

TMPA Core Specification（中文）

Zhu Wei - joinwell52 Research Center

2026-08-11 - S1.0 - TMPA V1.0

Contents

TMPA Core Specification	2
文本化多智能体流程架构——Core 对象、Reader 语义与一致性	2
1. 状态、范围与一致性边界	3
2. 术语与表示阶段	3
3. Core 架构模型	4
3.1 治理对象	5
3.2 文档类型注册表	5
3.3 角色与权限模型	5
3.4 生命周期模型	6
3.5 文本消息、单写者流与异步并行	6
3.6 读端聚合与治理重建	7
3.7 完整性与签名证据	7
3.8 从 FCoP 工程实践抽象的治理闭环	8
3.8.1 当前状态、迁移历史与业务完成	8
3.8.2 回执、声明与证据门控	8
3.8.3 父子工作与闭环汇总	8
3.8.4 治理裁决、风险与人工审批	8
3.8.5 失败、恢复、巡检与漂移	9
4. 规范对象、编码与重建	9
4.1 规范对象 Schema	9
4.2 规范文本编码 Profile	11
4.3 聚合与治理重建程序	12
4.4 冲突与验证处理	13
5. 威胁模型与信任假设	13
5.1 受保护属性	13
5.2 信任根	13
5.3 纳入考虑的威胁	14
5.4 恶意参与者与存储表面受损	14
首次发布时结构正确但事实错误的证据	14
5.5 角色与身份分离层级	14
5.6 AI Agent 身份	15
5.7 安全声明	15
6. 生命周期与权限求值	15
6.1 必需注册表	15
6.2 迁移求值顺序	15
6.3 状态重建	16

6.4 权限时间与撤销	16
7. 三值治理逻辑	16
7.1 判断域	16
7.2 原子分类规则	16
7.3 组合规则	17
7.4 判断与视图映射	17
8. Reader 输入与输出契约	17
8.1 输入 Bundle	17
8.2 规范结果	17
8.3 规范化与排序	18
8.4 失败与部分输出	18
9. TMPA Core 规范	18
9.1 规范语言	18
9.2 对象要求	18
9.3 类型注册要求	19
9.4 角色要求	19
9.5 流要求	19
9.6 生命周期要求	20
9.7 引用要求	20
9.8 完整性要求	20
9.9 聚合与 Reader 重建要求	20
9.10 恢复要求	21
9.11 治理闭环、声明与人工控制要求	21
10. 一致性与可测试性	22
10.1 一致性层级	22
10.2 C01-C14 必需测试	22
10.3 可执行测试用例契约	23
10.4 裁决算法与一致性声明	23
10.5 Fixture 与结果报告	24
10.6 合规映射	24
11. Profile、版本与出版规则	24
11.1 Core 与 Profile 分离	24
11.2 版本	25
11.3 出版与证据边界	25
11.4 S1.0 发布与证据记录	25
11.5 S0.5 FCoP 派生历史基线	25
附录 A: 历史来源可追踪性（说明性）	25
附录 B: FCoP 来源交叉映射（说明性）	26

TMPA Core Specification

文本化多智能体流程架构——Core 对象、Reader 语义与一致性

规范版本： 正式发布版 S1.0

历史抽取基线： TMPA Draft V1.0-R24；当前规范直接在本 GitHub 文档中维护

状态： 正式稳定发布

抽取日期： 2026-07-31

发布日期：2026-08-11

权威性：本 GitHub 文档是 TMPA Core S1.0 的唯一规范性来源。Architecture Paper 负责理论阐释，Implementation Case Report 负责工程证据；二者均不得重定义本规范。

1. 状态、范围与一致性边界

本文档定义供应商中立的 TMPA Core，固定规范治理对象模型、单写者与生命周期语义、来源聚合、确定性治理重建、三值判断代数、Reader 输出契约、信任假设、规范要求以及 C01-C14 一致性行为。

TMPA Core 治理流程—责任证据。它不规定模型 Runtime、调度器、存储引擎、消息传输、数据库、文件系统布局、身份提供方、密钥管理系统或企业控制平面。Profile 可以绑定这些机制，但它 **MUST** 保留 Core 语义，并且 **MUST** 明确每一项附加假设或保证。

本正式发布版区分三类文本：

- **规范要求：**第 9、10 节，使用第 9.1 节规定的规范语言；
- **解释规范条款所必需的架构与算法定义：**第 2-8 节；
- **说明性的版本、出版与可追踪信息：**第 11 节和附录 A。

除非命名 Profile 明确增加，否则以下能力不属于 TMPA Core：经过认证的企业身份、凭证签发、递归委托、运行时准入控制、受保护或防篡改存储、分布式共识、语义真实性验证、拜占庭韧性、法律认证以及特定司法辖区合规。

符合规范的实现 **MAY** 使用文件、数据库行、对象存储对象或事件作为来源工件。实现 **MUST NOT** 仅因为存储 Markdown、生成日志或实现 workflow 状态机就声称符合 TMPA。一致性是行为性质，取决于第 9、10 节规定的可观察对象、生命周期、重建、冲突、恢复与测试结果。

三份维护文档的出版契约固定如下：Architecture Paper A1.0 解释理论，本 Core Specification 定义规范行为，Implementation Case Report I1.0 报告边界明确的工程证据。当前指导关系与历史协同演进必须分开表达：

当前指导与落实关系
TMPA 理论与架构
↓ 由下列文档形式化为规范行为
TMPA Core Specification
↓ 通过文件型 Profile 投影
FCoP 协议
↓ 被下列系统使用
CodeFlowMu 工程系统

历史协同演进
小典 AI 实践 → 早期 TMPA → FCoP 抽取与成熟
→ CodeFlowMu 工程落实
FCoP + CodeFlowMu 反馈 → 当前 TMPA 形式化

TMPA 理论指导 CodeFlowMu 工程落实，本 Core 固定接受评估的规范行为，FCoP 提供 CodeFlowMu 使用的协作协议。fcop 与 fcop-mcp Python Package 是 FCoP 协议的参考实现，不是 FCoP 本身。工程反馈可以推动后续 TMPA 修订，但 FCoP 协议、其参考实现和 CodeFlowMu 均不定义、也不穷尽 TMPA Core。

2. 术语与表示阶段

本规范固定以下词汇，防止把语义对象、物理存储、消息功能与重建视图混为一谈。

规范英文术语	固定中文译名	TMPA 中的固定含义	不等于
governed work item	受治理工作项	其责任与生命周期受到治理的任务、请求、决策或流程主题	一个文件、一个 Session 或一个 Runtime Job
primary carrier	主载体	锚定一项工作标识符和最低治理上下文的稳定治理对象	所有参与者共同编辑的可变记录

规范英文术语	固定中文译名	TPMA 中的固定含义	不等于
governance object	治理对象	由一个创建者、以一个责任角色、在一条写者流中编写的规范语义单元	其存储路径、传输封装或派生视图
textual message	文本消息	治理对象在传递工作、证据、复核或决策语义时承担的通信功能	单独的对象类别或短暂队列消息
state carrier	状态载体	对象、迁移记录或 Profile 规定位置对声明状态或当前状态证据作出贡献的持久化功能	共享可变应用状态
source artifact	来源工件	证据的一种物理表示或观测，例如文件、数据库行、对象存储项或收到的事件	验证后的语义治理对象
source candidate	来源候选	提交给聚合阶段的已发现来源工件，包括格式错误或互相冲突的观测聚合返回的、保留来源、完成解析、索引与确定性规范化的集合	已接受的权威对象
canonical candidate set	规范候选集合	一个可归属写者发布的、局部有序的治理对象序列	最终治理结论
writer stream	写者流	发现、保留、解析、索引与规范化来源候选，但不决定治理真值的阶段	全局 Event Log 或总时间线
source aggregator	来源聚合器	把固定 Profile 应用于规范候选集合的确定性阶段	治理 Reader
governance reader	治理 Reader	重建得到的偏序流程视图，以及规范化的未解决条件输出	存储层、编排器或模型 Runtime
governance graph and issue set	治理图与问题集合	Profile 状态机中由有效迁移或状态观测重建的当前阶段	原始来源证据或人为强加的总顺序
lifecycle state	生命周期状态	有权角色针对交付声明与支持证据发布的独立接受结论	业务验收、语义真实性或任务价值已经实现
business acceptance	业务验收	对某一工作项、交付物或子工作集合已经满足要求的可审查断言	执行者自述、终态目录位置或 done 标签
completion claim	完成声明	父工作与子工作之间保留范围、责任和闭环要求的显式关系	自证成立的完成事实
work derivation	工作派生关系	复核、批准、拒绝、要求修改、回避或升级人工的独立对象	运行时临时分叉或无来源的任务列表
governance decision	治理裁决	Reader、审计器或治理告警组件发现的可复现偏差或风险信号	生命周期中的 review 阶段本身
inspection finding	巡检发现		已自动执行的修复或业务裁决

三份文档统一使用三个语义判断：valid（**有效**）表示适用证据与规则建立接受结论；invalid（**无效**）表示规则建立拒绝或违规；undetermined（**未确定**）表示证据不完整、相互冲突或等待授权解决。authoritative、quarantined、partial、disputed、pending_human 等视图标签只解释呈现原因，不构成额外语义值。

同一个规范治理对象可以由不同物理 Profile 实现。在 FCoP 中，来源工件通常是文件及其路径与事件证据；其他 Profile 可以使用数据库行、对象或事件。两个来源工件若声明相同对象 ID、却包含不同规范内容，则不是无害副本，而是必须保留并按 Profile 评估的冲突候选。

本文中的 ****object（对象）****指语义单元，****artifact（工件）****指物理或已发布的工程表示，****view（视图）****指 Reader 派生的结果。实现及文档 **MUST** 保持这三个层次的区别。

3. Core 架构模型

TPMA 定义治理语义，而不是某个 Runtime 组件。它规定必须表示哪些治理事实、责任与生命周期如何表达，以及独立证据如何重建为可判断的治理视图。除非 TPMA Profile 明确绑定，否则存储、传输、调度与模型行为仍属于实现问题。

3.1 治理对象

治理对象是 TMPA 中最小的、能够独立归属责任的语义单元。它可以表示任务、报告、复核、批准、问题、生命周期迁移、角色分配、恢复动作或协议 Profile 发布的其他文档类型。对象不同于物理来源工件。

文本消息与状态载体描述对象承担的功能，而不是额外对象类别。对象在传递受治理工作或证据时承担文本消息功能；其内容、迁移证据或 Profile 规定位置参与生命周期重建时承担状态载体功能。

每个治理对象包含或标识：

- 稳定对象 ID；
- 文档类型；
- 一个创建者身份；
- 一个责任角色；
- 写者流 ID 与序号；
- 创建时间；
- 生命周期 Profile 与声明状态；
- 指向相关对象的类型化引用；
- 可选的父工作与 Thread 标识；
- 可选的完成、失败、恢复或验收声明及其证据引用；
- Profile 要求时的风险等级与人工审批要求；
- 规范文本内容；
- 完整性证据。

已发布治理对象不可变。更正 **MUST NOT** 擦除或重写原对象，而 **MUST** 创建新的对象，对原对象进行取代、拒绝、限定或解决。多个字节相同的来源观测可以指向同一对象；相同 ID 配合不同规范内容构成冲突，不构成更新。

对象声明的生命周期状态是发布时按照其 Profile 关联的状态。受治理工作的当前权威状态 **MUST** 从有效对象集合、已接受迁移和 Profile 规则中重建；实现 **MUST NOT** 通过修改旧对象或选择时间戳最新的对象得到权威状态。

生命周期状态与业务验收是两个正交维度。对象位于终态、归档位置或声明 done 只能构成状态证据；除非适用 Profile 明确规定验收权限、必需交付证据和职责分离，并且 Reader 找到有效的接受对象，否则它 **MUST NOT** 被重建为业务完成。

3.2 文档类型注册表

TMPA Core 不规定统一的业务文档分类。每个实现 Profile **MUST** 发布有限的文档类型注册表。每项注册内容包括：

- 类型名称与版本；
- 类型所表达的治理责任；
- 允许的创建者角色；
- 必填字段；
- 允许的引用关系；
- 适用的生命周期 Profile；
- 验证规则。

文档类型在权威上 **MUST NOT** 模糊重叠。例如，执行报告不得自动充当自己的独立复核或批准。部署允许职责分离例外时，例外及其批准权限 **MUST** 由显式对象表示。

3.3 角色与权限模型

TMPA 角色是具有确定范围的治理权限，不是 Prompt 标签或自然语言人格。

每个角色定义包括：稳定角色 ID、允许创建的对象类型、允许执行的生命周期动作、职责分离约束、角色授予方、分配有效期与撤销状态。

Profile **MAY** 把角色组织为执行、治理与管理等能力层，但必须声明层级含义、允许方向、禁止方向以及由何种身份或 Runtime 控制实际执行。声明性能力边界与已强制执行的能力边界 **MUST** 分开报告。

对象的 role 字段声明创建者以何种权限行动，但它不会自行创造权限。Reader **MUST** 根据有效角色分配与适用策略 Profile 验证该声明。

只有实现 Profile 明确允许时，一个参与者 **MAY** 同时占据多个角色。声称独立复核的部署 **MUST** 禁止同一身份对同一受治理结果同时担任执行者和复核者，除非存在经过记录并获得授权的例外对象。

在企业身份 Profile 中，逻辑角色应分别绑定到可验证的 Agent 或工作负载身份、继续承担责任的人类或组织主体、此次动作使用的凭证、委托来源与范围，以及有效或撤销状态。TMPA Core 记录和验证治理声明，但不签发凭证，也不强制递归权限衰减。

3.4 生命周期模型

生命周期 Profile 由以下元素构成：

- 有限状态集合 S ；
- 初始状态 s_0 ；
- 终止状态集合 F ；
- 动作集合 A ；
- 迁移关系 $T \subseteq S \times A \times S$ ；
- 授权函数 $\text{Auth}(\text{role}, \text{action}, \text{object})$ ；
- 验证函数 $\text{Valid}(\text{object}, \text{transition})$ 。

只有同时满足以下条件，迁移才被接受：来源状态匹配当前权威状态；迁移存在于 T ；发起角色已获授权；必需引用与前置条件已满足；迁移证据通过 Schema 与完整性验证。

非法或未经授权的迁移 **MUST NOT** 改变权威生命周期状态。该尝试 **MUST** 通过拒绝、ISSUE、告警或 Profile 规定的等价记录保持可观察，不得静默丢弃或修复。

生命周期 Profile 还 **MUST** 声明：(1) 哪些状态需要独立业务验收；(2) 哪些关系构成报告、复核、接受与归档授权；(3) 父子工作如何汇总；(4) 哪些风险等级需要人工批准；(5) 失败类型、恢复动作及其持久证据。生命周期工具或物理位置可以实现这些规则，但不能取代这些语义定义。

3.5 文本消息、单写者流与异步并行

TMPA 写入面由稳定工作载体、单写者对象、局部串行性和异步组合构成。

对于每个受治理工作项 t ，面向任务的 Profile **MUST** 定义一个稳定主载体 c_t 。主载体建立工作 ID 和最低治理上下文。接受、执行报告、复核、决策、更正与恢复记录 **MUST** 是引用 c_t 的独立对象，不得成为任务的额外可变副本。“一任务一载体”指一个稳定主引用点，而不是一份多人共同编辑的文档。

令 A 为责任写者集合。每个已发布对象恰好有一个写者；每个写者 $a \in A$ 发布一条可独立归属的串行流：

$S_a = \langle o_{\{a,1\}}, o_{\{a,2\}}, \dots, o_{\{a,n\}} \rangle$

S_a 内的序号是权威局部顺序。每个对象 **MUST** 具有正整数序号， $(\text{stream_id}, \text{sequence})$ 标识它在写者责任历史中的位置。创建时间仅供说明，**MUST NOT** 取代流内顺序。

在观测时刻 τ ，可见候选可以是不同流的不同前缀：

$O_\tau = \text{union}_{\{a \in A\}} \text{prefix}(S_a, k_a(\tau))$

各 $k_a(\tau)$ 无需同步推进。一个参与者可以先发布任务，另一个参与者稍后提交报告，多条流可以并行进展。这就是多条串行流形成异步并行的方式。TMPA 不要求共享时钟、同时在线或使用一个全局事件日志。

流内前驱关系产生局部顺序；显式引用与 Profile 定义的生命周期依赖产生跨流因果边。若两个对象既不存在流内关系，也不存在 Profile 定义的跨流依赖，它们 **MUST** 保持并发且不可比较。

单写者对象与独立责任流消除了主要的语义共享写冲突，但不消除所有存储争用、文件系统竞态或基础设施故障。Profile **MUST** 定义原子发布、重复处理与恢复行为。

一项任务有一个主载体；每个已发布对象只有一个写者；每个写者保持串行；多条流异步推进并形成并行协作。

3.6 读端聚合与治理重建

TMPA 的读取面包含两个概念上独立的阶段：**来源聚合与治理重建**。

令 O_τ 为观测时刻 τ 可见的有限来源候选集合。保留来源的聚合器 A 发现来源工件，保留来源身份与字节，解析候选封装，索引 ID 与引用，并执行 Reader 所需的确定性规范化：

$$C_\tau = A(O_\tau)$$

聚合器 **MUST NOT** 判断哪项声明为真，**MUST NOT** 静默修复冲突，**MUST NOT** 发明缺失证据，也 **MUST NOT** 把到达顺序转换为治理顺序。它的职责是构造治理 Reader 可用的完整规范候选集合 C_τ ，同时保留每项来源候选的 Provenance。

令 P 为固定规则 Profile，其中包含 Schema、类型规则、角色分配、生命周期规则、关系语义、规范化规则、冲突策略与输出规范化规则。治理 Reader 计算：

$$R_P(C_\tau) = (G_\tau, I_\tau)$$

其中 G_τ 是规范重建的偏序流程与治理图， I_τ 是规范问题集合。 G_τ 表达工作进度、责任、生命周期、复核、批准、拒绝、恢复与审计关系，同时保留局部流顺序、显式跨流依赖和不可比较对象之间的并发；它不是权威总时间线。

后文可用 $R_P(O)$ 简写组合 Pipeline $R_P(A(O))$ 。核心确定性要求是排列不变性。对于同一规范候选集合的任意排列 π ：

$$R_P(A(\pi(O))) = R_P(A(O))$$

等式适用于 G 与 I 的规范序列化，不适用于偶然的内存顺序或诊断格式。延迟到达会改变当前可用集合，因此可能合法地把 partial 视图改变为 authoritative 或 disputed；同一集合的不同枚举顺序 **MUST NOT** 改变结果。

确定性命题。 令 O 为有限来源多重集合， P 为固定 Profile。若：(1) 来源规范化是来源身份和被覆盖字节的纯函数；(2) 重复分类、验证、权限、生命周期检查和问题 ID 均由规范对象值与 P 决定；(3) 图边只从流内序号与 Profile 声明关系导出；(4) 图与问题序列化使用已发布的确定性排序及 Tie-break 规则，则 $R_P(A(O))$ 对来源枚举顺序不变。

证明概要。 任意枚举均被聚合为同一规范索引候选多重集合；逐对象验证与顺序无关；重复 ID、流缺口、缺失引用、禁止环、权限冲突和生命周期冲突均在同一集合与关系上计算。因此，每次排列得到相同的接受节点集合与有向边集合。规范问题 ID、确定性排序和只用于表示不可比较节点的稳定 Tie-break 产生字节等价的 G 与 I 。该命题只证明固定 Profile 下的排列不变性，不证明语义真实性、Profile 正确性、对受损信任根的抵抗或不同证据集合之间的相等。更强保证仍需机械化证明与可执行语料库。

重建主题或子图的视图分类为：

- **authoritative**：必需证据有效，且没有影响结论的未解决完整性、权限、生命周期、顺序或必需引用问题；
- **partial**：必需证据缺失，或流/依赖不完整；
- **disputed**：多项有效但互不兼容的治理声明尚未解决；
- **quarantined**：Profile 定义的完整性、身份、重复 ID 或禁止环条件，使受影响证据不能进入权威重建。

认证是正交的保证标签。对象或视图可以在 TMPA Core 下结构性 authoritative，但在更强身份 Profile 下仍未认证；此时 **MUST NOT** 将其表述为具有认证完整性。

TMPA 要求**保留冲突**：有效但矛盾的对象必须保持可见，直至出现新的授权解决对象。TMPA 也要求**扩展时保留证据**：增加候选证据不得擦除既有来源证据。治理状态不保证对任意集合扩展单调，因为新增有效证据可能把 authoritative 合法地改变为 partial、disputed 或 quarantined。

确定性拓扑序列 **MAY** 用于交换或显示，但不得增加不可比较节点之间的治理顺序。C11 验证聚合与重建确定性；C03、C04、C09、C10、C12 分别验证 ID、顺序、依赖、环与冲突保留行为。

3.7 完整性与签名证据

TMPA 区分三个经常被混淆的属性：

1. **归属 (Attribution)**：对象声明创建者与责任角色；

2. **完整性 (Integrity)**：能够检测对象发布后的修改；
3. **认证完整性 (Authenticated integrity)**：对象通过密码学方式绑定到已验证身份或密钥。

TMPA Core 要求归属与完整性证据。只有应用可信身份、签名与密钥管理 Profile 时，部署 **MAY** 声称认证完整性。

完整性记录标识：规范化 Profile、Hash 算法、内容 Digest、Profile 要求的前驱或引用 Digest，以及可选的签名算法、密钥 ID 与签名值。

角色标签不是密码学签名。没有可信身份绑定的 Digest 可以检测修改，却不能证明对象由谁创建。有效签名在部署信任模型内证明来源与完整性，但不证明签名内容在语义上真实。

3.8 从 FCoP 工程实践抽象的治理闭环

S1.0 从 FCoP 协议规范、Rules、Schema 与 ADR，以及 fcop / fcop-mcp 参考实现的有界观察中提炼供应商中立的 Core 约束。FCoP 是协议与参考 Profile，不是应用程序，也不是 TMPA 的定义者；Python Package 只是参考实现。因此本节吸收的是可移植语义，而不是 `_lifecycle/`、文件名或 MCP 工具名。

这种历史提炼属于工程反馈进入规范设计，不表示当前权威关系倒置。规范发布后，TMPA 理论与本 Core 约束 CodeFlowMu 的预期工程落实；观测到的产品行为可以支持、质疑或推动后续修订，但不能静默重定义当前要求。

3.8.1 当前状态、迁移历史与业务完成

Profile **MUST** 分别定义当前状态观测与迁移历史证据。FCoP 以路径表达当前阶段、以只增 transitions 表达历史；数据库 Profile 可以使用当前状态行与 Event 表。Reader 遇到两者冲突时 **MUST** 保留两份来源并输出 STATE_EVIDENCE_CONFLICT 或 Profile 声明的等价规范问题，不得用“最新时间戳”覆盖冲突。

生命周期进入 done、终态或归档 **MUST NOT** 自动建立业务验收。执行者的报告是可归属的交付声明；只有有权且满足职责分离的复核/接受对象才能建立业务完成。缺少接受对象时，完成结论为 undetermined，视图为 partial 或 pending_human。

3.8.2 回执、声明与证据门控

面向工作的 Profile **MUST** 定义任务与响应的互惠关系。每个被接收的工作对象最终 **SHALL** 关联报告、问题、拒绝、取消或后续工作对象之一；静默不能被 Reader 推断为成功。

完成、失败、恢复与验收声明 **MAY** 由 claims 表示。每个声明必须具有稳定 Claim ID、Predicate、Subject 与证据对象 ID 集合。完成声明缺少 Profile 要求的测试、产物、Commit、报告或其他证据时，Reader **SHALL** 输出 CLAIM_EVIDENCE_MISSING，并保持 undetermined。这条规则治理未经证实的声明，不声称消除模型幻觉。

3.8.3 父子工作与闭环汇总

子工作 **SHALL** 通过 governed_work.parent_id 或 Profile 声明的等价关系指向父工作；共享 Thread 可以通过 thread_id 表示，但 Thread **MUST NOT** 取代父子范围关系。父工作存在未结束、被阻塞但未处理或缺少回执的子工作时，父级完成声明 **SHALL** 产生 CHILD_WORK_OPEN 并保持 undetermined。

修正范围 **MUST** 通过新对象、取代关系或新派生关系表达，不得原地重写已发布父对象。Reader **SHALL** 保留父工作、全部子工作、各自回执及汇总结论。

3.8.4 治理裁决、风险与人工审批

生命周期中的待复核阶段与治理裁决对象是正交机制。Profile **MUST** 为两者分配不同类型或关系语义；实现 **MUST NOT** 仅因工作进入 review 阶段就推断存在独立复核，也不得用治理 REVIEW 取代执行报告。

Profile **MAY** 采用 low、medium、high、irreversible 风险等级。若对象声明需要人工审批，或其风险等级落入 Profile 的人工审批集合，则在有效人工批准对象存在之前，Reader **SHALL** 输出 HUMAN_APPROVAL_REQUIRED，判断为 undetermined，视图为 pending_human。Agent 自行改写批准结论无效。

3.8.5 失败、恢复、巡检与漂移

Profile **MUST** 发布有限失败类型与恢复动作注册表，并说明重试、继续、回滚、中止和升级如何形成新对象。失败 **MUST NOT** 通过成功报告隐藏；恢复动作 **MUST** 引用触发它的失败与被恢复工作。

协议巡检与治理告警是观察输出，不是自动修复。INSPECTION、ALERT 或等价对象 **MAY** 报告阻塞、规范或整洁级别的发现，但建议命令 **MUST NOT** 被 Reader 当作已经执行的迁移。独立治理信号与执行者自述 **MUST** 分开分类。

4. 规范对象、编码与重建

4.1 规范对象 Schema

以下 JSON Schema 定义 TMPA Core S1.0 规范治理对象。它约束单个治理对象的结构。ID 唯一性、流连续性、角色授权、生命周期合法性、引用解析和确定性重建等跨对象属性，由适用 Profile 与 Reader 评估，不能由单对象 Schema 单独建立。

实现只能在 extensions 下增加 Profile 特定字段，并且 **MUST** 保留所有 Core 字段的既定含义。

```
{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "$id": "urn:tmpa:schema:governance-object:s1.0",
  "title": "TMPA Governance Object S1.0",
  "$comment": "Structural validation does not establish role authority, lifecycle legality, cross-object uniqueness, or integrity verification.",
  "type": "object",
  "additionalProperties": false,
  "required": [
    "tmpa_version",
    "id",
    "type",
    "governed_work",
    "stream",
    "creator",
    "role",
    "created_at",
    "lifecycle",
    "references",
    "content",
    "integrity"
  ],
  "properties": {
    "tmpa_version": { "const": "S1.0" },
    "id": { "type": "string", "minLength": 1 },
    "type": { "type": "string", "minLength": 1 },
    "governed_work": {
      "type": "object",
      "additionalProperties": false,
      "required": ["id", "primary_carrier_id"],
      "properties": {
        "id": { "type": "string", "minLength": 1 },
        "primary_carrier_id": { "type": "string", "minLength": 1 },
        "parent_id": { "type": "string", "minLength": 1 },
        "thread_id": { "type": "string", "minLength": 1 }
      }
    },
    "stream": {
      "type": "object",
      "additionalProperties": false,
      "required": ["id", "sequence"],
      "properties": {
        "id": { "type": "string", "minLength": 1 },
        "sequence": { "type": "integer", "minimum": 1 }
      }
    },
    "creator": { "type": "string", "minLength": 1 },
    "role": { "type": "string", "minLength": 1 },
    "created_at": { "type": "string", "format": "date-time" },
    "lifecycle": {
      "type": "object",
      "additionalProperties": false,
      "required": ["profile", "state"],
      "properties": {
        "profile": { "type": "string", "minLength": 1 },
        "state": { "type": "string", "minLength": 1 },
        "transition": {

```

```

    "type": "object",
    "additionalProperties": false,
    "required": ["from", "action", "to"],
    "properties": {
      "from": { "type": "string", "minLength": 1 },
      "action": { "type": "string", "minLength": 1 },
      "to": { "type": "string", "minLength": 1 }
    }
  },
  "references": {
    "type": "array",
    "uniqueItems": true,
    "items": {
      "type": "object",
      "additionalProperties": false,
      "required": ["relation", "target"],
      "properties": {
        "relation": { "type": "string", "minLength": 1 },
        "target": { "type": "string", "minLength": 1 }
      }
    }
  },
  "claims": {
    "type": "array",
    "uniqueItems": true,
    "items": {
      "type": "object",
      "additionalProperties": false,
      "required": ["id", "predicate", "subject", "evidence"],
      "properties": {
        "id": { "type": "string", "minLength": 1 },
        "predicate": { "type": "string", "minLength": 1 },
        "subject": { "type": "string", "minLength": 1 },
        "evidence": { "type": "array", "uniqueItems": true, "items": { "type": "string", "minLength": 1 } }
      }
    }
  },
  "risk": {
    "type": "object",
    "additionalProperties": false,
    "required": ["level", "requires_human_approval"],
    "properties": {
      "level": { "enum": ["low", "medium", "high", "irreversible"] },
      "requires_human_approval": { "type": "boolean" }
    }
  },
  "content": {
    "type": "object",
    "required": ["media_type", "body"],
    "properties": {
      "media_type": { "type": "string", "minLength": 1 },
      "body": {}
    }
  },
  "additionalProperties": false,
  "integrity": {
    "type": "object",
    "additionalProperties": false,
    "required": ["canonicalization", "hash_algorithm", "digest"],
    "dependentRequired": {
      "signature_algorithm": ["key_id", "signature"],
      "key_id": ["signature_algorithm", "signature"],
      "signature": ["signature_algorithm", "key_id"]
    },
    "properties": {
      "canonicalization": { "type": "string", "minLength": 1 },
      "hash_algorithm": { "type": "string", "minLength": 1 },
      "digest": { "type": "string", "minLength": 1 },
      "signature_algorithm": { "type": "string", "minLength": 1 },
      "key_id": { "type": "string", "minLength": 1 },
      "signature": { "type": "string", "minLength": 1 }
    }
  },
  "oneOf": [
    {
      "properties": {
        "signature_algorithm": { "type": "string", "minLength": 1 },
        "key_id": { "type": "string", "minLength": 1 },
        "signature": { "type": "string", "minLength": 1 }
      }
    }
  ],
  {

```

```

    "required": ["signature_algorithm", "key_id", "signature"],
    "properties": {
      "signature_algorithm": { "type": "string", "minLength": 1 },
      "key_id": { "type": "string", "minLength": 1 },
      "signature": { "type": "string", "minLength": 1 }
    }
  },
  "extensions": {
    "type": "object",
    "additionalProperties": true
  }
}

```

各字段具有以下运行含义：

字段	Reader 义务
tmpa_version	选择兼容的 Core 对象 Schema 版本线；不得静默降级未知主版本
id	索引规范身份并检测同 ID 不同内容冲突
type	解析一个带版本的类型注册项
governed_work.id	对治理同一工作项的对象分组
governed_work.primary_carrier_id	标识后续证据必须解析到的唯一稳定主载体
governed_work.parent_id / thread_id	保留工作派生与会话无关的协作 Thread；Thread 不取代父子关系
stream	在不使用时间戳的情况下建立可归属局部顺序
creator 与 role	针对有效 Assignment 求值权限声明；字段本身不创建权限
lifecycle	标识 Profile 与声明状态；存在 transition 时提供显式 from/action/to 证据
references	根据关系注册表构建类型化顺序或非顺序链接
claims	表示可审查声明及其证据对象集合；字段存在不代表声明已经成立
risk	表示 Profile 规定的风险等级和人工审批要求；不自行构成批准
content	以声明 Media Type 承载受治理 Payload
integrity	标识被覆盖字节的规范化与验证程序
extensions	包含全部 Profile 特定扩展；未知扩展只能依据声明 Profile 处理

主载体对象把自己的 id 用作 governed_work.primary_carrier_id。同一工作项的其他对象重复该载体 ID。生命周期迁移文档类型 **SHALL** 要求 lifecycle.transition；非迁移类型 **MAY** 省略它。该条件由类型注册表而不是通用单对象 Schema 强制。

C01 使用的 Schema Processor **SHALL** 为 created_at 执行 JSON Schema Draft 2020-12 format 断言；仅把 date-time 当作注释不足以通过 C01。下方链接的 S1.0 机器可读工件是规范 Schema 字节序列；上方嵌入展示 **SHALL** 与其保持语义一致。

S1.0 机器可读工件	SHA-256
治理对象 Schema	a2829cd7149c3054a52886365f2293a23106b636b0c52799739bfabdab1ff4fa
生命周期 Profile Schema	481a61ac2485bbaf15d90e9c5a255ad9ce6a55971190f0fe404856be4b10f993
Reader 结果 Schema	4527df7096fe840b85b245e50d5cea576ff359d50a54d17c8873a7b4f458d431
一致性结果 Schema	4b1ecebf83e62d2aa1aff0e79a0cd0ea0a85fbc14a426d5fe873ab40aefdc2fe

生命周期 Profile Schema 除状态、动作、迁移与恢复规则外，还强制要求显式 acceptance、work_graph、risk_policy 与 failure_model 区段。S1.0 进一步要求风险策略标识允许的批准对象类型，以及是否要求批准人相互独立。这些区段使 FCoP 派生的协作周期语义可以审查，同时不把 TMPA 绑定到 FCoP 参考实现或 CodeFlowMu。

lifecycle.state 记录此不可变对象发布时声明的状态，不是可变当前状态字段。当前权威生命周期状态 **MUST** 从有效对象集合、已接受迁移证据与适用生命周期 Profile 重建。

规范化 Profile **MUST** 定义 Digest 覆盖的精确表示；使用签名时，还 **MUST** 定义签名覆盖的精确表示与自引用完整性字段的排除或规范化方法。TMPA Core S1.0 要求显式声明该 Profile，但不强制唯一的字节级规范化算法。

Schema 有效只是进入权威治理视图的必要条件而非充分条件。Reader 仍 **MUST** 检查 ID 唯一性、类型规则、流顺序、权限、生命周期、引用、Digest 与适用的签名策略。

4.2 规范文本编码 Profile

仅选择 JSON、YAML 或 Markdown 不能定义规范文本表示。只有当独立实现能够从同一受治理内容生成相同的被覆盖字节序列时，一个版本化规范化 Profile 才算完整。

该 Profile 至少 **MUST** 定义：

- 字符编码与 Unicode 规范化形式；
- 换行符规范化；
- 对语义上无序字段与集合的确定性排序；
- 空白、转义、引号与分隔符规则；
- 数字表示，包括指数、符号与精度；
- 时间戳语法、时区要求与小数秒规范化；
- 缺失、null、空字符串与空集合的区别；
- 文本正文是否逐字节覆盖，或如何规范 Markdown 空白与换行；
- 附件与外部证据的媒体类型、字节长度、内容 Digest 与可选定位元数据；
- 扩展字段是否被 Digest/签名覆盖，以及未知扩展如何排序或拒绝；
- 与对象绑定的 Schema、类型注册表与规范化 Profile 版本；
- Hash 或签名前如何排除或规范化自引用完整性字段。

语义等价不足以完成完整性验证。Unicode 形式、换行、数字拼写、时间精度、字段顺序或扩展处理不同的对象，即使人类认为含义相同，也可能产生不同 Digest。因此，C08 与 C11 只有在使用同一规范化 Profile 及版本时才有意义。

外部附件或可变 URL 不会因被引用就自动成为权威证据。声称外部内容完整性的 Profile **MUST** 记录内容 Digest 以及识别被覆盖字节所需的元数据。Locator **MAY** 帮助检索，但 Locator 本身不能保存被引用证据。

4.3 聚合与治理重建程序

给定有限无序来源候选集合 O 与固定规则 Profile P，符合规范的实现执行两个阶段：

AGGREGATE(O):

1. 发现每项来源工件，但不把发现顺序、文件系统顺序、传输顺序、修改时间赋予治理意义。
2. 保留来源身份与字节并解析候选封装；解析失败仍作为来源证据和确定性诊断保留。
3. 规范索引对象 ID、写者流、序号、主载体、引用、生命周期位置与完整性元数据。
4. 投影时可以去重字节相同的来源观测，但不得删除 Provenance；相同 ID、不同内容必须作为不同候选保留。
5. 返回保留来源的规范候选集合 C 与聚合诊断。

RECONSTRUCT(C, P):

1. 验证对象结构与文档类型；无效候选保留用于诊断，但排除在权威候选集合之外。
2. 按声明的规范化 Profile 重算 Digest；Digest 不匹配对象作为完整性失败证据保留，但不得进入完整权威候选集合。
3. 存在签名时验证签名，记录验证状态并应用 P 的接受策略；无法验证的签名不能建立认证完整性。
4. 按对象 ID 对完整候选分组：
 - a. 相同 ID、相同规范内容：投影为一个对象，同时按 Unicode 码点顺序保留每个贡献来源的 `source\_id`；
 - b. 相同 ID、不同规范内容：全部变体隔离，产生重复 ID 冲突。
5. 验证面向任务 Profile 的一主载体规则；后续报告、复核、决策和更正必须引用载体，不得形成歧义的可变任务副本。
6. 按写者流分组接受对象并依序号排序；检测重复序号与序号缺口，不得发明缺失对象，也不得用时间戳替代序号。
7. 只为 P 声明为顺序或生命周期依赖的关系类型建立有向边；其他引用保留为非排序链接。
8. 验证角色权限、职责分离、生命周期来源状态、迁移合法性、前置条件与必需证据；无效动作保持可观察但不改变权威状态。
9. 检测缺失引用与 P 禁止的环；只隔离受影响子图，保留未受影响的有效对象。
10. 从流内序号边与 Profile 定义的跨流依赖构造偏序流程与治理图；无关节点保持并发且不可比较。
11. 交换或显示需要线性序列时，生成确定性拓扑序列；对象 ID 字典序只能作为不可比较节点的 Tie-break，不得把该关系加入治理图。
12. 投影任务、消息、流程、责任、生命周期、复核、批准、恢复与审计视图。
13. 规范化治理图/视图与问题集合，并将二者一起返回。

组合运算为：

$$R_P(A(O)) = (G, I)$$

其中 A(O) 是保留来源的规范候选集合，G 是偏序流程与治理图，I 是规范问题集合。程序 **MUST** 保持以下不变量：

- 对于同一来源集合，发现、枚举和到达顺序不影响最终规范输出；
- 聚合保留来源证据，不静默决定治理冲突；
- 面向任务的 Profile 中，一项受治理任务只有一个稳定主载体；
- 每个已发布对象只有一个写者并属于一条局部串行流；
- 时间戳不得覆盖流序号或显式依赖；
- Profile 未定义因果或生命周期关系时，不得发明跨流顺序；
- 规范线性序列只是图的表示，不是额外治理事实；
- invalid、disputed 或 rejected 证据不得被静默擦除；
- 冲突只能由新的授权治理对象解决；
- 同一规范来源集合与固定 Profile 产生同一候选集合、治理视图与问题集合；
- partial 或 disputed 视图必须与 authoritative 视图区分。

延迟证据会改变当前来源集合，因此可以合法改变当前视图。只有在最终来源集合相同时，确定性才要求相同输出；异步流程不同阶段的输出不必相同。

4.4 冲突与验证处理

Reader **MUST** 按以下规则稳定处理异常：

条件	必需行为
Schema 或类型无效	保留来源用于诊断，排除在权威重建之外，产生验证问题
Digest 不匹配	保留为完整性失败证据，排除在完整权威集合之外
缺少签名	其他要求满足时允许 Core 处理，但不得声称认证完整性
签名无法验证	产生签名问题，不得进入认证视图
相同 ID、相同规范内容	投影时安全去重，同时保留每个贡献来源的身份
相同 ID、不同规范内容	全部变体从权威图隔离，产生严重重复 ID 问题
缺失引用	对象保留在 partial 视图，产生未解决引用问题
流序号缺口	标记流不完整，不推断缺失对象或迁移
重复流序号	保留冲突对象，标记受影响流不符合规范，状态保持 partial 或 disputed
时间戳冲突	不把时间戳作为权威，使用序号与 Profile 定义依赖
非法生命周期迁移	保留尝试，不改变权威状态，产生生命周期问题
未授权角色动作	保留尝试但不应用，产生授权问题
禁止环	隔离受影响子图并报告环，继续重建未受影响对象
并行矛盾复核	保留全部有效复核，直到出现授权解决对象

5. 威胁模型与信任假设

5.1 受保护属性

TPMA 旨在保护：对象对声明创建者与角色的归属；对象发布后修改的可检测性；生命周期与权限违规的可见性；冲突证据的保留；Runtime 中断后受治理工作的可重建性；执行、复核与批准职责的分离。

Core **MUST NOT** 把这些结构保证扩大解释为事实真实性、强身份认证或防篡改存储保证。

5.2 信任根

部署 **MUST** 标识其信任根，包括适用的：

- 身份提供方；
- 角色分配权限；
- Schema 与 Profile 发布权限；
- 规范化与完整性 Profile；
- 签名密钥与撤销来源；
- 存储访问控制；
- 时间来源；
- 人工批准方或治理机构。

如果信任根能够被同一攻击者静默改写，TMPA Core 的结构检查不能恢复更强保证。部署 **MUST** 明确说明其信任假设，并 **MUST NOT** 把未声明或未验证的基础设施属性表述为 TMPA Core 保证。

5.3 纳入考虑的威胁

符合规范的实现 **SHOULD** 考虑：身份冒充、未授权角色声明、对象篡改、有效对象重放、非法生命周期迁移、必需证据遗漏、首次发布时伪造结构正确但事实错误的证据、通过冲突对象进行双重陈述、受损工具或 Connector、导致越权动作的 Prompt Injection、证据删除或隐匿、时钟偏差与时间戳操纵、由错误/过期/对抗性审计结论触发的自动修复、扩大而非收敛权限的传递委托、能力组合形成越权结果、过期或撤销不足的委托证据、合法单步组合成非法路径，以及由同一模型、控制器、凭证、Host 或管理主体控制的名义独立复核者。

5.4 恶意参与者与存储表面受损

TMPA Core 不假设所有参与者诚实。实现 **MUST** 保留归属、冲突对象、被拒迁移与验证问题，使不当行为可以被发现或调查。

首次发布时结构正确但事实错误的证据

TMPA 区分首次发布时的伪造与发布后的篡改。恶意、受损或错误的参与者可能发布 Schema 有效、Digest 一致、生命周期合法、甚至签名正确的 REPORT、REVIEW 或 DECISION，但其中事实声明仍然错误。Core 验证可以建立结构有效性、连续性、声明权限与已发布字节完整性，不能由这些属性推导语义真实性。

事实保证需要与声明相适配的证据 Profile，例如工具回执、外部可验证输出、可复现执行、测试结果、独立数据源、在真正独立安全主体下的交叉复核或人工批准。执行者与复核者共享同一受损控制器、凭证、证据源或管理主体时，名义职责分离可能产生相关性伪造。TMPA 可以保留 Provenance、矛盾与后续更正，但不能检测隐蔽串谋，也不保证首次发布声明为真。

FCoP 把文件系统纳入协议边界，因此存在特定攻击面：有直接写权限的参与者可能把对象直接放入 _lifecycle/done/、修改已发布工件、删除证据、重放旧文件，或制造路径/事件不一致。文件存在本身 **MUST NOT** 被视为有效性证明。

Reader **SHOULD** 至少区分：

1. **未授权插入**：工件出现在生命周期位置，但没有有效创建者、角色分配、前驱状态或迁移记录；
2. **发布后变更**：内容与记录的 Digest 或签名不一致；
3. **状态证据分歧**：生命周期路径、迁移历史、引用与预期配对工件互不一致。

这些攻击只能在部署保护或独立验证身份绑定、完整性记录、只增事件与存储历史的范围内被检测。如果攻击者能够同时改写工件及全部可信完整性、身份与审计记录，本地文件系统视图无法建立真实历史。更强部署 **MAY** 增加受限写权限、只增或版本化存储、远程公证、透明日志、复制或密码学签名；这些属于最低 FCoP 文件 Profile 之外的部署控制。

TMPA Core **MUST NOT** 被描述为提供拜占庭共识。若身份提供方、角色权限、密钥注册表、可信存储边界和验证器均受损，TMPA 无法保证真实历史。需要拜占庭容错的部署 **MUST** 增加外部共识、复制、公证或透明日志机制。

Inspect-only 审计 Profile 可以缩小一次失败的影响：受损 Inspector 可以生成误导性发现或建议，但审计功能不直接修改受治理业务状态。这不是完整防护；建议的 Provenance、复核、批准与执行证据仍然 **MUST** 保留。

5.5 角色与身份分离层级

TMPA 区分逻辑责任分离与安全域分离：

1. **Prompt 级角色分离**：参与者使用不同角色指令，但可以共享 Runtime、凭证与存储权限；
2. **进程级身份分离**：不同 Agent/进程实例具有稳定 Runtime 身份与独立可归属 Session；
3. **凭证级分离**：使用可独立验证、限制和撤销的不同凭证、密钥或委托授权；
4. **Host 级隔离**：OS 账户、容器、Sandbox 或强制访问控制阻止参与者直接修改他人受保护证据；
5. **管理域分离**：独立组织、信任根、审计服务或透明系统降低单一管理员改写全部记录的风险。

Prompt 级分离可以支持流程清晰，但不是安全边界。进程级身份改善归属，但不能阻止共享凭证或跨进程存储修改。声称独立复核、认证责任或防篡改的部署 **MUST** 采用与威胁模型相称的凭证、Host 或管理域控制。

最低 FCoP Profile 下的文件名、文件所有者、角色文档或 Frontmatter sender 都只是**声明性归属**。只有验证执行进程、凭证/密钥、有效角色分配与受保护写入边界之间的绑定后，它才成为已验证归属。由同一模型、服务账户或无限制 Host 控制的多个逻辑角色 **MUST NOT** 仅因 Prompt 或文件名不同就被描述为独立安全主体。

已验证委托链还 **MUST** 区分委托主体、委托任务/意图、授予能力集合、权限衰减规则、时间有效性、执行次数或撤销条件，以及每次下游再委托。既有 TASK、REPORT 或角色标签不会自动授权新动作，除非有效身份与授权 Profile 明确认可其为当前委托证据。

5.6 AI Agent 身份

AI Agent **MUST NOT** 被视为自认证法律身份。FCoP 仍要求 Agent 获得自己能够读取的显式运行身份：角色、团队上下文、责任边界与当前工作范围。运行身份背后的权限来自人类或组织主体、部署身份、角色分配权限、Runtime 凭证与策略范围。

身份记录 **SHOULD** 分开表示：组织主体、人类授权者、Agent 实例、模型或 Runtime 版本、有效角色、委托权限、凭证或密钥 ID。这样可以避免把 Agent 动作只归属于借用的人类账户或服务账户。

5.7 安全声明

实现 **MUST** 明确声明其支持的保证层级：

声明	最低要求
文本可追踪性	持久规范对象与引用
篡改检测	对保留或可信完整性元数据执行确定性 Digest 验证
认证完整性	已验证签名与可信密钥绑定
授权执行	已验证角色分配与动作策略
语义声明验证	Core 之外的声明特定证据、可复现输出或独立领域验证
不可否认	TMPA Core 之外的法律与密码学 Profile
拜占庭韧性	外部共识或等价机制

实现 **MUST NOT** 声称超出实际部署控制的更强属性。

6. 生命周期与权限求值

6.1 必需注册表

实现 Profile 发布带版本的生命周期、角色与关系注册表。生命周期注册项包含：Profile ID 与版本、状态集合、初始与终止状态、动作集合、合法 from/action/to 元组、每项动作允许的角色、必需引用与前置条件、职责分离规则，以及任何授权重新打开或恢复规则。角色注册项包含：角色 ID、Assignment 对象类型、允许的文档类型和生命周期动作、Scope 维度、不兼容角色、分配权限方与撤销语义。关系注册项声明关系属于顺序、非顺序、必需或无环关系。

注册表字节是重建输入。因此，注册表版本与 Digest 属于 Reader 输入契约和一致性报告；修改注册表而保留原 ID 不会得到同一个固定 Profile。

6.2 迁移求值顺序

对于候选迁移 x、Profile P、规范候选集合 C 和当前重建状态 s，按以下固定顺序求值：

- EVALUATE_TRANSITION(x, s, C, P):
1. 验证对象 Schema、类型规则、身份与完整性
 2. 解析受治理工作项、主载体与生命周期 Profile
 3. 从已接受前驱证据重建唯一当前状态
 4. 验证 x.from 等于该当前状态
 5. 验证 (x.from, x.action, x.to) 是合法迁移元组

- 6. 解析有效角色 Assignment 并验证动作 Scope
- 7. 求值职责分离规则与授权例外
- 8. 解析必需引用、前置条件与证据
- 9. 赋予 valid、invalid 或 undetermined，并生成规范问题
- 10. 仅当迁移判断为 valid 时应用 x.to

已证明的违规——例如非法元组、已撤销权限、超出 Scope 的动作或禁止的角色组合——产生 invalid。证据缺失——例如 Assignment 不可用、前驱未解析、必需引用缺失或当前状态有歧义——产生 undetermined。只有 valid 迁移改变权威生命周期投影。

6.3 状态重建

对每个受治理工作项，Reader 在接受有效主载体后从生命周期 Profile 的初始状态开始，再按写者流序号和声明顺序依赖建立的偏序求值迁移对象。不得使用挂钟时间选择下一项迁移。

如果两个有效迁移候选消费同一来源状态，效果互不兼容，且不存在顺序关系或授权解决，则当前状态为 undetermined，视图为 disputed。Reader 保留两个分支，不选择最后到达者。终止状态保持终止，除非生命周期注册表显式定义授权恢复或重新打开迁移。

6.4 权限时间与撤销

Reader 针对适用于该动作的 Assignment 与撤销证据验证权限。created_at 本身不是可信授权时钟。进行时间敏感权限声明的 Profile 要定义用于判断 Assignment 是否有效的可信时间或序列证据。

证据证明权限无效，动作判断为 invalid；无法确定相关权限区间，动作判断为 undetermined。Profile 还声明撤销是前瞻性的，还是可以使已定义类别的早期动作无效；Reader 不得虚构追溯效力。

7. 三值治理逻辑

7.1 判断域

TMPA Core 的规范判断域为：

J = { valid, invalid, undetermined }

valid 表示固定输入和适用规则证明要求成立；invalid 表示它们证明存在违规或拒绝；undetermined 表示证据缺失、冲突或尚未解决，因而无法得出二值结论。undetermined 不是 invalid 的委婉说法，也不是默认接受。

7.2 原子分类规则

Reader **SHALL** 按下表分类原子检查：

条件	判断	主要视图原因
要求已证明满足	valid	authoritative
规则已证明被违反	invalid	rejected 或 quarantined
必需证据缺失或无法读取	undetermined	partial
多个有效候选互相矛盾且无授权解决	undetermined	disputed
必需人工决定尚未发布	undetermined	pending_human
可选签名在 Core 下缺失	Core 判断不变	unauthenticated 保证标记
认证 Profile 要求的认证无法建立	按已发布 Profile 为 undetermined 或 invalid	unauthenticated 或 quarantined

unauthenticated 是认证状态，而不是第四个治理判断值。要求认证完整性的 Profile 中，缺失认证证据产生 undetermined；已证明签名或身份无效产生 invalid。

7.3 组合规则

对于全部条件都必须满足的合取 ALL(a,b)，以及至少一个条件必须满足的析取 ANY(a,b)，Reader **SHALL** 使用下表：

a	b	ALL(a,b)	ANY(a,b)
valid	valid	valid	valid
valid	invalid	invalid	valid
valid	undetermined	undetermined	valid
invalid	valid	invalid	valid
invalid	invalid	invalid	invalid
invalid	undetermined	invalid	undetermined
undetermined	valid	undetermined	valid
undetermined	invalid	invalid	undetermined
undetermined	undetermined	undetermined	undetermined

必需依赖为 invalid 时，依赖的接受条件为 invalid；必需依赖为 undetermined 时，依赖结论 **SHALL** 保持 undetermined。互相矛盾的有效结论不会相互抵消，也不会变成 invalid；它们形成 undetermined 的争议结论。只有解决对象本身有效、已授权并显式引用所解决的冲突时，解决才改变结论。

Profile **MAY** 定义领域特定聚合，但必须发布真值表，且不得把缺失或冲突的强制证据直接映射为 valid。

7.4 判断与视图映射

判断是语义；视图状态解释运行原因。valid 映射为 authoritative。动作 invalid 映射为 rejected；证据或子图被排除时映射为 quarantined。undetermined 根据规范问题原因映射为 disputed、partial 或 pending_human。

一个主体有多个原因时，全部原因都保留在问题集合中。需要一个主视图标签时，按 quarantined → rejected → disputed → partial → pending_human → authoritative 选择。认证是独立保证状态，不形成第四个语义判断。

8. Reader 输入与输出契约

8.1 输入 Bundle

一次可复现 Reader 运行 **SHALL** 固定并记录以下输入：

- TPM Core 对象 Schema 版本与 Digest；
- 一致性 Profile ID、版本与 Digest；
- 类型、生命周期、角色、关系、完整性和规范化注册表的版本与 Digest；
- 有限来源候选多重集合，其中每个候选具有稳定 source_id、媒体类型、精确字节与字节 Digest；
- 声明的信任根与认证策略；
- Reader 实现标识与版本；
- 规范输出格式版本；
- 任何实现扩展，以及扩展是否影响规范语义。

只有以上输入相等的两次运行才能用于 C11 比较。环境特定 Locator、发现时间戳、日志顺序、内存地址与本地化诊断不是规范输入。

8.2 规范结果

Reader **SHALL** 输出至少包含以下字段的规范结果 Envelope：

```
{
  "core_version": "S1.0",
  "output_version": "1",
  "profile": {},
  "reader": { "id": "<id>", "version": "<version>" },
  "source_set_digest": "sha256:<hex>",
  "judgment": "valid | invalid | undetermined",
  "view_state": "authoritative | rejected | quarantined | partial | disputed | pending_human",
  "nodes": [],
}
```

```

"edges": [],
"issues": []
}

```

每个节点和边 **SHALL** 包含稳定 ID 与其来源对象 ID。每个问题 **SHALL** 包含稳定 issue_id 与 source_id；解析产生来源对象 ID 时还记录 source_object_id。节点 **SHOULD** 另记录规范 Digest、受治理工作项 ID、主载体 ID、类型、流位置、判断、视图状态与保留的来源 ID；边 **SHOULD** 记录关系与排序语义；问题 **SHALL** 记录代码和严重级别，并 **SHOULD** 记录受影响判断、规范规则与确定性参数。

Core 问题代码为：SCHEMA_INVALID、UNKNOWN_TYPE、INTEGRITY_MISMATCH、SIGNATURE_UNVERIFIED、DUPLICATE_ID_CONFLICT、PRIMARY_CARRIER_CONFLICT、STREAM_DUPLICATE_SEQUENCE、STREAM_GAP、AUTHORITY_UNDETERMINED、AUTHORITY_DENIED、SOD_VIOLATION、LIFECYCLE_UNDETERMINED、ILLEGAL_TRANSITION、MISSING_REFERENCE、PROHIBITED_CYCLE、UNRESOLVED_CONFLICT、CLAIM_EVIDENCE_MISSING、ACCEPTANCE_UNDETERMINED、HUMAN_APPROVAL_REQUIRED、CHILD_WORK_OPEN、RECIPROCITY_MISSING 与 STATE_EVIDENCE_CONFLICT。Profile **MAY** 增加命名空间化代码，但不得重定义 Core 代码。

8.3 规范化与排序

source_set_digest 按声明输出 Profile 从 (source_id, byte_digest) 对的确定性排序列表计算。节点按 (id, source_object_id) 排序；边按 (source_id, relation, target_id, id) 排序；问题按 (severity, code, object_id, relation, target_id, issue_id) 排序，其中严重级别顺序为 critical、error、warning、info。元组中不存在的字段按空字符串处理。

issue_id **SHALL** 由 (code, object_id, relation, target_id, profile_digest) 的规范编码确定性导出。人类可读消息、Stack Trace、本地路径与执行时间戳排除在规范等价性之外。对象键、主体、保留的来源 ID、节点、边和问题在 Profile 定义规范化后按 Unicode 码点排序；依赖 Locale 的排序不得作为规范排序。

对于相同固定输入，规范结果序列化必须字节稳定。非规范日志与用户界面排序可以变化，但不得改变用于 C11 的结果 Envelope。

8.4 失败与部分输出

即使某些候选格式错误或子图无效，Reader 仍返回规范结果与问题集合。只有固定 Profile、Schema、注册表 Bundle 或输出规范化契约无法加载或验证时，Reader 才可使整个调用失败。此类调用失败不同于 invalid 治理判断，并作为一致性运行错误报告。

9. TMPA Core 规范

9.1 规范语言

术语 **MUST**、**MUST NOT**、**SHALL**、**SHALL NOT**、**SHOULD**、**SHOULD NOT** 与 **MAY** 定义一致性要求。**MUST**、**MUST NOT**、**SHALL**、**SHALL NOT** 表示强制要求。

本章之外的说明性示例、实现观察与未来工作陈述不产生额外 Core 要求，除非命名一致性 Profile 明确把它们纳入规范。

9.2 对象要求

每个治理对象 **MUST**：

- 符合 TMPA Core Schema 与已发布的文档类型定义；
- 在其治理域内具有全局唯一 ID；
- 恰好具有一个声明创建者身份；
- 标识恰好一个责任角色；
- 标识一条写者流和一个正整数序号；
- 标识一个文档类型；

- 标识一个受治理工作项和恰好一个主载体 ID；
- 标识一个生命周期 Profile 与声明状态；
- 包含规范文本内容；
- 包含可为空的 references 数组；
- 包含完整性证据。

符合规范的 Validator **SHALL** 强制验证 created_at 声明的 date-time 格式；只把格式当作注释不足以通过 C01。

记录生命周期迁移的对象类型 **SHALL** 包含完整的 from、action、to 元组。非迁移类型 **SHALL NOT** 使用该元组产生隐式状态变化。

已发布对象 **SHALL** 不可变。更正、拒绝、取代、回滚或解决 **SHALL** 创建新的对象或迁移记录，并 **SHALL** 保留旧证据。

Schema 有效性 **SHALL NOT** 被解释为 ID 唯一、角色授权、生命周期合法、引用有效、Digest 正确或身份已认证的证明。

面向任务的 Profile **SHALL** 为每个受治理工作项定义一个稳定主载体 ID。后续接受、报告、复核、决策、更正与恢复对象 **SHALL** 引用该载体或 Profile 定义的后继关系，不得创建同一任务的歧义可变副本。

每个已发布治理对象 **SHALL** 只有一个写者。其他参与者 **SHALL** 通过新的可归属对象或迁移记录响应，并 **SHALL NOT** 修改其他写者已发布对象的内容。

9.3 类型注册要求

符合规范的实现 **SHALL** 发布文档类型注册表。每个类型定义 **SHALL** 指明允许创建者角色、必填字段、允许引用关系、适用生命周期 Profile 与验证规则。

注册表 **SHALL** 具有稳定 ID、版本与字节 Digest。Reader **SHALL** 把结果绑定到该精确注册表修订。每个类型定义还 **SHALL** 指明该类型是否要求生命周期迁移元组。

文档 **SHALL NOT** 同时充当自己的独立复核或批准，除非实现 Profile 允许一个已记录例外，而且例外标识批准权限。

9.4 角色要求

角色声明 **SHALL** 根据对相关对象与动作有效的权威角色分配进行验证。参与者 **SHALL NOT** 在有效角色范围之外执行受保护动作。

声称职责分离的部署 **SHALL** 定义并执行同一受治理结果上的不兼容角色组合。

角色分配、撤销、委托与职责分离例外 **SHOULD** 自身表示为治理对象。

角色与权限求值 **SHALL** 遵循第 6.2 节的顺序。已证明拒绝或超出 Scope 的动作 **SHALL** 为 invalid；分配证据缺失或有歧义 **SHALL** 为 undetermined。

9.5 流要求

每条流 **SHALL** 具有稳定流 ID。

每个已发布对象 **SHALL** 恰好属于一条写者流。符合规范的 Profile **SHALL** 保留该写者的局部发布顺序，并 **SHALL NOT** 要求多个写者共同编写一个可变对象。

独立流 **MAY** 异步推进，**SHALL NOT** 被要求同步前进。当 Profile 未定义依赖时，一条流暂时没有新对象 **SHALL NOT** 阻止无关流继续推进。

序号 **SHALL** 为正整数，并 **SHALL** 在流内唯一。

Reader **SHALL** 把重复序号报告为流完整性错误，并 **SHALL** 保留全部冲突候选用于检查。

Reader **SHALL** 报告序号缺口，**SHALL NOT** 发明缺失对象、推断其内容或用时间戳替代缺失序号位置。

墙上时钟时间戳 **SHALL NOT** 成为唯一权威排序机制。

除非适用 Profile 定义跨流因果、生命周期或依赖关系，否则 Reader **SHALL NOT** 推断不同流对象之间的权威顺序。没有此类关系的对象 **SHALL** 在治理图中保持并发或不可比较。

9.6 生命周期要求

每个生命周期 Profile **SHALL** 定义状态、初始状态、终态、动作、合法迁移、授权角色、前置条件与必需证据。

每个生命周期 Profile **SHALL** 具有稳定 ID、版本与字节 Digest。Reader **SHALL** 按第 6.2 节定义的顺序验证迁移，按第 6.3 节重建状态，并只应用判断为 valid 的迁移。无法重建唯一当前状态时，候选迁移 **SHALL** 为 undetermined，且 **SHALL NOT** 改变权威状态。

非法或未经授权的迁移 **SHALL NOT** 改变权威生命周期状态。

尝试迁移 **SHALL** 通过拒绝、ISSUE、告警或 Profile 规定的等价记录保持可观察；只有部署声明的威胁模型确实无法捕获该尝试时才可例外。

终态或归档操作 **SHALL** 保留重建该状态如何到达所需的对象与迁移证据。

9.7 引用要求

每项引用 **SHALL** 标识关系类型和目标对象 ID。

Profile **SHALL** 定义哪些引用类型产生顺序依赖、哪些是非排序链接，以及哪些关系类别必须无环。

关系注册表 **SHALL** 具有稳定 ID、版本与字节 Digest，Reader 结果 **SHALL** 标识所用的精确修订。

缺失目标 **SHALL** 被报告。引用对象 **MAY** 保留在 partial 视图，但缺失依赖 **SHALL NOT** 被视为已满足。

Reader **SHALL** 隔离禁止环的受影响子图，不得静默删除边或任意选择顺序。未受影响的有效对象 **SHOULD** 继续可重建。

9.8 完整性要求

规范化与 Digest 算法 **SHALL** 由版本化完整性 Profile 声明。

完整性 Profile **SHALL** 定义 Digest 以及适用签名覆盖的精确字段或字节，并 **SHALL** 定义如何排除或规范化 digest、signature_algorithm、key_id 与 signature，避免自引用。

Profile 还 **SHALL** 定义字符编码、Unicode 规范化、换行、字段顺序、空白与转义、数字和时间戳表示、缺失与 null 的区别、文本正文处理、附件 Hash、扩展字段处理，以及绑定到规范形式的 Schema/Profile 版本。未知扩展字段 **SHALL** 被确定性地纳入覆盖表示或被拒绝；它们 **SHALL NOT** 在影响权威语义的同时被静默排除在完整性保护之外。

Reader **SHALL** 在接受对象为完整对象之前重算并验证 Digest。Digest 不匹配 **SHALL** 被报告；对象 **SHALL NOT** 进入完整权威对象集合，但来源 **SHALL** 保留为失败证据。

存在签名元数据时，signature_algorithm、key_id 与 signature **SHALL** 作为完整组提供。Reader **SHALL** 在声称认证完整性前验证签名、密钥状态与身份绑定。

TPMA Core 允许没有签名。无法验证的签名 **SHALL NOT** 被视为认证完整性的有效证据。

部署 **SHALL NOT** 把 Schema、Digest、签名、角色授权或生命周期合法性表述为对象事实声明真实的证明。语义保证需要 Core 之外声明的证据与验证 Profile。

部署 **SHALL NOT** 仅因对象带有同处存储的 Digest，就声称能够抵抗同时修改内容与未锚定完整性元数据的攻击者。此类声明要求认证完整性、外部锚定 Digest、可信存储或等价控制。

9.9 聚合与 Reader 重建要求

符合规范的来源聚合器 **SHALL** 保留来源身份与内容，在不使用枚举顺序作为治理顺序的情况下发现候选，并为 Reader 生成确定性规范候选集合。它 **SHALL NOT** 静默解决冲突、发明缺失对象，或把传输/文件系统到达顺序转换为权威流程顺序。

对于同一规范候选集合与固定规则 Profile，治理 Reader **SHALL**：

- 对每种输入排列产生相同的规范偏序图/视图与问题集合；
- 保留流内顺序、Profile 定义的跨流关系和不可比较对象之间的并发；
- 原样保留来源对象；
- 保留有效冲突对象，直至出现授权解决对象；
- 把 Schema 无效和 Digest 无效对象排除在权威对象集合之外，同时保留诊断证据；
- 报告重复 ID、序号缺口、重复序号、非法迁移、越权动作、缺失引用、禁止环与完整性失败；
- 为每项受治理结论输出 valid、invalid 或 undetermined 三者之一，并保留形成该判断的原因；
- 在适用时区分 authoritative、partial、disputed、quarantined 与 unauthenticated；
- 对一致性问题与序列化视图元素使用确定性排序。

固定规则 Profile **SHALL** 包含第 8.1 节列出的全部输入。Reader **SHALL** 输出第 8.2 节定义的 Envelope，使用其中定义的 Core 问题代码与 ID，并应用第 7.3 节的三值组合规则。规范输出排序 **SHALL** 遵循第 8.3 节。

确定性拓扑序列或显示 Tie-break **SHALL NOT** 被解释为治理决定、真实性优先级或新增跨流顺序。

Reader **SHALL NOT** 使用输入到达顺序、文件系统枚举顺序或墙上时钟顺序解决治理冲突。

若依赖对象为 undetermined，依赖该对象的结论 **SHALL** 保持 undetermined，直至授权解决对象满足适用 Profile。视图分类用于解释判断原因，**SHALL NOT** 替代或扩展三个语义值。

9.10 恢复要求

替代参与者 **SHALL** 能够从持久治理对象与适用 Profile 判断：

- 当前 authoritative 或显式 partial 生命周期状态；
- 责任角色；
- 未解决要求；
- 相关结果、复核、批准与拒绝；
- 完整性、权限、顺序、引用与验证问题。

恢复 **SHALL NOT** 要求访问前任参与者的隐藏思维链。

治理对象中未表示的执行上下文 **MAY** 不可获得；这种缺失 **SHALL** 被报告，不得被猜测。

9.11 治理闭环、声明与人工控制要求

面向工作的 Profile **SHALL** 定义报告、问题、拒绝、取消、后续工作、复核、接受、父子派生、人工批准与归档授权的关系语义。

生命周期状态与业务验收 **SHALL** 分别重建。终态、done 标签、物理归档或执行者完成声明 **SHALL NOT** 单独建立业务完成。缺少有效接受证据时，Reader **SHALL** 输出 ACCEPTANCE_UNDETERMINED。

治理裁决对象 **SHALL** 与生命周期待复核阶段保持正交。执行报告 **SHALL NOT** 充当自己的独立复核；治理 REVIEW **SHALL NOT** 取代必需回执。

每项完成、失败、恢复或验收 Claim **SHALL** 标识稳定 Claim ID、Predicate、Subject 与证据对象 ID 集合。缺失必需证据 **SHALL** 输出 CLAIM_EVIDENCE_MISSING，且相应结论 **SHALL** 为 undetermined。

每个已接收工作 **SHALL** 最终具有报告、问题、拒绝、取消或后续工作回执。Reader 在 Profile 要求闭环但未找到回执时 **SHALL** 输出 RECIPROCITY_MISSING，不得把沉默解释为成功。

子工作 **SHALL** 显式标识父工作。父工作有未结束、未处理阻塞或缺少回执的子工作时，父级完成或验收 **SHALL** 输出 CHILD_WORK_OPEN 并保持 undetermined。

需要人工批准的风险对象 **SHALL** 保持 undetermined / pending_human，直至 Reader 验证一个独立的人工批准对象：其对象类型与关系得到风险策略允许，创建者具有指向允许批准角色的有效 Assignment，并且在 Profile 要求独立参与者时，创建者不同于风险对象创建者。Agent 自签批准、只有角色标签但缺少 Assignment 证据、缺失批准、错误对象类型或越权批准 **SHALL NOT** 满足该要求，并 **SHALL** 输出 HUMAN_APPROVAL_REQUIRED 或适用权限问题。

Profile **SHALL** 发布失败类型与恢复动作注册表。失败与恢复对象 **SHALL** 引用受影响工作；恢复对象还 **SHALL** 引用触发失败。失败 **SHALL NOT** 被成功回执覆盖或隐藏。

当前状态观测与迁移历史冲突时，Reader **SHALL** 输出 STATE_EVIDENCE_CONFLICT，保留两种来源，并阻止冲突状态成为唯一 authoritative 结论。

巡检与治理告警 **MAY** 产生发现和建议方案，但它们 **SHALL NOT** 在没有独立执行证据时被解释为已实施修复、生命周期迁移或业务裁决。

10. 一致性与可测试性

第 9、10 节保留历史综合 TMPA Draft V1.0 使用的条款 ID，使一致性报告与 Fixture 可以在 Architecture Paper、Core Specification 与 Implementation Case Report 之间引用相同的规范基础。该历史来源不具有当前编辑权威；本 GitHub Core Specification 是这些条款的唯一现行规范性来源。

10.1 一致性层级

TMPA 定义三个一致性层级：

- TMPA Core Conformance**：实现持久文本消息与状态对象、主载体规则、单写者流、异步多流推进、确定性聚合与治理重建、类型规则、角色、生命周期、完整性验证和恢复要求；
- FCoP Profile Conformance**：通过有文档记录的投影满足 TMPA Core 与已发布 FCoP 协议的命名、生命周期、原子迁移、路由及证据规则。某个 FCoP 参考实现 Package 的测试通过只构成实现证据；它既不是“安装协议”，也不足以单独建立本层级一致性；
- Authenticated Governance Conformance**：满足 TMPA Core，并通过可信签名、密钥与授权 Profile 验证创建者身份。

任何层级都不认证参与者事实声明的语义真实性。Authenticated Governance Conformance 可以建立哪个已验证主体发布了授权对象，但声明正确性仍取决于适用证据、复核、工具证明或领域验证 Profile。

一致性声明必须绑定到特定实现版本、Profile 版本、Fixture 语料库与结果集合。产品身份、仓库所有权、Package 发布或 Demo 可访问性本身都不建立一致性。

10.2 C01–C14 必需测试

TMPA Core 一致性套件 **SHALL** 包含 C01–C14。每项结果 **SHALL** 标识其规范基础并保留复现裁决所需的实际输出。

ID	测试	规范基础	通过标准
C01	Schema 验证	4.1、9.2、9.3、9.11	缺少必填字段、Core 类型或版本错误、字段被禁止、迁移元组格式错误、签名组不完整、Claim 缺字段、风险枚举错误或声明 date-time 无效的对象被排除在权威集合之外，并确定性地产生 SCHEMA_INVALID
C02	主载体、工作派生与单写者不可变性	9.2、9.11	一个稳定任务载体保持可识别；父子工作关系可回读且不被 Thread 替代；其他写者不能替换或共同编辑已发布对象；更正/取代通过新对象表达
C03	重复对象 ID 与来源可追踪性	9.2、9.9	相同 ID、不同规范内容的候选被保留并隔离，产生确定性严重冲突；字节相同的观测投影为一个节点，同时保留每个贡献来源 ID
C04	串行流连续性与异步推进	9.5、9.9	每个写者保持局部序号；重复与缺口被报告；无关流可独立推进；不发明缺失对象；同一集合的到达顺序不改变结果
C05	角色权限	6.2、9.4、9.9	超出已验证角色 Scope 的动作产生 AUTHORITY_DENIED 与 invalid；Assignment 证据缺失或有歧义产生 AUTHORITY_UNDETERMINED 与 undetermined；两者都不被应用
C06	生命周期合法性、状态证据与业务验收分离	6.2–6.4、9.6、9.9、9.11	未定义迁移产生 ILLEGAL_TRANSITION；当前状态或前置证据不足产生 LIFECYCLE_UNDETERMINED；与重建状态冲突的观测产生 STATE_EVIDENCE_CONFLICT；完成状态缺少独立接受证据产生 ACCEPTANCE_UNDETERMINED；均不得虚构业务完成
C07	职责分离与人工控制	9.3、9.4、9.11	同一身份不能执行并独立复核同一结果；自签批准、未绑定 Assignment 的角色标签及错误批准对象类型继续保持 pending；只有独立的允许对象由已分配、经 Profile 授权且在要求时相互独立的批准人发布，才满足门禁
C08	完整性篡改	9.8、9.9	修改被覆盖内容但保留原完整性元数据会导致 Digest 验证失败；对象保留为失败证据但排除在完整权威集合之外
C09	缺失引用与声明证据	9.7、9.9、9.11	未解决必需目标产生 MISSING_REFERENCE；完成 Claim 的证据缺失产生 CLAIM_EVIDENCE_MISSING；依赖或声明不被视为已满足

ID	测试	规范基础	通过标准
C10 C11	禁止环 聚合与重建确定性	9.7、9.9 8、9.9	受影响的禁止环子图被隔离并报告，未受影响有效对象继续可重建 同一来源集合与完整固定输入 Bundle 的所有测试枚举、延迟交付排列和聚合顺序，都产生字节等价的规范结果 Envelope、治理图与问题集合；Unicode 码点排序不依赖 Locale，无关跨流对象保持不可比较 矛盾的有效复核保持可见且 disputed，直至出现新的授权解决对象
C12 C13	冲突保留 恢复与父子闭环	9.9 9.10、9.11	全新 Reader 从持久治理证据重建责任、生命周期、未解决依赖、失败/恢复及父子关系；父工作有开放子工作时产生 CHILD_WORK_OPEN，无需隐藏 Runtime 上下文
C14	验收后的终态历史保留	9.2、9.6、9.11	只有获得所需验收与归档授权后进入终态/归档，并保留重建任务、报告、复核、接受与迁移历史所需的全部对象

测试是行为测试。实现 **MAY** 使用不同存储、索引或执行机制，但可观察一致性结果 **MUST** 满足相同标准。

10.3 可执行测试用例契约

每个可执行测试用例 **SHALL** 发布机器可读 Manifest，包含：

- 稳定的 test_case_id 和恰好一个 C01–C14 criterion；
- Core、对象 Schema、输出 Schema、Profile 和注册表的版本与字节 Digest；
- 显式前置条件；
- 包含 source_id、仓库相对 path、媒体类型与字节 Digest 的来源 Fixture 清单；
- 包含稳定断言 ID、Target、Operator、Expected Value 与 Mandatory 标记的断言；
- 预期规范结果 Digest；
- Runner ID、命令、执行环境，以及任何排列方法或 Seed；
- stdout、stderr、规范输出与支持证据的仓库相对路径。

Runner **SHALL** 保留精确输入 Manifest、规范结果、退出状态、stdout、stderr 与执行环境身份。测试 **SHALL NOT** 依赖未固定的网络响应、墙上时钟顺序、文件系统枚举顺序或未声明的可变状态。

```
{
  "test_case_id": "C06-illegal-transition-001",
  "criterion": "C06",
  "core_version": "S1.0",
  "inputs": [{"source_id": "transition-1", "path": "fixtures/C06/transition-1.json", "media_type": "application/json", "byte_digest": "sha256:<hex>"}],
  "assertions": [{"id": "state-unchanged", "target": "/nodes/work-1/state", "operator": "equals", "expected": "active", "mandatory": true}],
  "expected_result_digest": "sha256:<hex>",
  "runner": {"id": "tmpa-conformance", "version": "<version>", "command": "<command>"}
}
```

10.4 裁决算法与一致性声明

Runner **SHALL** 为每项标准赋予恰好一个裁决：

- **PASS**：全部强制断言都已执行且通过；
- **FAIL**：至少一个强制断言已执行且失败；
- **PARTIAL**：至少一个强制断言已执行且通过、没有失败，并且至少一个未执行；
- **NOT RUN**：没有强制断言执行，或前置条件阻止了求值。

基础设施失败 **SHALL** 另记为 run_state: error，并产生 NOT RUN，而不是 PASS。聚合优先级为 FAIL、PARTIAL、NOT RUN、PASS：任一 FAIL 使聚合为 FAIL；没有 FAIL 时，任一 PARTIAL 使聚合为 PARTIAL；两者均无时，任一 NOT RUN 使聚合为 NOT RUN；只有全部 PASS 才得到 PASS。

只有 C01–C14 针对同一固定输入 Bundle 全部 PASS，且完整证据 Package 已发布，产品才 **MAY** 声明 **TMPA Core S1.0 Conformance**。“未观察到失败”、PARTIAL、NOT RUN、旧版 Core 结果或未发布结果 **SHALL NOT** 被表述为完整 S1.0 一致性。

specified、implemented、demonstrated 与 independently adopted 描述证据成熟度，**SHALL** 与测试裁决分开报告。作者演示不建立独立采用。

```
{
  "core_version": "S1.0",
  "implementation": {"id": "<id>", "version": "<version>"},
  "criteria": [{"id": "C01", "verdict": "PASS", "manifest_digest": "sha256:<hex>", "result_digest": "sha256:<hex>"}],
  "aggregate_verdict": "PASS | FAIL | PARTIAL | NOT RUN",
}
```

```
"evidence_level": "specified | implemented | demonstrated | independently_adopted"
}
```

10.5 Fixture 与结果报告

一致性 Package **SHOULD** 发布：

- 有效 TMPA Core 对象 Fixture；
- 从已发布 Schema 派生的有效 FCoP TASK、REPORT、ISSUE 与 REVIEW Fixture；
- 无效 Schema 与格式 Fixture；
- 非法及未经授权的迁移 Fixture；
- 越权角色与职责分离 Fixture；
- 损坏 Digest 与签名 Fixture；
- 重复 ID、重复序号与序号缺口 Fixture；
- 缺失引用与禁止环 Fixture；
- 并行冲突复核 Fixture；
- 受控中断与恢复快照；
- 终态历史或归档保留 Fixture；
- 预期的规范聚合候选集合、重建流程/治理图或视图，以及问题集合。

每个可执行 Fixture 集合 **SHOULD** 标识：

- TMPA Schema 版本；
- Profile 与规则集版本；
- Reader 实现与版本；
- 规范化 Profile；
- 输入对象 ID 与 Digest；
- 预期接受、partial、disputed、quarantined 与 rejected ID；
- 预期规范视图与问题集合输出；
- 适用时的排列方法、Seed 与排列数量；
- Runner 命令、执行日期与结果。

只有当聚合器产生预期规范候选集合，而且治理图/视图及问题集合的规范序列化对同一最终来源集合的每个测试枚举和交付排列均与预期 Fixture 一致时，Pipeline 才通过 C11。非规范日志或内部数据结构顺序不同不构成失败，除非它改变规范输出。

10.6 合规映射

TMPA 提供技术控制，不自动提供法律认证。部署 **MAY** 把 TMPA 字段与测试映射到外部要求，包括组织问责、日志与保留、人工监督、身份与授权、职责分离、事件调查和证据完整性。

Crosswalk **SHALL** 标明每项外部要求是 fully supported、partially supported、unsupported 还是 outside TMPA scope，并 **SHALL** 标明映射依赖的外部身份、策略、保留与安全系统。

全球互操作 Profile 与特定司法辖区合规 Profile 是不同交付物。例如，把 FCoP 工件映射为 A2A Task 属于互操作问题；把 TMPA 证据映射到国家或行业法规属于合规问题。二者 **MAY** 共享治理对象，但 **SHALL NOT** 互相作为成立依据。

11. Profile、版本与出版规则

11.1 Core 与 Profile 分离

TMPA Core 定义可移植治理语义。Profile **MAY** 增加文档类型、生命周期、存储映射、身份绑定、完整性策略或应用规则。Profile **MUST** 标识版本，**MUST** 声明其 Core 一致性层级，并且在继续声称相应层级时 **MUST NOT** 削弱 Core **MUST**。

Profile 特定工件不会自动成为规范 Core 对象。Profile **MUST** 定义从来源工件到来源候选、规范对象、治理图节点/边及问题集合条目的确定性投影。

11.2 版本

改变必填字段、权限语义、生命周期合法性、规范化、Reader 输出、问题分类或 C01–C14 通过标准的修改，需要新的 Core 版本。不会改变可观察行为的编辑澄清 **MAY** 保留当前 Core 版本，但 **SHOULD** 记录于 Changelog。

一致性报告 **MUST** 标识精确 Core 版本、Profile 版本、Reader 实现、规范化 Profile、Fixture、来源修订与执行环境。

11.3 出版与证据边界

发布规范建立 ****specified**（已规定）****证据等级**。可执行代码可以为已测试路径建立 **implemented**（已实现）；有界运行可以建立 **demonstrated**（已演示）。若不存在外部实现、独立重跑或外部组织实际采用，以上均不能建立 ****independently adopted**（已被独立采用）****或独立验证**。

首个作者生成的 C01–C14 语料库作为独立经验工件维护，不嵌入本 Core Specification。产品裁决与案例证据属于 Implementation Case Report；规范标准的含义只由本文档定义。

所有规范修订 **MUST** 直接进入 joinwell52-AI/joinwell52 的本 GitHub 文档，并由 Git Commit 表示正式版本历史。Architecture Paper、Implementation Case Report、网站文案或外部副本 **MUST NOT** 覆盖或静默重定义 Core 条款。

11.4 S1.0 发布与证据记录

2026-08-10 的取证候选审查把中英文文档、四份机器可读 Schema、生命周期 Profile、规范化 Profile、Reference Reader 与 C01–C14 Fixture 冻结为同一个版本输入 Bundle，对应提交为 942cbb097eb3d662662f96a2269818ec9d7ca2ed。S1.0 在不增加新治理概念的前提下，把已经审查的 S0.6 规范设计提升到正式版本线。

作者生成的 S1.0 Reference Reader 通过全部十四项 S1.0 Fixture。冻结候选提交中的产品基线继续保留为 NOT RUN，不得改写。后续精确版本外部登记记录 CodeFlowMu V1.8.0 提交 c1e1f724293e8048fc3a956b6f6df8cf83f54830 使用产品 GovernanceReader.readSync 对该 Bundle 执行，结果为 **14 PASS / 0 PARTIAL / 0 NOT RUN / 0 FAIL**。输入 Bundle 摘要为 sha256:f98764987760cdc8ac356b1265fc98485f33345e7d6ffc8575ccb059ddd34daa，结果摘要为 sha256:0f0f642449db1853371861751a7a8ea36dce00013f53e32012a5e4dae45f4c39。

V1.8.0 证据包 SHA-256 为 9b92b06c3e46c8019362f55075be4ed066cb7a9c0d9859945b6c3b7de8840d04。它只建立该精确产品修订与输入 Bundle 的作者运行已演示行为，不建立独立验证、第三方认证、语义真实性、普遍一致性、消除幻觉或独立采用。I0.8 与 CodeFlowMu V1.6.0 证据包继续作为前序 S0.6 证据，不得改标。

11.5 S0.5 FCoP 派生历史基线

S0.5 从完整、版本固定的 FCoP 协议来源集合中派生生命周期状态/业务验收分离、父子工作、完成声明、角色能力分层、风险与人工批准门、互惠闭环、失败/恢复动作、巡检发现与漂移处理。FCoP 仍是协议与参考 Profile；fcop 与 fcop-mcp Python Package 仍是参考实现，不是协议本身。S0.4/I0.5、S0.5/I0.6 与 S0.5/I0.7 的精确历史含义由 Git 历史及其已发布证据包保留。

附录 A：历史来源可追踪性（说明性）

Core Specification 内容	历史来源章节	当前处理
术语与表示阶段	1.5	保留并独立维护
治理对象、角色、生命周期、流、聚合、完整性	4.1–4.7	保留并形成 Core 架构定义

Core Specification 内容	历史来源章节	当前处理
规范 Schema、编码、Reader 算法、冲突处理	6.1-6.2、6.5-6.6	保留；FCoP 映射与产品示例移入实施报告
威胁模型与安全边界	第 8 章	保留并形成 5.1-5.7
规范 Core 条款	第 9 章	保留 9.x 历史标识
一致性层级与 C01-C14	10.1-10.2	保留历史标识
Fixture 与结果报告	10.5	保留；当前产品基线移入实施报告
合规映射边界	10.6	保留

Architecture Paper **MAY** 总结本规范，但不得重定义其含义。Implementation Case Report **MAY** 按条款提供证据，但不得改变条款含义。历史综合草稿仅用于说明来源，不是当前编辑或规范权威；当前 S1.0 及后续规范版本只在本 GitHub Core Specification 中维护。

附录 B：FCoP 来源交叉映射（说明性）

S1.0 关注点	固定版本的 FCoP 来源	TPMA Core 处理
协议对象、文档与事件词汇	spec/fcop-v3-spec.md 与 spec/fcop-v3-spec.zh.md；仓库 Tag v3.2.5	投影为治理对象、类型化引用、写者流与保留来源的 Reader 输入
角色边界与协作周期规则	AGENTS.md, Rules 版本 3.2.5	声明能力与强制权限分离；验收与职责分离裁决要求可归属证据
机器可读载体与验证	spec/schemas/	为 S1.0 对象/Profile Schema 提供输入，但不替代 TPMA Core Schema 验证
生命周期、原子迁移、恢复与审计决策	FCoP Specification 与适用 ADR	形式化为生命周期状态、业务验收、失败/恢复动作、巡检发现及确定性历史重建
父子工作派生与闭环	FCoP v3.2.5 parent 协议表面	表示为 governed_work.parent_id、父子汇总与 CHILD_WORK_OPEN
可执行软件	fcop 与 fcop-mcp Package	只作为 FCoP 参考实现；Package 测试是实现证据，不是协议本身
下游采用	CodeFlowMu 及有界的 WP-13 证据	只作为 Implementation Case Report 中的应用证据，不作为理论证明或协议定义；小典 AI 仅保留为作者报告的历史谱系，不纳入评估证据

本表只用于可追踪，不构成引用即纳入。FCoP 与 TPMA 抽象不同时，由本 Core Specification 控制 TPMA 含义；应用或参考实现偏离协议来源时，应把偏差报告为实现证据，不得静默改写任一规范。