

综合上下文协议 unified@1

本文前半部分描述保留兼容的 unified@1; 新增 unified@2 的映射、工具和发布说明见文末。

实现日期: 2026-10-05。

选择与范围

在新 AgentVersion 中选择 unified@1 (综合上下文), 在系统提示词中保留 `{{context_protocol}}`。协议通过现有目录、参数表单、发布校验和 MCP 自动发现; 已有 AgentVersion 不自动迁移。各 API/Worker 节点必须部署包含新协议的同一版本。

协议包含各窗口独立的前缀压缩、窗口切换和消息折叠。不包含压栈/出栈、监控 Agent 或经验库。折叠没有取消或展开工具; 回读不会修改上下文状态。

默认参数:

```
{
  "guiProjection": true,
  "pageSize": 200,
  "checkpointMaxCharacters": 12000,
  "maxWindows": 128,
  "maxFolds": 1024
}
```

maxFolds 计入已经被前缀压缩覆盖的记录, 确保原文回读入口仍存在。通用 KV 的键数和正文大小限制仍然有效。新协议不启用任务结果缓存。

原始消息与状态

原始消息不可变, 同一会话使用统一的局部 MessageIndex。消息写入时附加 windowId; role=user, kind=input 还附加 inputRun=<RunId 的 N 格式>。窗口切换不重新分组旧消息。

状态继续使用现有 KV:

- activeWindow: 当前窗口, 初始为 main。
- taskMemory: 跨窗口目标、约束和进度。
- window/<id>: 窗口目标、交接输入、工作记忆、成果和状态。
- checkpoint/<id>、coveredThrough/<id>: 该窗口的前缀摘要和覆盖边界。
- fold/<id>: 窗口、首尾消息编号、有序原 MessageId 清单、调用 Agent 提供的简述。

窗口 A 的消息为 #1、#2、#7 时, A 的 coveredThrough=7 仅覆盖这三条, 不覆盖其他窗口的 #3~#6。当前 Run 的用户输入属于其接收时的窗口, 但在所有窗口的 LLM 视图中保留原文。新 Run 不再自动保留上一 Run 全部输入; 长期有效约束应保存到 taskMemory。

LLM 映射

每轮冻结 KV revision、当前 RunId 和会话消息上界，执行以下顺序：

- 1. 读取当前窗口中大于其 coveredThrough、且不超过冻结上界的原始消息。
- 2. 读取当前 Run 的 user/input 原文，与第 1 步按 MessageId 去重，按 MessageIndex 正序合并。这些输入不因压缩或窗口切换消失。
- 3. 将仍位于未压缩区域的折叠段替换为一条 assistant/context_fold 派生消息。其他窗口历史不自动进入本轮上下文。
- 4. 执行 GUI 历史投影，然后由宿主执行大型工具结果预览。
- 5. 拼接 system、assistant/context_header 和上述历史投影，校验 Tool Call/Result 配对后适配 Provider。

上下文头依次包含 agent_inputs、task_memory、active_window、working_checkpoint。不注入全部窗口目录或所有窗口的摘要。折叠消息位于原范围的起点，内容包含折叠 ID、窗口、首尾编号、简述和读取方法。

例如 A 为当前窗口，#1 是本轮用户输入，A 的 #2~#8 已压缩，#9~#12 是一个折叠，#13~#14 属于 B，A 的 #15/#16 是工具调用/结果，则映射为：

```
system
assistant/context_header: 公共任务、A 状态、A 前缀摘要
user: #1 原文
assistant/context_fold: #9~#12 简述
assistant/tool_call: #15
tool: #16
```

折叠不是工具结果，不带 ToolCallId，也不使用其中某条原文作为 SourceMessageId。快照中增加 ProjectionSource，记录派生类型、折叠 ID、会话、窗口、首尾编号和全部原 MessageId。相同状态及原文产生相同投影；派生头和折叠消息都不回写原始历史。

工具

复用窗口的 open、memory_set、complete、list、search、read 及公共任务记忆工具，另外提供：

工具	参数（均另需 purpose）	行为
context_history_list	afterIndex=-1、limit=20（最大 50）	返回当前窗口未压缩历史的完整交互组、真实编号、短预览及 foldable；用 nextAfterIndex 续页
context_fold_create	fromIndex、throughIndex、summary（最多 2000 字符）	折叠连续完整组，返回 foldId
context_fold_list	windowId（默认当前）、offset=0、limit=20（最大 50）	返回折叠简述和 compressed 状态；用 nextOffset 续页
context_fold_read	foldId、cursor=0	读取原始消息 JSON 的文本片段，每次最多 12000 个 UTF-16 字符；按 nextCursor 续读，片段不保证构成完整 JSON

范围须来自实际查询结果。工具调用和结果必须整体折叠，多个尚未结束的调用组成同一交互组。未配对调用、用户输入、审批反馈、运行时指令不可折叠；不允许重叠、嵌套、跨窗口或跨前缀边界。创建折叠的调用自身在冻结上界之外，不会被本次操作折叠。

简述由执行 Agent 提供，不额外启动摘要 Agent；它属于历史工作记录，不代表新指令或已经核验的事实。折叠读取允许读同一授权会话内其他窗口的指定折叠，但不切换窗口，不允许指定其他会话或实例。

所有协议工具经原有 dispatcher 执行。宿主从原始调用记录读取生成时的消息上界和状态 revision，并提供会话受限 reader；模型不能指定这些授权参数。过期状态返回可纠正错误，重新推理后再调用。KV 与工具执行回执同事务提交，成功调用重放原回执，不重复创建折叠。运行时每轮仅支持一个工具调用。

窗口切换的调用和结果都属于原窗口，下一次推理才使用目标窗口；即使调用已经冷归档，恢复后的结果归属也相同。

前缀压缩与折叠

沿用 ManagedContextProtocol 的 context-maintainer 和真实 input token 阈值自动维护；若 Agent 绑定了 context-maintainer，也可以在阶段完成后主动调用。

维护读取该窗口的原始消息，不用折叠简述替代原文。统一协议额外冻结维护期间的父状态 revision、范围和源内容哈希；分页读取和提交时检查状态一致性，禁止维护边界切开折叠。原始消息和折叠回读记录均保留。

前缀完整覆盖折叠后，LLM 不再看到该折叠简述，由前缀摘要承接；context_fold_list 标记 compressed=true，context_fold_read 仍可回读原文。不回退 coveredThrough，也不恢复整段旧历史。

当前 Run 的用户原文始终属于受保护输入。单条超大用户输入、公共记忆或工具 Schema 本身超限，不能依赖折叠历史解决；沿用系统现有容量限制。

验证入口

- UnifiedContextProtocolTests：窗口筛选、输入保护、分页、来源映射、配对/范围限制、压缩优先级、跨窗口原文回读和过期状态拒绝。
- Unified_context_fold_archive_window_and_maintenance_roundtrip：开启 AGENTCORE_E2E=1，使用本地临时 PostgreSQL/Redis，验证真实 dispatcher、快照、Provider 消息、归档恢复、独立维护和折叠回读；不调用外部模型。
- 旧协议的上下文恢复与 Managed maintenance 专项用于兼容回归。

本次验证记录

- 上下文相关单元测试 31 项通过，其中新增综合协议用例 13 项；架构测试 1 项通过。
- 开启 AGENTCORE_E2E=1 的综合协议与旧上下文存储/恢复专项 2 项通过，使用临时 PostgreSQL 16、Redis 7 和脚本化模型。另验证了旧协议的自动维护 Worker 链路。
- 综合协议专项包含：真实 Provider 消息、工具回执重放、冷热原文合并、模拟旧归档内未完成调用的恢复、压缩后输入保护与原文回读、维护期间 revision 变化拒绝分页与提交。
- 扩大回归时发现两项外围失败：旧主动维护测试仍绑定全局 Agent 工具，创建运行时被 cross_application_tool_forbidden 拒绝；审批长文本单测仍断言超过 100000 字符报错，与工作区同期修改的 SessionFileInputPolicy 不一致。这两项未作为本协议修改范围处理。全量单元测试当次结果为 180 通过、1 失败。
- 没有调用真实模型供应商，也没有部署或迁移业务数据库。

unified@2：窗口恢复与操作意图校验（2026-10-05）

首次 PPE 真实模型测试见 qa/unified-ppe-20261005/report.json。v1 的窗口、折叠与主动压缩机制工作正常，但恢复 alpha 时，模型受该窗口历史末尾旧的“打开 beta”影响，误将 beta 排期写入 alpha。单独追加系统提示词仍产生同类错误意图。

v2 是单独注册的行为版本，保留 v1 的提示词、工具接口及选择方式；不自动迁移旧 AgentVersion。核心实现与提示词均在 UnifiedContextProtocolV2.cs。

模型消息映射

1. system：静态综合协议提示词，通过 `{{context_protocol}}` 插入。
2. assistant/context_header：Inputs、公共记忆、当前窗口、当前窗口检查点。当前窗口 JSON 直接显示中文，减少转义开销。
3. 本轮受保护用户原文，以及当前窗口未压缩历史；完整折叠以带来源的 assistant/context_fold 替代。
4. assistant/context_state：本轮 KV 冻结状态生成的当前执行位置，始终放在历史之后；包含当前 windowId/status/revision、本次从哪个窗口进入、进入前的消息上界、交接 input、检查点位置、本次进入后仍可见的工具结果数量和最近工具名。

末尾消息不是原始消息，不写入会话，不赋予原始消息 ID/序号，不伪造工具结果。原始切换调用/结果仍属于离开的窗口，保持配对。切换成功时 currentWindowEntry 与窗口更新在同一次 KV 事务中保存；恢复后旧的离开记录仍可审计，但不能代表当前执行位置。交接 input 是进入时的任务数据，模型需结合随后记录和检查点判断进度，不能每轮重演交接。

工具接口

context_window_open、context_window_memory_set、context_window_complete、context_fold_create 新增必填 expectedWindowId。它表示模型认为当前所在的窗口，不是指定任意写入目标。open 的 windowId 仍表示目的窗口；open 另要求非空 input，传递简短的已完成事项、下一步和返回位置。

例如从 beta 恢复 alpha：

```
{"purpose": "复核报价", "expectedWindowId": "beta", "windowId": "alpha", "resume": true, "input": "beta 排期已完成；只核对 alpha 检查点，不覆写报价，随后汇报。"} 
```

恢复后若 memory_set 声明 expectedWindowId=beta，返回 success=false、error.code=context_window_mismatch、实际 currentWindowId=alpha 与 stateRevision，不修改任何窗口。缺失参数返回 context_window_expectation_required（通过宿主 Schema 校验调用时也可能先被拒绝）。原有冻结 revision 检查继续有效。宿主记录结构化错误后可继续推理，不会把它误当字符串解析而中断。

此校验防止模型认为自己在另一个窗口时静默写错。它不能证明模型的记忆内容正确；模型即使填写正确窗口 ID，也必须遵守用户“不覆写”的要求。因此发布后仍需真实模型复测。

使用规则与成本

v2 使用统一决策提示词，不再拼接三段功能介绍。按任务边界选择窗口，按同一目标内的冗长完整片段选择折叠，仍需维护大量历史时才压缩；三者不是固定执行序列。完成全部业务后直接回答，不重复初始化或查询已可见的信息。摘要不抄工具流水账，公共记忆、窗口记忆和成果不重复全文。

v2 默认 checkpointMaxCharacters=3000，v1 仍为 12000；范围仍是 1000~24000，可显式配置。这里限制的是摘要字符数，不是输入容量或累计 token 预算。其他默认参数不变。未修改共享 context-maintainer 模板，也未改动自动压缩的容量调度逻辑。

发布与复测

将本次后端代码部署到 PPE 的所有 API 和 Worker 节点，避免只有 API 认识新版本。无数据库迁移或前端发布要求。发布后 MCP 协议目录应同时提供 unified@1 与 unified@2。

测试时新建/复制专用 AgentVersion，选择 contextProtocolId=unified、contextProtocolVersion=2，保留 `{{context_protocol}}`，Options 用 `{}` 采用新版默认摘要预算；不必修改业务 Agent。若测试主动压缩，继续绑定应用内的 context-maintainer。由 Codex 在发布后完成测试配置和真实模型运行。

复测沿用原报价/排期恢复用例和三阶段业务任务，核对末尾状态确实进入 Provider 请求、错误窗口被拒绝且能继续、恢复后的报价记忆不被覆盖、折叠与压缩回读仍有效；比较重复工具调用、实际 token 成本和最终结果。不能将本地脚本化模型测试当作真实 LLM 效果验证。

本次本地验证：全量单元测试 190 项、架构测试 1 项通过；开启 AGENTCORE_E2E=1 的 PostgreSQL/Redis 综合协议集成测试 v1/v2 两项通过。集成测试覆盖 Provider 末尾状态、错误窗口写入拒绝后的回执重放和继续运行、新版摘要预算传递、冷热归档、维护和折叠回读。解决方案编译通过，零警告、零错误。PPE 的 v2 真实模型复测需在 API/Worker 发布后进行，尚未作为本次通过项。

unified@3：消息映射对照实验（2026-10-05）

v2 后续 PPE 实测见 qa/unified-v2-ppe-20261005/report.json：底层窗口校验有效，但两次简单预算题答非所问；同模型 v1 对照正确。请求快照中的用户输入正确，不能仅凭这个对照就断言末尾 assistant 是唯一原因，因为 v1/v2 的提示词和工具也不同。

v3 仅用于实验，不自动替换 v1/v2。配置 stateMessagePlacement 选择以下映射；三组共用相同的位置中立提示词、工具 Schema、状态文本、窗口行为与默认 3000 字符摘要预算。动态状态不提升为 system。

组	配置值	原始消息到 LLM 消息的顺序
A（默认控制组）	trailing_assistant	system → assistant/header → 原始历史及折叠 → assistant/state
B（只改位置）	before_history_assistant	system → assistant/header → assistant/state → 原始历史及折叠
C（只改角色）	trailing_user	system → assistant/header → 原始历史及折叠 → user/state

例如 Options：{"stateMessagePlacement":"before_history_assistant"}。不能传入任意角色或 system。三组的 context_state 都是同一次冻结状态派生的模型视图，不写入原始会话、没有原始消息 ID/序号，不属于用户输入、窗口历史、折叠或压缩范围；C 的 user 角色也不构成用户的新授权。原始历史的顺序、用户原文、工具调用/结果配对与来源保持不变。先验证原始工具配对，再放置状态块，派生消息不能补足缺失的 tool_result。

为避免组间措辞自相矛盾，v3 三组共同移除了 v2 中“最后的/末尾状态”和“以上是历史记录”的位置限定。因此 A 与旧 v2 不是完全相同的提示词；旧 v2 会作为额外复现哨兵。若三组都恢复正常，只能说共同改动后的异常未复现，不能归功于状态位置。

实验步骤、固定输入、预期结果、预算与停止条件已预先写入 qa/unified-mapping-ab-20261005/experiment-plan.json。先做每组 3 次相同简单题，再让通过组做 2 次工具结果续接；全部通过的组才进入真实三轮 conversation（报价 → 排期 → 预算改为 9000 后回报价）。多轮使用同一根会话和 run_send_input，不以独立单轮冒充多轮。

多轮同时比较 prefix@2 的宿主维护普通会话基线。该基线不暴露手动上下文工具，也没有阶段意图自动切窗路由器；短用例未触发容量阈值时，不得宣称已经验证自动压缩效果。此轮只筛选自主控制是否比最低复杂度方案有收益，不预先建设完整的宿主阶段路由系统。

发布需要更新 PPE 所有 API/Worker，使目录包含 unified@3 及上述选项；无数据库迁移，无必须的前端变更。发布前的本地机制验证与发布后的真实模型验证分别记载，实验通过也不等于生产稳定性保证。

本地验证已完成：197 项单测、1 项架构测试通过；开启 AGENTCORE_E2E=1，使用临时 PostgreSQL 16、Redis 7 的 5 组集成测试通过（v1、v2 和 v3 三种映射）。验证了实际 Provider 请求中的状态角色/位置、原文与工具配对、冷热归档、窗口切换、错误后继续、维护与折叠回读。解决方案编译 0 警告、0 错误。实验阶段尚未在 PPE 启动；等待发布后按固定计划执行。