臻融数据分发服务DDS性能测试工具需求说明文档

版本：V2.4.4

日期：2025.8

编写人：段嘉文 肖飞鸿 杨熙锐 王佳琪 邹欣纯

# 一、引言

## 1.1文档目的

本文档旨在定义和阐述DDS中间件性能测试工具的需求。其首要目的是提供清晰、一致的功能性与非功能性需求说明，确保所有相关方对项目的目标、范围和交付物有共同的理解。本文档将作为后续设计、开发、测试和验收阶段的基准依据。

## 1.2项目背景与问题陈述

数据分发服务（Data Distribution Service, DDS）是一种由对象管理组织发布的以数据为中心的通信中间协议和API标准。它具备高效性、实时性、可靠性和灵活的Qos（服务质量）策略，现已成为分布式实时系统，是诸多关键领域的首选通讯基础设施。

随着DDS在这些对性能有严格要求的领域的广泛运用，如何客观、准确、全面地评估和比较不同DDS中间件实现的性能表现，成为了系统架构师、开发者和决策者面临的核心挑战。然而，在实际应用中，由于不同应用场景对吞吐量、时延、抖动、系统资源占用以及QoS有不同要求，如何科学、准确地评估DDS中间件的性能成为系统设计和优化的重要前提。现有的一些性能测试工具功能有限，难以灵活配置测试参数，无法满足多样化的性能评估需求。

因此，项目旨在开发一款可配置的DDS中间件性能测试工具，通过配置化的设计实现对DDS系统的灵活测试和多维度性能分析。该工具将支持通过JSON配置文件灵活设定域号、主题名、域参与者、数据写者、数据读者QoS策略、数据长度、发送周期这些参数，并能够按照json文件内容完成DDS实体的创建、性能数据的采集与统计分析，最终以标准化格式输出测试结果，方便进行性能对比与优化。

本项目的目标是提供一套功能完善、可扩展、易用性强的DDS中间件性能测试工具，帮助开发人员和系统工程师快速评估DDS系统的性能瓶颈，为上层应用优化和架构设计提供数据支撑，从而提升分布式系统的稳定性与可靠性。

# 二、目标与范围

## 2.1目标

### 2.1.1.性能量化

通过自动化测试，精确测量ZRDDS的以下核心性能指标：

·端到端时延 (End-to-End Latency): 测量从发布者调用DataWriter::write()到订阅者在on\_data\_available()回调中接收到数据的时间差。根据手册第12章的通信机制，设定端到端时延的性能基线为典型场景下的合理值，具体值需通过初始测试确定。

·吞吐量 (Throughput): 测量单位时间内成功传输的消息数量（msg/s）或数据量（Mbps）。目标参考手册第12章和第10.2节，重点验证Batch等优化策略对小数据包吞吐量的提升效果，具体数值需基于测试数据确定。

·可靠性与丢包率 (Reliability & Packet Loss): 在DDS\_RELIABLE\_RELIABILITY\_QOS模式下，验证在正常网络和系统资源充足的条件下，消息丢失率接近0%，如果出现丢失，需记录并分析原因；在DDS\_BEST\_EFFORT\_RELIABILITY\_QOS模式下，记录并报告在不同网络负载下的实际丢包情况，以评估尽力而为传输的性能边界。

·系统资源占用：监控并记录测试过程中，发布者和订阅者进程的CPU利用率、内存占用等系统资源消耗。

### 2.1.2场景覆盖

全面覆盖ZRDDS支持的多种通信模式和部署场景：

·拓扑结构：支持单发布者-单订阅者（1:1）、多发布者-单订阅者（N:1）、单发布者-多订阅者（1:N）及多发布者-多订阅者（N:N）的测试。

·数据模型: 支持有Key和无Key的数据类型，以验证实例管理对性能的影响（参考手册第2.7节）。

·跨平台通信: 验证Windows与Linux平台间的互操作性和性能表现（参考手册第11章）。

·传输协议: 支持UDP和TCP的性能对比测试。（注：RapidIO通信支持特定硬件平台（如华睿2号），其测试需在具备相应硬件环境的条件下进行。相关配置参考手册第21章。）

### 2.1.3 QoS影响分析

系统性地测试和对比关键QoS策略的不同配置对DDS中间件核心性能指标（吞吐量、抖动、时延、CPU/内存占用）的影响。通过提供详尽的量化数据，为用户优化其DDS应用的性能提供科学依据。将重点分析以下直接影响数据传输效率和系统资源消耗的QoS策略：

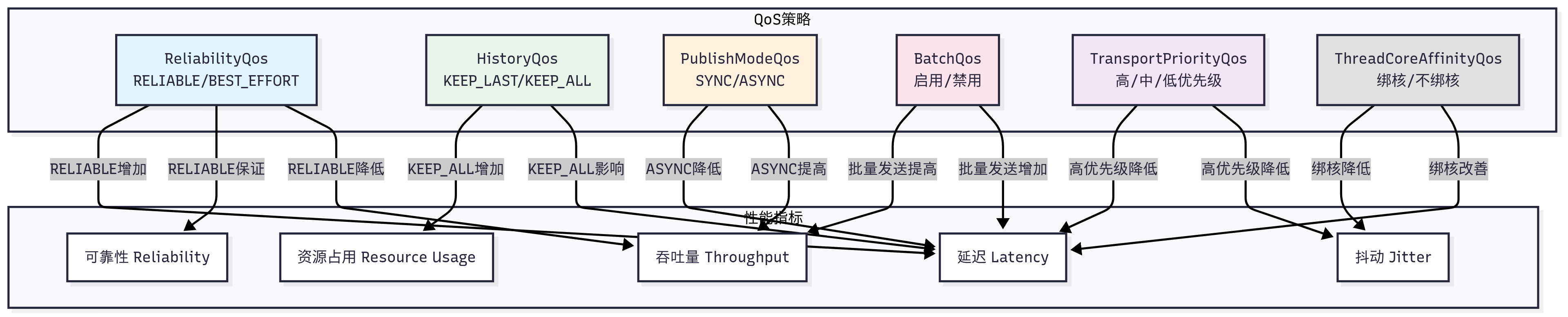


图1 QoS策略对性能指标的影响关系图

·ReliabilityQosPolicy (可靠性): 对比 DDS\_RELIABLE\_RELIABILITY\_QOS（可靠）与 DDS\_BEST\_EFFORT\_RELIABILITY\_QOS（尽力而为）两种模式。分析可靠传输的握手、重传机制对吞吐量、时延和抖动带来的性能开销。（参考手册第10.25节和第12章）

·HistoryQosPolicy (历史): 对比 DDS\_KEEP\_LAST\_HISTORY\_QOS（保留最新N个）与 DDS\_KEEP\_ALL\_HISTORY\_QOS（保留所有）两种模式。分析不同数据保留策略对DataWriter发送队列和DataReader接收队列的内存资源占用，以及在高负载下对时延的潜在影响。（参考手册第10.10节）

·PublishModeQosPolicy (发布模式): 对比 DDS\_SYNCHRONOUS\_PUBLISH\_MODE\_QOS（同步）与 DDS\_ASYNCHRONOUS\_PUBLISH\_MODE\_QOS（异步）两种模式。重点分析同步模式下write()操作在用户线程中直接发送数据对用户线程执行时间的影响，以及异步模式下后台线程带来的CPU开销。（参考手册第10.20节）

·BatchQosPolicy (批量发送): 对比启用与禁用批量发送（enable）的性能差异。分析批量发送将多个小数据包合并为一个网络报文的行为对吞吐量（减少协议开销）的提升效果，以及由此引入的时延和抖动增加。（参考手册第10.2节）

·TransportPriorityQosPolicy (传输优先级): 测试不同传输优先级设置对数据包在网络栈中处理优先级的影响。评估高优先级数据是否能获得更低的网络时延和抖动，特别是在网络拥塞的情况下。（参考手册第10.36节）

·ThreadCoreAffinityQosPolicy (线程绑核): 测试将DDS内部关键线程（如接收线程、异步发送线程）绑定到特定CPU核心的效果。评估线程绑核对CPU缓存命中率的改善，以及对时延抖动的降低作用，从而提升性能的稳定性和可预测性。（参考手册第10.27节和第21.4节）

### 2.1.4易用性与扩展性

通过解析用户预定义的JSON或XML配置文件来驱动整个测试流程，实现自动化测试。支持用户通过IDL文件定义自定义数据类型，并灵活配置测试场景。（参考手册第20章）

## 2.2 范围

### 2.2.1功能范围

1.核心实体管理：工具将创建并管理 DomainParticipant、Publisher、Subscriber、Topic、DataWriter 和 DataReader 等DCPS模型中的核心实体（参考手册第6-9章）。

2.主题 (Topic) 配置：

·支持创建和管理多个 Topic 实例。

·每个 Topic 关联一个由IDL文件定义的、已注册的 type\_name。

·支持为 Topic 配置其专属的QoS策略（TopicQos），并利用此QoS为 DataWriter 和 DataReader 的QoS提供初始化参考（参考手册7.2.4节，例8-9, 9-9）。

3.QoS策略配置：

·重点覆盖：工具将实现对第2.1节中列出的核心QoS策略的配置。

·配置方式：

a.直接API调用：工具在代码中直接构建 DataWriterQos 和 DataReaderQos 对象，并通过 create\_datawriter() 和 create\_datareader() 传入。

b.XML QoS文件（推荐）：工具将支持通过 DDS::DomainParticipantFactory::reload\_qos\_profiles() 接口加载外部的XML QoS配置文件，并使用 create\_datawriter\_with\_qos\_profile() 等接口应用预定义的QoS配置（参考手册第20章）。这将极大提升配置的灵活性和复用性。

·配置时机： QoS配置主要在实体创建时完成。对于创建后允许修改的QoS（如 DeadlineQosPolicy），工具将提供 get\_qos() 和 set\_qos() 的支持（参考手册8.3.3.2, 9.3.3.2）。

4.特殊功能测试：

·大包零拷贝：在满足手册第14章要求的前提下（如配置 DataWriterMessageModeQosPolicy 启用报文头合并，使用 ZeroCopyBytesDataWriter::write 接口），测试其在大消息（>1KB）传输时的性能优势，并与普通模式进行对比。

·基于内容的过滤 (ContentFilteredTopic)： （可选）支持创建 ContentFilteredTopic 来测试过滤功能对网络流量和接收端处理性能的影响（参考手册第7.3节）。

### 2.2.2数据范围

1.消息大小：测试数据大小覆盖从128字节到16384字节的范围，以模拟从小数据包到大块数据的不同负载场景。

2.数据类型：支持用户通过IDL文件定义任意复杂的数据结构。工具将调用zrddsgen.exe生成相应的C/C++代码，并在运行时通过register\_type()接口注册数据类型。

### 2.2.3环境范围

1.操作系统：工具本身及测试用例均支持在Windows和Linux平台上编译和运行（参考手册第11.4, 11.5节）。

2.硬件与网络： 基准测试假设在1Gbps以太网环境下进行，参考硬件配置为2GHz CPU、4GB RAM（参考手册第11.1节）。测试结果将明确标注其环境依赖性。

### 2.2.4限制条件

1.本工具仅作为ZRDDS的测试客户端，所有操作均通过ZRDDS提供的公共API（C/C++）完成，不涉及对ZRDDS中间件源码的任何修改或逆向工程。

2.本工具专为ZRDDS设计，不保证兼容或支持其他厂商的DDS实现（如RTI Connext, eProsima Fast DDS）。

3.所有性能测试结果均受制于执行测试的具体硬件、操作系统、网络状况和系统负载。测试结果仅在相同或高度相似的环境下具有可比性。

# 三、功能需求

## 3.1解析json文件

功能描述：DDS性能测试工具工具需自动读取并解析用户预定义的JSON配置文件。

json文件包含：domainID（域号），topic\_name（主题名），DomainParticipant（域参与者），DataWriter（数据写者） , DataReader（读者），QoS名称，payload\_size（数据长度），message\_rate（发送周期）。

具体输入：预先定义的json文件

具体输出：在内存中生成结构化的实体对象，供后续模块使用。

## 3.2 DDS实体的配置与管理

功能描述：根据解析出的配置信息，动态创建和配置DDS实体

​ 创建流程​：

1.创建DomainParticipant（域参与者），注册Topic（主题）。

2.根据QoS名称查找并应用相应的QoS策略，创建数据写入者（DataWriter）和/或数据读取者（DataReader）。

​ 3.可能的需求​：工具可能需要同时支持创建数据发布端和数据订阅端的实体，以完成完整的端到端测试。

## 3.3性能测试套件

1.吞吐量测试：测量系统在单位时间内成功传输的数据量（通常为bps或pps）。设定数据量和数据类型，在不同QoS策略下，配置发布端以最大速率发送数据。记录发送时间戳、数据包大小，订阅端统计接收到的包数，计算传输数据总量并验证不同QoS下的吞吐量表现。

2.时延测试：测量数据从发布端发出到订阅端接收到所经过的时间（往返时延RTT或单向时延）。嵌入时间戳，计算时延 = (T2 - T1) / 2。测试不同QoS下的时延，并验证是否符合预期，通常Reliable模式会导致较高时延。

3.抖动测试：在稳定网络下进行多次时延测试，计算时延变化的标准差或方差，测量时延的变化程度（即连续报文时延的差值），确保系统能稳定处理数据传输。

4.系统资源占用：使用性能监控工具观察CPU、内存和带宽使用，确保系统资源占用在合理范围内，负载增加时不超出系统能力，监控测试过程中DDS应用本身的资源消耗。

5.配置不同topic，QoS的能力：测试是否可以顺利配置不同主题，QoS——创建并测试多个主题的配置，确保每个主题的数据传输独立，并且在不同QoS下数据传输性能符合预期，测试Reliable和Best Effort等QoS对吞吐量、延迟的影响。确保DDS能够在不同主题上进行数据发布和订阅，并且每个主题的数据传输是独立的，互不干扰。验证系统是否能够根据不同的QoS策略正确地处理数据传输。

## 3.4数据采集与结果输出​

在测试过程中，实时将各项指标（时间戳、吞吐量、时延、抖动、CPU、内存等）记录在内存或缓冲区。测试结束后，将数据写入CSV文件。使其易于进行后续分析和可视化。最后考虑集成开源图表库直接生成可视化图表报告，动化采集测试数据并生成结构化报告并且将结果自动输出到表格。

# 四、非功能性需求

在设计DDS中间件性能测试工具时，除了功能性要求之外，还必须明确工具在性能、可用性、稳定性等方面的非功能需求。这些要求决定了工具在真实测试环境中的实用性和可靠性。

首先是性能。作为一款性能测试工具，系统必须保证自身开销尽量可控，避免因过高的计算和存储消耗而对测试结果造成干扰。在延迟与抖动的测量场景下，系统应尽可能降低额外的测量延时，并确保这种额外开销在文档中可被量化，从而保证数据的可信度。

其次是可扩展性。性能测试往往随着应用场景的变化而增加新的需求，因此工具必须具备灵活的扩展能力。在配置解析模块中，应当允许JSON文件定义新的参数或QoS策略，并保证向下兼容。在性能测试模块中，应支持便捷地增加新的测试指标，例如丢包率、带宽占用率或延迟分布曲线。这样的设计能够保证该工具在未来仍能满足不同场景下的测试需求。

在易用性方面，工具应尽可能降低使用门槛。用户只需准备JSON配置文件即可运行测试，无需复杂的命令行操作或代码修改。测试结果的输出应简洁直观，CSV文件不仅要包含原始数据，还应生成带有统计指标的结构化表格，以方便用户使用Excel或数据分析工具进行后续可视化与对比。

稳定性则关系到测试的可靠性。工具必须能够支持长时间运行，即使在高负载和高频率消息收发的情况下，也能维持稳定的运行状态。对于常见的网络异常、数据丢失或QoS配置错误，工具应具备清晰的日志和提示信息，使用户能够快速发现并解决问题。

最后是可维护性。系统的源代码结构应当模块化清晰，配置解析、实体管理、性能采集和结果输出应当相互独立，便于单独修改和优化。

# 五、验收标准

验收标准是确保DDS中间件性能测试工具满足用户需求和设计目标的重要依据。本章将从功能完整性、性能指标准确性、系统稳定性、易用性等多个维度制定详细的验收标准，为项目验收提供明确的评估框架。

## 5.1功能验收标准

### 5.1.1 JSON配置文件解析功能验收

工具必须能够准确解析符合规范的JSON配置文件，并正确处理各种异常情况。验收测试将使用包含完整配置项的标准JSON文件，验证工具能否正确提取域号、主题名、QoS策略、数据长度、发送速率等关键参数。测试用例应涵盖各种数据类型的配置项，包括整数、字符串、布尔值和嵌套对象结构。对于格式错误的JSON文件，工具应能在30秒内检测出错误并输出具体的错误位置和原因描述，错误信息应包含行号、列号和问题类型，帮助用户快速定位和修复配置问题。同时，工具还需支持配置文件的向下兼容性，当遇到未知的配置项时应给出警告而非直接终止程序执行。

### 5.1.2 DDS实体创建和管理功能验收

验收测试需要验证工具能够根据配置信息正确创建完整的DDS实体层次结构。测试过程将检查DomainParticipant是否能在指定的域中成功创建，Topic是否能正确注册指定的数据类型，以及Publisher、Subscriber、DataWriter和DataReader是否能按照配置的QoS策略正常创建。特别需要验证的是，当配置多个Topic时，工具应能为每个Topic创建独立的实体对象，确保数据传输的隔离性。验收标准要求工具在创建任何DDS实体失败时，能够提供详细的失败原因和建议解决方案。对于QoS策略的应用，验收测试将通过DDS API查询已创建实体的QoS配置，确认实际应用的QoS策略与JSON配置文件中指定的策略完全一致。

### 5.1.3 性能测试执行功能验收

性能测试功能的验收将通过一系列标准化测试场景来验证工具的测试能力。验收测试需要确认工具能够准确执行吞吐量、延迟、抖动和系统资源占用等四类核心性能测试。在吞吐量测试方面，工具应能在指定的测试时间内稳定发送数据，发送速率的误差应控制在配置值的5%以内。对于延迟测试，验收标准要求工具能够准确记录每条消息的发送和接收时间戳，时间戳精度应达到微秒级别，并且能够正确计算端到端延迟。抖动测试的验收要求工具能够连续监测延迟变化，并提供标准差、方差等统计指标。系统资源监控功能的验收将验证工具是否能够实时采集CPU使用率、内存占用量和网络带宽利用率等关键指标，监控数据的更新频率应不低于每秒一次。

## 5.2 性能指标验收标准

### 5.2.1 测量精度验收标准

性能测量的准确性是工具价值的核心体现，因此验收标准对测量精度提出了严格要求。延迟测量的验收标准要求工具在1Gbps网络环境下能够检测到微秒级的时间差异，测量误差应控制在实际延迟值的2%以内。为验证测量精度，将使用已知延迟特性的参考系统进行对比测试，确保工具测量结果与参考值的偏差在可接受范围内。吞吐量测量的验收要求工具能够准确统计单位时间内传输的消息数量和数据总量，计算精度应达到小数点后两位。对于系统资源监控，验收标准要求CPU使用率的监控误差不超过3%，内存占用量的监控误差不超过5MB，这些精度要求将通过与系统自带的性能监控工具进行对比来验证。

### 5.2.2 测试覆盖度验收标准

验收测试将评估工具对不同测试场景的覆盖能力，确保工具能够适应多样化的应用需求。测试覆盖度的验收要求工具支持从128字节到16KB的不同数据包大小测试，每个数据包大小都应能获得准确的性能测试结果。对于不同的QoS策略组合，验收标准要求工具至少支持RELIABLE和BEST\_EFFORT两种可靠性模式、KEEP\_LAST和KEEP\_ALL两种历史策略，以及同步和异步两种发布模式的测试。验收测试还将验证工具对不同发送速率的支持能力，要求工具能够稳定执行从1 msg/s到10000 msg/s范围内的各种发送速率测试。跨平台兼容性验收将确认工具在Windows和Linux平台上都能正常运行，并且两个平台之间能够进行互操作性测试。

## 5.3 稳定性和可靠性验收标准

### 5.3.1 长时间运行稳定性验收

稳定性验收测试将评估工具在长时间运行条件下的表现，确保工具能够支持持续的性能监测需求。验收标准要求工具能够连续运行24小时而不出现崩溃、内存泄漏或性能显著下降的问题。在长时间运行过程中，工具自身的内存占用增长应控制在初始值的20%以内，CPU占用应保持相对稳定，不应出现持续上升的趋势。验收测试将监控工具在高负载条件下的表现，当配置为高发送速率（如5000 msg/s以上）时，工具应能维持稳定的测试执行，不应出现测试中断或数据丢失的情况。对于异常情况的处理，验收标准要求工具能够优雅地处理网络中断、目标进程异常退出等常见故障，在故障恢复后能够自动重新建立连接并继续测试。

### 5.3.2 数据完整性验收标准

数据完整性是确保测试结果可信度的关键因素。验收测试将通过发送已知序列号的测试数据来验证工具的数据完整性保证能力。在RELIABLE模式下，验收标准要求工具能够检测到任何丢失的消息，丢包检测准确率应达到100%。工具应能正确维护消息序列号，并在测试报告中准确统计发送消息总数、接收消息总数和丢失消息数量。对于接收到的重复消息，工具应能正确识别并在统计中进行相应处理。验收测试还将验证工具的时间戳记录功能，确保每条消息的发送和接收时间戳都能被正确记录和保存，时间戳数据不应出现丢失或错乱的情况。

## 5.4 易用性验收标准

### 5.4.1 用户界面和操作便利性验收

易用性验收将从用户体验角度评估工具的实用性。验收标准要求工具能够通过简单的命令行指令启动，用户只需提供JSON配置文件路径即可开始测试，无需复杂的参数配置过程。工具的帮助信息应当完整且易于理解，当用户输入"--help"或"-h"参数时，应显示所有可用选项和使用示例。错误提示信息的验收要求错误描述准确、具体，能够帮助用户快速定位和解决问题。当配置文件路径错误时，工具应提示正确的文件路径格式；当QoS配置不合理时，应指出具体的配置项和建议值。进度显示功能的验收要求工具在执行长时间测试时能够实时显示测试进度，包括已完成的测试时间、剩余时间估算等信息，让用户能够了解测试执行状态。

### 5.4.2 输出结果的可读性和实用性验收

测试结果的输出质量直接影响工具的实用价值。验收标准要求工具生成的CSV文件格式规范，表头信息清晰，数据列对齐整齐，数值精度统一。CSV文件应包含完整的测试配置信息和详细的性能指标数据，使用户能够通过Excel等常用工具进行进一步分析。验收测试将检查输出文件的编码格式是否为UTF-8，确保包含中文字符的配置项能够正确显示。对于性能指标的统计汇总，验收标准要求工具提供平均值、最大值、最小值、标准差等基本统计量，并且计算结果准确无误。测试报告的可视化功能验收将确认工具能够生成直观的性能趋势图表，图表应包含适当的标题、坐标轴标签和图例说明，便于用户理解测试结果和进行性能分析。

## 5.5 扩展性和兼容性验收标准

### 5.5.1 配置扩展性验收

扩展性验收将评估工具对未来需求变化的适应能力。验收标准要求工具的JSON配置解析模块能够处理新增的配置项而不影响现有功能，当遇到未知配置项时应给出警告信息但继续执行测试。对于新的QoS策略支持，工具应具备通过配置文件扩展的能力，而无需修改源代码。验收测试将通过添加新的测试场景配置来验证工具的扩展能力，确保工具能够适应不断变化的测试需求。数据类型扩展性的验收要求工具能够支持用户通过IDL文件定义的自定义数据结构，并能正确生成相应的代码和进行类型注册。

### 5.5.2 环境兼容性验收

环境兼容性验收将确保工具能够在不同的系统环境中稳定运行。验收标准要求工具在Windows 10及以上版本和主流Linux发行版（如Ubuntu 18.04+、CentOS 7+）上都能正常编译和运行。跨平台兼容性测试将验证Windows和Linux平台间的互操作性，确保不同平台上运行的工具实例能够正常进行数据通信和性能测试。网络环境适应性的验收要求工具能够在不同的网络条件下工作，包括有线网络、无线网络和跨子网的网络环境。验收测试还将验证工具对不同ZRDDS版本的兼容性，确保工具能够与目标DDS中间件的当前版本和未来版本保持良好的兼容性。

# 六、参考资料

《ZRDDS用户手册》

# 七、附录

## 7.1 附录A：示例测试场景

|  |  |  |  |
| --- | --- | --- | --- |
| 配置类别 | 配置项 | 配置内容 | 说明 |
| 域与主题 | 域号 | 0 | DDS域ID |
| 主题名 | TestTopic | 测试主题名 |
| Qos设置 | domain\_participant\_qos.entity\_factory.autoenable\_created\_entities | true | 创建的DDS实体自动启用 |
| data\_writer\_qos.reliability.kind | RELIABLE | 数据写者可靠传输模式 |
| data\_writer\_qos.reliability.max\_blocking\_time | 0.1s | 阻塞发送的最大等待时间 |
| data\_writer\_qos.  durability.kind | VOLATILE | 数据不持久化 |
| data\_writer\_qos.history.  kind | KEEP\_LAST | 保留最近的n条历史消息 |
| data\_writer\_qos.history.  depth | 10 | 保留最近的10条 |
| data\_reader\_qos.reliability.kind | RELIABLE | 数据写者可靠传输模式 |
| data\_reader\_qos.reliability.max\_blocking\_time | 0.1s | 阻塞发送的最大等待时间 |
| data\_reader\_qos.  durability.kind | VOLATILE | 数据不持久化 |
| data\_reader\_qos.history.  kind | KEEP\_LAST | 保留最近的n条历史消息 |
| data\_reader\_qos.history.  depth | 10 | 保留最近的10条 |
| 数据配置 | 数据长度 | 1024 Bytes | 每条消息大小 |
| 发送速率 | 1000 条/秒 | 消息发送速率 |
| 发送周期 | 60 秒 | 单轮测试持续时间 |
| 数据字段 | timestamp  sequence\_number  patload | 消息字段定义 |
| 数据类型 | TestData | DDS数据类型 |
| 操作模式 | 传输协议 | udp | 使用UDP协议传输 |
| 角色 | publisher | 当前节点角色 |
| 日志级别 | INFO | 日志输出级别 |
| 输出结果 | true | 测试结果输出到表格 |

## 7.2 附录B：预期输出表格模板

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 域名 | 主题名 | 数据大小(Bytes) | 发送速率  (msg/s) | Qos策略 | 吞吐量（msg/s） | 平均时延（ms） | 最大时延(ms) | 抖动（ms） | CPU占用（%） | 内存占用（MB） | 丢包率（%） |
| 0 | TestTopic | 1024 | 1000 | reliable/KeepLast（10） | 990 | 13.4 | 26.7 | 3.4 | 43 | 130 | 0.1 |