 0}

liveliness

kind

AUTOMATIC_LIVEI

duration

{2147483647

reliability

kind

RELIABLE_RELIABI

lifespan

duration

{2147483647

DDS数据类型

报文统计信息

struct DDS_Bytes

{

sequence<unsigned char, 2147483647> value;///

ID 0

};///

@Extensibility DDS_EXTENSIBLE_EXTENSIBILITY

Console

匹配分析

通信质量分析视图

样本历史数据表视图

ZRDDS管理监控器日志

2025-08-25 14:30:15 [INFO] 发现域参与者[150,0xc0a85b0100004d340000000000000001c1]上线

2025-08-25 14:30:15 [INFO] 发现数据读者[hzy,DDS_Bytes,0xc0a85b0100004d340000000000000000a0000004]上线

2025-08-25 14:30:30 [INFO] 发现域参与者[150,0xc0a85b01000005800000000000000001c1]上线

2025-08-25 14:30:30 [INFO] 发现数据读者[wy,DDS_Bytes,0xc0a85b01000005800000000000000000a0000004]上线

2025-08-25 14:30:33 [INFO] 发现数据读者[hzy,DDS_Bytes,0xc0a85b01000005800000000000000000d0000003]上线

2025-08-25 14:30:33 [INFO] 发现数据读者[wy,DDS_Bytes,0xc0a85b0100004d340000000000000000d0000003]上线

2025-08-25 14:31:31 [INFO] 发现数据读者[seu,DDS_Bytes,0xc0a85b01000005800000000000000000f0000003]上线

2025-08-25 14:31:31 [INFO] 发现数据读者[seu,DDS_Bytes,0xc0a85b01000005800000000000000000f0000004]上线

2025-08-25 14:32:01 [INFO] 发现数据读者[seu,DDS_Bytes,0xc0a85b0100004d340000000000000000f0000003]上线

2025-08-25 14:32:01 [INFO] 发现数据读者[seu,DDS_Bytes,0xc0a85b0100004d340000000000000000f0000004]上线

南京臻融科技有限公司

地址：南京市江宁区将军大道迎翠路7号

电话：025-52106986

官网：<http://www.zrtechnology.com>

