可編譯世界：程式執行作為世界狀態差分

The Compilable World: Program Execution as Governed World-State Difference

v0.1 / 2026-07-27 / Neo.K with Aletheia / 初版完成

https://thisoneisneok.com/html/papers/pu-2-05.html

SECTION

The Compilable World: Program Execution as Governed World-State Difference

論文編號： PU-2-05

系列：「程式宇宙書系」第 2 冊《程式語言的本質》

作者： Neo.K with Aletheia

機構： EveMissLab／一言諾科技有限公司

版本： v0.1

日期： 2026 年 7 月 27 日

SECTION

摘要

前四篇依序建立：程式語言不等於文字；符號應被理解為受治理算子；程式語言具有語法、語意與效果三層；自然意圖需要先進入意圖中介表示，再生成多重可執行投影。本篇進一步處理最終執行問題：當一個已驗證的意圖 IR 取得執行權後，Runtime 真正做的事情是什麼？

本文提出：

> 【Program Execution
> ≠
> Instruction Consumption】

並將程式執行重新定義為：

> 在明示世界模型、權限、規則與證據條件下，提出、驗證、提交、觀測並核對一項世界狀態差分。
> 形式上：

> 【CompileWorld
> :
> (
> W⊠,
> mathcal I_R,
> Γ,
> P
>)
> →
> <
> widehatΔ W,
> Plan,
> Obligations,
> Guards
> >】

其中：

- W⊠：執行前世界狀態；
- mathcal I_R：已結構化意圖；
- Γ：語意、型別、規則與 Runtime 環境；
- P：權限與委任；
- widehatΔ W：預測世界差分；
- Plan：可執行計畫；
- Obligations：證據、補償、通知與人類決策義務；

- Guards：狀態、不變量、資源與安全守衛。

  真正提交則為：

> 【CommitWorld
> :
> (
> W⊠,
> widehatΔ W,
> Plan,
> P
>)
> →
> <
> W_(t+1),
> Events,
> Evidence,
> Residual
> >】

本文核心命題是：

> 【widehatΔ W
> ≠
> Δ W】

預測差分不等於實際世界差分。模型所預測的狀態變更、外部效果與受影響主體，可能因：

- 世界狀態已變；
- 外部服務失敗；
- 物理設備未響應；
- 訊息重複或延遲；
- 權限被撤銷；

- 規則版本改變；
- 人類拒絕；
- 模型遺漏；

  而與實際結果不同。

  本文因此建立世界提交六階鏈：

> 【Propose
> →
> Validate
> →
> Authorize
> →
> Commit
> →
> Observe
> →
> Reconcile】

其中：

- Propose：由 IR 建立候選世界差分；
- Validate：檢查前置狀態、不變量、版本、資源與規則；
- Authorize：確認主體、委任、批准與作用域；
- Commit：實際改變權威數位狀態或發出外部作用；
- Observe：收集事件、回執、感測與人類回應；
- Reconcile：比較預測與實際差分，處理未知、部分成功與殘差。

  本文將可編譯世界表示為：

> 【W
> =
> <

其中：

- E：Entities，具有身份的世界實體；
- S：States，權威狀態與局部狀態；
- R：Rules，合法轉移與不變量；
- A：Actors，行動者、批准者與受影響者；
- B：Boundaries，責任、權限與系統邊界；
- H：History，事件、版本與承諾歷史；
- X：Externalities，外部數位、制度與物理世界；
- O：Observability，可取得證據與未知範圍。

  一個世界能被編譯，不表示世界可被完全計算。本文所謂「可編譯」，是指世界中至少有一部分狀態、規則、權限、效果與證據可被結構化，使候選差分能在提交前被檢查，在提交後被觀測與核對。

  本文提出：

> 【Compilable
> ≠
> Fully Predictable】

外部世界、社會制度與物理環境具有不可完備觀測、延遲、他者意志與隨機性。可編譯世界的成熟性不在於消除未知，而在於把未知、不可觀測與不可控制邊界正式寫入執行模型。

本文區分四種世界差分：

- 內部計算差分：記憶體、值與暫時結構；
- 權威數位差分：資料庫、身份、狀態機與事件歷史；
- 制度差分：權限、責任、承諾、批准與法律地位；
- 物理差分：設備、位置、能源、物體與人體狀態。

  四者的提交、觀測與可逆性不同。內部計算可能可自由重跑；權威數位狀態需交易與版本；制度差分需要合法主體與程序；物理差分則必須依賴感測、現場安全與不可逆風險治理。

本文提出世界差分判定：

> Γ;W⌧;P
> |-
> Δ W
> :
> Legal
> !
> ε
> triangleright
> O

表示在環境 Γ、世界狀態 W⌧ 與權限 P 下，差分 Δ W 合法，具有效果簽章 ε，並產生觀測與證據義務 O。

本文進一步建立「世界交易」概念。傳統資料庫交易主要維持資料一致性；世界交易則需同時處理：

- 權威狀態；
- 外部事件；
- 物理效果；
- 人類承諾；
- 部分成功；
- 結果未知；
- 補償；
- 證據；
- 使用者接受。

形式上：

> 【WorldTransaction
> =
> StateTransition
> +
> ExternalEffects

本文特別區分「提交」與「完成」。本地資料庫寫入成功，只能證明局部提交；外部 API 接受，只能證明外部系統已受理；感測器確認，只能證明某一物理訊號；最終完成還可能需要人類接受、制度確認或長時程觀測。

因此：

> 【Committed
> not⇒
> Completed
> not⇒
> Accepted】

本文也提出「世界版本」與「差分衝突」。意圖 IR 可能基於 W☒ 生成計畫，但執行時世界已進入 W_(t+k)。Runtime 必須檢查：

> Version(W_(planned))
> =
> Version(W_(current))

或重新評估差分是否仍合法。這使 optimistic concurrency、狀態版本、因果時鐘與人類再確認成為世界編譯的一部分，而不是只屬於資料庫技術。

本文進一步處理模擬、dry-run、沙盒、分階段提交、兩階段批准、補償、重建與回放。模擬可以預測，但不能替代真實世界證據；dry-run 可以揭露預期差分，但不能保證外部世界不變；沙盒可以限制能力，但若仍可呼叫外部網路、產生費用或改變治理狀態，就不是真正效果沙盒。

本文使用檔案修改、資料庫遷移、訂單付款、網站部署、AI Agent 多工具執行、研究發布與物理設備控制等案例，說明世界差分如何被提出、驗證、提交、觀測與核對。本文最後提出可證偽研究綱領，包括預測—實際差分誤差、提交—完成落差、世界版本衝突、外部效果證據率、部分成功治理、物理效果確認、世界交易恢復率、沙盒效果洩漏及 AI Agent 差分預覽對事故率的影響。

本文為第 2 冊最後一篇〈後文本程式語言〉建立執行地基：當語言能表示意圖、算子、語意、效果與世界差分後，下一步就是統合文字、視覺、自然語言、驗證、Runtime 與物理耦合，形成後文本程式語言的完整框架。

關鍵詞： 可編譯世界、世界狀態差分、世界交易、Runtime、效果提交、物理耦合、結果未知、差分核對

SECTION

Abstract

This paper redefines program execution as the governed proposal, validation, authorization, commitment, observation, and reconciliation of a world-state difference.

A compilable world is not a fully predictable world. It is a world in which enough entities, states, rules, permissions, effects, and evidence obligations are structured to support pre-execution validation and post-execution reconciliation.

The paper distinguishes predicted and actual world differences, local commitment and final completion, digital, institutional, and physical effects, world transactions, version conflicts, partial success, residuals, and evidence obligations.

Keywords: compilable world, world-state difference, world transaction, runtime, effect commitment, physical coupling

傳統敘事常說：

CPU 執行指令。
這在機器層正確，但對系統層不完整。

使用者真正關心的不是：

- 哪個 opcode 被執行；
- 哪個暫存器改變；
- 哪個函式回傳。

而是：

- 文件是否真的修改；
- 付款是否真的完成；
- 網站是否真的公開；
- 權限是否真的撤銷；
- 機器是否真的啟動；

- 受影響者是否真的收到結果。

  因此，程式執行必須被提升為世界差分問題。

  本文定義：

> 【W
> =
> <
> E,
> S,
> R,
> A,
> B,
> H,
> X,
> O
> >】

SECTION

2.1 實體 E

世界中具有身份、生命週期與關係的對象。

SECTION

2.2 狀態 S

權威狀態、局部狀態、投影與未知狀態。

SECTION

2.3 規則 R

合法轉移、不變量、例外與生效版本。

SECTION

2.4 主體 A

提出意圖、批准、執行、受影響與承擔責任的主體。

SECTION

2.5 邊界 B

系統控制、觀測、權限、責任與非保證邊界。

SECTION

2.6 歷史 H

事件、承諾、版本、失敗、補償與決策歷史。

SECTION

2.7 外部性 X

外部服務、制度、物理環境與他者意志。

SECTION

2.8 可觀測性 O

可取得的回執、事件、感測、證據與未知範圍。

SECTION

3.1 可編譯

若世界中的某一部分具有：

- 可識別實體；

- 可表示狀態；
- 明示規則；
- 可判定權限；
- 可描述效果；
- 可取得證據；

則候選差分可以被編譯與驗證。

SECTION

3.2 不可完備世界

真實世界仍包含：

- 未知；
- 隨機；
- 延遲；
- 人類自由選擇；
- 感測誤差；
- 外部制度；
- 未被模型表示的關係。

SECTION

3.3 正式命題

> 【Compilable
> ≠
> Fully Predictable】

SECTION

3.4 成熟性的判準

成熟世界編譯器不是假裝未知不存在，而是明示：

known
assumed
predicted
unobservable
externally-controlled
human-dependent
contested
本文提出：

> 【Propose
> →
> Validate
> →
> Authorize
> →
> Commit
> →
> Observe
> →
> Reconcile】

SECTION

4.1 Propose

由意圖 IR 產生候選世界差分：

> widehatΔ W

SECTION

4.2 Validate

檢查：

- 前置狀態；
- 世界版本；
- 不變量；
- 規則；
- 資源；
- 依賴；
- 安全守衛。

SECTION

4.3 Authorize

檢查：

- 主體；
- 委任；
- 作用域；
- 時效；
- 人類批准；
- 不可代理選擇。

SECTION

4.4 Commit

提交權威數位狀態變更，或向外部世界發出作用。

SECTION

4.5 Observe

收集：

- 事件；
- 回執；
- 日誌；
- 追蹤；
- 感測；
- 人類回應。

SECTION

4.6 Reconcile

比較：

> widehatΔW
>
> 與
>
> ΔW

並處理：

- 部分成功；
- 結果未知；
- 額外效果；

- 補償；
- 重建；
- 人類結案。

SECTION

5.1 預測差分

$$
\widehat{\Delta W} = \mathrm{Predict}(W_{\square}, \mathcal{I}_R)
$$

SECTION

5.2 實際差分

$$
\Delta W = \mathrm{Observe}(W_{t+1}) - \mathrm{Observe}(W_{\square})
$$

SECTION

5.3 差分誤差

> ε_W
> =
> Δ W
> -
> widehatΔ W

SECTION

5.4 誤差來源

- 世界模型遺漏；
- 外部服務行為；
- Runtime 漂移；
- 並行行動；
- 權限改變；
- 人類拒絕；
- 感測不足；
- 物理故障。

SECTION

5.5 誤差不是單純 bug

差分誤差可能揭示：

- 模型邊界；

- 外部性；
- 受影響者遺漏；
- 規則衝突；
- 不可觀測世界。

SECTION

6.1 內部計算差分

例如：

- 記憶體值；
- 暫時 AST；
- 中間計算；
- 快取；
- 模擬狀態。

此類差分通常：

- 作用域小；
- 可重算；
- 可丟棄；
- 不直接形成外部責任。

SECTION

6.2 權威數位差分

例如：

- 資料庫狀態；

- 身份；
- 事件歷史；
- 權限表；
- 版本；
- 已發布數位內容。

需要：

- 交易；
- 版本；
- 不變量；
- 稽核；
- 恢復。

SECTION

6.3 制度差分

例如：

- 批准；
- 委任；
- 撤銷；
- 合約承諾；
- 規則生效；
- 責任轉移。

制度差分不能只以資料列修改理解，它需要合法主體、程序與生效條件。

SECTION

6.4 物理差分

例如：

- 馬達運轉；
- 門鎖開啟；
- 物品移動；
- 能源消耗；
- 人體處置；
- 環境改變。

  需要感測、現場安全與不可逆性治理。

  本文提出：

> 【Γ;W⊠;P
> |-
> Δ W
> :
> Legal
> !
> ε
> triangleright
> O】

SECTION

7.1 Γ

包含：

- 型別；
- 語意；
- 規則；
- Runtime；
- 契約；
- 工具版本。

SECTION

7.2 W□

當前權威世界狀態與外部觀測。

SECTION

7.3 P

權限、委任、批准與禁止範圍。

SECTION

7.4 Legal

差分是否符合世界規則與責任邊界。

SECTION

7.5 ε

效果簽章，包括讀寫、事件、外部、物理與治理效果。

SECTION

7.6 O

提交後必須完成的觀測、證據、通知與核對義務。

SECTION

8.1 前置守衛

state-version matches
actor authorized
resource available
external dependency healthy
human approval valid

SECTION

8.2 不變量

提交前後均必須維持：

> Inv(W⊠)
> ∧
> Legal(Δ W)
> ⇒
> Inv(W_(t+1))

SECTION

8.3 安全守衛

對物理與高風險效果還需：

- 感測器有效；
- 緊急停止可用；
- 人員安全區域；
- 最大效果範圍；
- 雙重批准；

- 失效安全模式。

SECTION

8.4 守衛失效

守衛失效應產生正式結果：

rejected
stale-world
permission-revoked
unsafe-environment
approval-expired
而不是繼續以舊計畫執行。

SECTION

9.1 計畫基準

IR 生成於世界版本：

> Version(W_(planned))=v

SECTION

9.2 執行時世界

執行時可能已是：

> Version(W_(current))=v+k

SECTION

9.3 版本檢查

若：

> v≠ v+k

Runtime 需判定：

- 差分仍合法；
- 可自動重基；
- 需要重新編譯；
- 需要人類確認；
- 必須拒絕。

SECTION

9.4 樂觀並行

差分可攜帶：

expected_world_version: 18
版本不符時不直接覆寫。

SECTION

9.5 不只是資料列版本

世界版本還可能包含：

- 規則版本；
- 權限版本；
- 外部服務狀態；
- 感測狀態；
- 人類批准狀態。

本文提出：

> 【WorldTransaction
> =
> StateTransition
> +
> ExternalEffects
> +
> EvidenceObligations
> +
> ResidualGovernance】

SECTION

10.1 狀態轉移

修改權威世界狀態，並維持不變量與版本。

SECTION

10.2 外部效果

可能涉及：

- 第三方服務；
- 通知；
- 金流；
- 公開網路；
- 制度承諾；
- 物理設備。

SECTION

10.3 證據義務

每一重要效果應定義：

- 成功證據；
- 失敗證據；
- 結果未知證據；
- 人類接受；
- 保存期限。

SECTION

10.4 殘差治理

世界交易可能無法完全達到預測差分，需保存：

- 未完成；
- 不可逆；
- 補償中；
- 爭議；
- 外部待確認；
- 模型外影響。

SECTION

10.5 資料庫交易只是子集

資料庫原子性、隔離性與持久性非常重要，但不能包住所有外部、制度與物理效果。

SECTION

11.1 Committed

權威系統已接受並保存差分，或外部命令已正式送出。

SECTION

11.2 Completed

所有必要狀態、外部效果與證據義務已達成。

SECTION

11.3 Accepted

合法主體或制度已接受結果，必要申訴與補救位置已滿足。

SECTION

11.4 三者分離

> 【Committed
> not⇒
> Completed
> not⇒
> Accepted】

SECTION

11.5 例子

付款請求已提交，不代表銀行已授權。

銀行授權完成，也不代表客戶爭議已結案。

部署已提交，不代表使用者旅程正常。

訊息已送達，不代表收件人已閱讀與接受。

SECTION

12.1 Prepare

建立候選差分、鎖定資源或保存檢查點。

SECTION

12.2 Approve

由合法主體批准高風險節點。

SECTION

12.3 Commit

正式修改權威狀態或啟動外部效果。

SECTION

12.4 Verify

收集外部與 Runtime 證據。

SECTION

12.5 Finalize

更新整體工作流為完成或接受。

SECTION

12.6 優點

分階段提交可避免：

- 一次性不可逆跳躍；

- 舊批准套用新世界；
- 低階成功冒充整體完成；
- 無證據就關閉任務。

SECTION

13.1 模擬

在模型世界中執行候選差分：

> Simulate
> (
> W⊠,
> widehatΔ W
>)
> =
> widehatW_(t+1)

SECTION

13.2 Dry-Run

執行真實解析、驗證與規劃，但阻止某些提交效果。

SECTION

13.3 模擬價值

可提前發現：

- 狀態衝突；
- 權限不足；
- 資源不足；

- 影響範圍；
- 不可逆節點；
- 缺少證據。

SECTION

13.4 模擬限制

模擬不能完整預測：

- 真實網路；
- 人類回應；
- 外部制度；
- 未知物理環境；
- 並行世界行動。

SECTION

13.5 Dry-Run 洩漏

若 dry-run 仍會：

- 呼叫真實外部服務；
- 寫入追蹤；
- 產生費用；
- 發出通知；
- 修改配額；

它就不是零效果執行。

SECTION

14.1 程序沙盒

限制：

- CPU；
- 記憶體；
- 檔案；
- 系統呼叫。

SECTION

14.2 世界效果沙盒

還需限制：

- 網路；
- 金流；
- 公開發布；
- 權限；
- 敏感資料；
- 外部設備；
- 自動委任；
- 費用。

SECTION

14.3 虛擬效果

可以把外部效果替換為：

- 模擬服務；
- 假事件；
- 測試帳戶；
- 虛擬設備；
- 暫存命名空間。

SECTION

14.4 沙盒證據

系統必須能證明哪些效果被攔截、模擬或允許。

SECTION

14.5 Agent 沙盒

AI Agent 的沙盒應按算子效果分類，而不只是限制其程式程序。

SECTION

15.1 請求送出

只表示本系統建立了外部作用嘗試。

SECTION

15.2 外部接受

外部系統確認收到與受理。

SECTION

15.3 外部完成

外部系統宣稱已完成。

SECTION

15.4 本地觀測

本系統取得回執、回呼或再次查詢結果。

SECTION

15.5 世界接受

必要主體或制度接受外部效果。

SECTION

15.6 分層狀態

effect-proposed
request-sent
externally-accepted
externally-claimed-complete
locally-verified
human-accepted
不同層級不能由同一個 success=true 取代。

SECTION

16.1 命令不是物理效果

> CommandSent
> not⇒
> PhysicalEffectOccurred

SECTION

16.2 感測確認

物理效果需要：

- 感測器；
- 多重訊號；
- 時間窗口；
- 合理範圍；
- 故障辨識；
- 必要人工確認。

SECTION

16.3 感測器也可能錯

因此證據應保存：

- 感測器身份；
- 校準；
- 信心；
- 時間；
- 交叉驗證；
- 不可觀測區域。

SECTION

16.4 物理安全

若物理環境與計畫不同，Runtime 應拒絕、停止或進入安全降級。

SECTION

16.5 物理殘差

即使控制效果被補償，也可能留下：

- 能源消耗；
- 機械磨損；
- 人員影響；
- 環境變化；
- 時間延誤。

SECTION

17.1 制度狀態

例如：

- 批准；
- 撤銷；
- 合約生效；
- 角色任命；
- 帳號停權；
- 爭議結案。

SECTION

17.2 合法主體

制度差分只有在合法主體、程序與證據成立時生效。

SECTION

17.3 數位紀錄不等於制度有效

資料庫寫入 approved=true 不自動創造合法批准。

SECTION

17.4 生效時間

制度差分可能：

- 立即生效；
- 未來生效；
- 等待通知；
- 等待冷靜期；
- 可申訴；
- 暫時有效。

SECTION

17.5 治理 Runtime

未來 Runtime 必須能執行程序規則，而不只是技術流程。

SECTION

18.1 定義

執行後無法判斷效果是否發生：

> Outcome
> =
> Unknown

SECTION

18.2 常見原因

- 超時；
- 回覆遺失；
- 感測器故障；
- 外部系統不可查；
- 並行事件；
- 身份不一致。

SECTION

18.3 安全處理

結果未知時可：

- 查詢原行動身份；
- 等待外部回執；
- 對帳；
- 人工核對；
- 阻止重複不可逆提交。

SECTION

18.4 不得布林化

Unknown 不是 false。

它表示世界狀態與系統認知之間存在未閉合差距。

候選差分可拆為：

> widehatΔ W
> =
> {
> δ⊠,δ⊠,⋯,δ⊠
> }

執行後：

> Δ W
> =
> Δ W_(complete)
> ∪
> Δ W_(failed)
> ∪
> Δ W_(unknown)

SECTION

19.1 已完成子差分

具有足夠證據。

SECTION

19.2 失敗子差分

明確未發生或被拒絕。

SECTION

19.3 未知子差分

無法確認。

SECTION

19.4 核對計畫

每一子差分需決定：

- 保留；
- 重試；
- 補償；
- 查詢；
- 人工決策；
- 接受殘差。

SECTION

20.1 回滾

將權威數位狀態回到先前版本。

SECTION

20.2 補償

以新差分抵銷已發生效果。

SECTION

20.3 重建

由事件、快照、外部證據與人工核對重建世界狀態。

SECTION

20.4 差異

> Rollback
> $\neq$
> Compensation
> $\neq$
> Reconstruction

SECTION

20.5 世界歷史不能被假裝刪除

任何回滾與補償都應保留事件、原因、責任與殘差。

SECTION

21.1 執行後不是結束

世界提交後，Runtime 還需持續收集：

- 狀態版本；
- 領域事件；
- 外部回執；
- 感測；
- 人類回應；
- 錯誤與延遲。

SECTION

21.2 核對函數

> Reconcile
> (
> $\widehat{\Delta W}$,
> ΔW,

SECTION

21.3 核對結果

matched
partially-matched
diverged
outcome-unknown
compensation-required
human-acceptance-required

SECTION

21.4 額外效果

若實際世界產生未預測效果：

> $$\Delta W_{(extra)} = \Delta W - \widehat{\Delta W}$$

就應觸發：

- 效果提升；
- 安全審查；
- IR 更新；
- 使用者通知；
- 停止後續提交。

SECTION

21.5 遺漏效果

若預測效果未發生，則需判斷：

- 失敗；
- 延遲；
- 外部拒絕；
- 觀測不足；
- 世界模型錯誤。

SECTION

22.1 不只是技術日誌

世界差分日誌應保存：

- 原始意圖；
- IR 版本；
- 差分預測；
- 批准；
- 提交；
- 實際事件；
- 證據；
- 殘差；
- 核對；
- 補償。

SECTION

22.2 差分身份

每一世界提交應具有：

world_change_id
intent_id
plan_id
actor_id
authorization_id

SECTION

22.3 隱私與最小化

世界日誌可能含敏感資料，需：

- 欄位最小化；
- 分層存取；
- 加密；
- 保留期限；
- 可撤回投影；
- 稽核。

SECTION

22.4 歷史用途

可用於：

- 事故分析；
- 世界重建；
- 責任；
- 語意差分；
- AI 學習；
- 規則改進。

意圖：

修改這份設定檔。

SECTION

23.1 預測差分

- 讀取檔案；
- 修改指定節點；
- 保存新版本；
- 不改其他區域。

SECTION

23.2 守衛

- 檔案身份；
- 預期雜湊；
- 工作區權限；
- 語法與 schema；
- 是否由其他程序修改。

SECTION

23.3 提交

使用暫存檔、原子替換或版本控制提交。

SECTION

23.4 觀測

重新讀取並比較語意結構，而不只相信寫入函式回傳。

SECTION

23.5 殘差

格式化工具可能改變其他區域，因此文字差分與語意差分均需核對。

SECTION

24.1 候選差分

- 新增欄位；
- 轉換資料；
- 切換讀寫；
- 移除舊結構。

SECTION

24.2 世界版本

舊應用與新應用可能同時運作。

SECTION

24.3 分階段提交

expand
→ backfill
→ dual-read
→ switch-write
→ verify
→ contract

SECTION

24.4 證據

- 資料筆數；
- 一致性；
- 查詢結果；
- 應用錯誤；
- 回滾點；
- 使用者流程。

SECTION

24.5 失敗

遷移腳本執行成功，不代表所有歷史資料語意正確。

SECTION

25.1 意圖 IR

包含訂單、金額、幣別、客戶、付款身份與成功條件。

SECTION

25.2 預測差分

- 建立付款嘗試；
- 呼叫支付提供者；
- 形成付款事件；
- 更新訂單狀態；
- 產生財務證據。

SECTION

25.3 結果未知

超時後不能建立新的付款身份盲目重試。

SECTION

25.4 核對

應以原行動身份向提供者查詢，再決定：

- 記錄成功；
- 標記拒絕；
- 人工對帳；
- 補償退款。

SECTION

25.5 完成

付款授權、本地記錄、結算與訂單履約是不同世界差分。

SECTION

26.1 預測差分

- 建置新產物；
- 部署新版本；
- 通過驗證；
- 切換流量；
- 公開新內容。

SECTION

26.2 Dry-Run

可檢查配置、依賴、資源與預期路由，但不能保證真實流量正常。

SECTION

26.3 提交層級

artifact-created
deployment-committed
health-verified
traffic-shifted
public-journey-accepted

SECTION

26.4 外部殘差

即使回滾：

- CDN 可能仍有快取；
- 搜尋引擎可能已抓取；
- 使用者可能已看見；
- 外部連結可能已散布。

SECTION

26.5 世界交易

發布不是單一資料庫交易，而是多層世界提交。

SECTION

27.1 計畫

Agent 可能依序：

- 讀文件；
- 修改檔案；
- 執行測試；
- 建立部署；
- 發布；
- 更新索引。

SECTION

27.2 差分預覽

執行前必須列出：

- 所有寫入；
- 外部工具；
- 不可逆操作；
- 權限；
- 費用；
- 失敗與補償。

SECTION

27.3 動態世界

執行途中，檔案、規則、權限與外部狀態可能改變。

Agent 必須在高風險節點重新驗證世界版本。

SECTION

27.4 工具回覆

工具回傳是證據候選，不自動等於世界差分成立。

SECTION

27.5 停止條件

如果實際效果偏離預測，Agent 應停止、核對或請求人類，而不是繼續完成舊計畫。

SECTION

28.1 預測差分

- 文件版本凍結；
- metadata 建立；
- 網站公開；
- 索引更新；
- 引用識別建立。

SECTION

28.2 非目標

發布不表示：

- 命題被證明；
- 同行評審完成；
- 社群接受；
- 研究無錯誤。

SECTION

28.3 世界證據

需要：

- 發布版本；
- 公開頁面；
- 內容雜湊；
- 日期；
- 作者；
- 主張狀態。

SECTION

28.4 修改與撤回

後續修訂應形成新版本；撤回公開內容也不能消除既有下載與引用歷史。

意圖：

讓機器開始運作。

SECTION

29.1 差分計畫

- 檢查設備身份；
- 檢查安全感測；
- 取得批准；
- 啟動能源；
- 發送命令；

- 觀測運轉；
- 持續監控。

SECTION

29.2 物理確認

命令成功只表示控制系統接受，仍需轉速、位置、電流或其他感測證據。

SECTION

29.3 偏差

設備可能：

- 未啟動；
- 部分啟動；
- 以錯誤速度運轉；
- 造成額外震動；
- 觸發安全停止。

SECTION

29.4 核對

實際物理差分與預測不同時，必須停止後續計畫並進入安全程序。

- 執行即指令消耗： 忽略程式真正提交世界差分。
- 預測即真實： 將模型差分當成已發生效果。
- 可編譯即完全可控： 否認未知、他者意志與物理不確定。
- 世界狀態無版本： 使用舊狀態直接提交新差分。

- 資料版本即世界版本： 忽略規則、權限、外部與感測變化。
- 驗證後永久有效： 計畫執行時不重新檢查高風險守衛。
- 提交即完成： 本地寫入成功便關閉整體工作流。
- 完成即接受： 忽略人類、制度與申訴位置。
- 資料庫交易等於世界交易： 外部與物理效果被排除於一致性之外。
- 請求送出即外部成功： API 入列被當成外部完成。
- 命令送出即物理效果： 缺少感測確認。
- 數位紀錄即制度有效： 權限或批准缺少合法程序。
- Unknown 布林化： 結果未知被當成失敗或成功。
- 部分成功壓平： 多個子差分被單一狀態碼遮蔽。
- Dry-run 零效果迷思： 模擬流程仍產生外部呼叫、費用或紀錄。
- 沙盒只限制程序： 網路、金流、治理與物理效果仍可洩漏。
- 回滾刪除世界歷史： 忽略外部觀測與不可逆殘差。
- 補償冒充完全復原： 信任、時間、費用與資訊效果未被表示。
- 觀測不足仍宣稱核對： 只根據單一回覆判定世界完成。
- 額外效果不回饋： Runtime 發現未預測作用後仍繼續執行。
- 世界日誌只記技術細節： 無原始意圖、批准、責任與殘差。
- Agent 沿舊計畫前進： 世界偏離後不停止或重新編譯。
- 物理感測單點真相： 感測器故障未被交叉驗證。
- 研究發布等於研究成立： 發布效果與知識真值混淆。

SECTION

31.1 預測—實際差分誤差

測量：

> ε_W
> =
> Δ W
> -
> widehatΔ W

在狀態、事件、外部效果、受影響主體與殘差上的差異。

SECTION

31.2 提交—完成落差

統計已提交工作中，有多少未達外部完成、制度完成或人類接受。

SECTION

31.3 世界版本衝突率

測量候選差分生成後，到實際提交前，世界、規則、權限與外部狀態改變的比例。

SECTION

31.4 重新編譯效益

比較世界版本衝突後直接重試與重新建立 IR／差分計畫，對事故與錯誤提交的影響。

SECTION

31.5 外部效果證據率

統計宣稱完成的 API、訊息、金流與發布效果中，有多少具有足夠外部回執與本地核對。

SECTION

31.6 物理效果確認率

統計物理命令中，有多少具有感測確認、信心、時間與安全證據。

SECTION

31.7 結果未知治理

研究正式 Unknown 狀態是否降低重複付款、重複通知與重複物理操作。

SECTION

31.8 部分成功完整度

檢查多階段工作流是否能分別表示完成、失敗、未知、補償與人類接管。

SECTION

31.9 世界交易恢復率

分別測量回滾、補償、重建與接受殘差後，世界狀態與責任是否真正閉合。

SECTION

31.10 Dry-Run 效果洩漏

統計標示為 dry-run 或模擬的執行中，實際產生網路、費用、資料、權限或外部紀錄的比例。

SECTION

31.11 世界沙盒完整度

比較程序隔離與效果沙盒，在限制資料、外部、物理與治理效果上的差異。

SECTION

31.12 差分核對延遲

測量提交後多久能確認：

- matched；
- partial；
- diverged；
- unknown；
- compensation-required。

SECTION

31.13 額外效果發現率

測量 Runtime 觀測對未宣告寫入、網路、權限與物理效果的檢出能力。

SECTION

31.14 AI 差分預覽效益

比較 Agent 直接執行工具計畫，以及先展示預測世界差分、權限、不可逆節點與恢復後的事故率。

SECTION

31.15 制度差分有效性

統計數位批准紀錄中，有多少真正滿足合法主體、程序、生效時間與申訴要求。

SECTION

31.16 世界日誌可重建性

檢查僅依世界差分日誌，能否重建意圖、批准、提交、證據、殘差與補償歷史。

SECTION

31.17 物理模型殘差

測量設備控制中預測運動、能源與安全狀態和實際感測結果的偏差。

SECTION

31.18 跨世界層教學實驗

比較只教授函式執行，以及同時教授內部、權威數位、制度與物理差分的學習者，在真實系統設計上的表現。

- 程式執行不等於機器指令被消耗。
- 程式執行是候選世界差分的提出、驗證、授權、提交、觀測與核對。
- 可編譯世界不等於完全可預測、可觀測或可控制的世界。
- 世界編譯器必須正式表示未知、外部性與控制邊界。
- 預測世界差分不等於實際世界差分。
- 差分誤差是世界模型、Runtime 與外部性的共同訊號。
- 世界差分至少應區分內部計算、權威數位、制度與物理四層。
- 世界差分合法性需要環境、世界狀態、權限、效果與觀測義務。
- 高風險差分在提交前必須重新檢查世界版本與守衛。
- 資料版本只是世界版本的一部分。
- 世界交易由狀態轉移、外部效果、證據義務與殘差治理共同構成。
- 資料庫交易不能完整取代世界交易。

13.

> Committed
> not⇒
> Completed
> not⇒
> Accepted

- 模擬與 dry-run 能揭露候選差分，但不能替代真實世界證據。
- 效果沙盒必須限制網路、費用、敏感資料、治理與物理能力，而不只限制程序。
- 外部請求送出、外部接受、外部宣稱完成、本地驗證與人類接受是不同狀態。
- 物理命令只有在感測與安全條件確認後，才能提升為物理效果事件。
- 數位紀錄只有在合法主體與程序成立時，才能構成制度差分。
- 結果未知是正式世界狀態，不是 false。
- 部分成功必須分解為已完成、失敗與未知子差分。
- 回滾、補償與重建是不同的世界恢復形式。
- 提交後必須比較預測與實際差分，並處理額外效果與遺漏效果。
- Agent 在世界偏離原計畫時，應停止、重新編譯或請求人類，而非機械完成舊計畫。

24.

> 【程式真正執行的，】

> 【不是一串孤立指令，】

> 【而是一項必須被世界接受、】

> 【被證據確認並對殘差負責的狀態差分。】

SECTION

33.1 承接 PU-2-01

PU-2-01 將程式從文字本體中釋放出來。

本篇指出，無論程式以何種投影存在，執行都必須落到世界差分與實際證據。

SECTION

33.2 承接 PU-2-02

PU-2-02 將符號定義為受治理算子。

本篇將複合算子的效果簽章轉化為候選世界差分、守衛、提交與核對義務。

SECTION

33.3 承接 PU-2-03

PU-2-03 建立語法、語意與效果三層。

本篇集中展開效果層：語意如何在 Runtime 中成為數位、制度與物理差分。

SECTION

33.4 承接 PU-2-04

PU-2-04 建立意圖 IR、效果計畫與多重投影。

本篇讓 IR 進入執行生命週期，並區分預測、提交、完成、接受與殘差。

SECTION

33.5 銜接 PU-2-06

下一篇將完成第 2 冊與整個十八篇地基系列，統合：

- 非文本語言本體；
- 符號算子；
- 語法—語意—效果；
- 意圖 IR；
- 世界差分；

- 多角色投影；
- AI 生成；
- 驗證；
- Runtime；
- 數位、制度與物理耦合。

在低階計算觀中，Runtime 讀取指令、配置資源、更新記憶體並輸出結果。

這個描述仍然成立。

但當程式進入資料庫、網路、制度、Agent 與物理設備後，Runtime 的責任已經超越指令調度。

它必須知道：

- 目前世界版本；
- 哪些狀態是權威；
- 哪些規則仍有效；
- 誰有權提交；
- 哪些效果不可逆；
- 哪些主體會受到影響；
- 哪些證據代表外部完成；
- 哪些結果仍未知；
- 哪些殘差需要補償或接受。

本文因此將 Runtime 重新定義為：

> 【Runtime
> =
> World-Difference Committer
> +
> Effect Observer
> +

而世界編譯器則是：

> 【World Compiler
> =
> Intent-to-Difference Planner
> +
> Legality Checker
> +
> Obligation Generator】

兩者共同形成：

> 【Intent
> →
> Candidate Difference
> →
> Legal Commitment
> →
> Observed Difference
> →
> Reconciled World】

這個結構也重新定位「成功」。

成功不再是函式沒有拋出例外，也不只是資料寫入完成。

成功至少需要回答：

預期世界差分是什麼？
實際發生了什麼？
哪些部分有證據？
哪些部分仍未知？
哪些額外效果出現？
哪些主體尚未接受？
哪些殘差仍需處理？

因此，世界編譯不是將現實幻想成完全可計算。

相反地，它要求系統對自己的不可完備保持誠實。

凡是不能控制的，必須標示外部。

凡是不能觀測的，必須標示未知。

凡是不能逆轉的，必須在提交前揭露。

凡是必須由主體選擇的，必須保留決策節點。

凡是已經改變世界的，都必須留下證據與責任。

本文將可編譯世界收束為：

> 【Compilable World
> =
> Structured State
> +
> Legal Difference
> +
> Authorized Commitment
> +
> Observable Effects
> +
> Residual Governance】

最終命題是：

> 【程式不是因為能執行而完成，】

> 【而是因為它所提出的世界改變】

> 【能被合法提交、真實觀測、】

> 【誠實核對並對未閉合部分持續負責。】

```
world_diff_plan:
  world_change_id: "deploy-release-889"
  intent_ir: "publish-site-20260727"
  based_on_world_version: 42
```

```
  predicted_diff:
    authoritative_state:
      - "deployment.status: approved -> deploying"
      - "active_release: v2.7 -> v2.8"
    external:
      - "upload artifact"
      - "shift public traffic"
    institutional:
      - "production approval consumed"
    residual_risk:
      - "external cache may preserve old or new content"
  guards:
    - "release hash matches approval"
    - "production permission valid"
    - "current world version == 42"
  obligations:
    - "verify public user journey"
    - "preserve rollback evidence"
world_commit_result:
  world_change_id: "deploy-release-889"
  committed:
    - "artifact uploaded"
    - "deployment state updated"
  completed:
    - "service health check passed"
  unknown:
    - "global CDN propagation"
  failed:
    - "mobile authentication journey"
  extra_effects:
    - "analytics endpoint received test traffic"
  reconciliation:
    status: "partially-matched"
    next_action:
      - "hold traffic at 10%"
      - "investigate authentication"
      - "review undeclared analytics effect"
physical_effect:
  action_id: "start-motor-117"
  command:
    sent_at: "2026-07-27T09:00:00+08:00"
    accepted_by_controller: true
  predicted:
    rpm: 1200
    duration: "10m"
```

```
 observed:
   rpm:
     value: 1187
     sensor: "rpm-sensor-4"
     confidence: "high"
   vibration:
     value: "above-normal"
     sensor: "vibration-2"
 status: "diverged"
 safety_action: "controlled-stop"
 human_review_required: true
reconciliation:
 predicted_diff_id: "refund-order-932"
 actual_evidence:
   - "provider-refund-accepted"
   - "local-order-state-updated"
 matched:
   - "payment -> refund-pending"
 missing:
   - "final provider settlement"
 residual:
   - "currency conversion fee not recoverable"
 status: "outcome-pending"
 owner:
   workflow: "order-module"
   financial: "payment-module"
   user_resolution: "support-process"
```

- PU-2-01　程式語言不等於文字：結構、語意與執行的基本分離
- PU-2-02　符號作為算子：從靜態字元到可組合計算閉包
- PU-2-03　語法—語意—效果：程式語言的三層存在結構
- PU-2-04　意圖中介表示：從自然意圖到多重可執行投影
- PU-2-05　可編譯世界：程式執行作為世界狀態差分
- PU-2-06　後文本程式語言：意圖、結構、驗證與物理耦合的統一框架

SECTION

Neo.K／EveMissLab 相關理論

- Neo.K with Aletheia，《程式語言不等於文字：結構、語意與執行的基本分離》，2026。
- Neo.K with Aletheia，《符號作為算子：從靜態字元到可組合計算閉包》，2026。
- Neo.K with Aletheia，《語法—語意—效果：程式語言的三層存在結構》，2026。
- Neo.K with Aletheia，《意圖中介表示：從自然意圖到多重可執行投影》，2026。
- Neo.K with Aletheia，《三宇宙耦合動力學：從想要、表示到世界改變》，2026。
- Neo.K with Aletheia，《表示落差：意圖、計算模型與物理現實之間的不可完備映射》，2026。
- Neo.K with Aletheia，《失敗也是程式：驗證、可觀測、恢復與長期維護》，2026。

SECTION

一般理論背景

- Hoare, C. A. R., “An Axiomatic Basis for Computer Programming,” 1969.
- Lamport, L., Specifying Systems, 2002.
- Gray, J. and Reuter, A., Transaction Processing, 1992.
- Lynch, N., Distributed Algorithms, 1996.
- Kleppmann, M., Designing Data-Intensive Applications, 2017.
- Helland, P., “Life Beyond Distributed Transactions,” 2007.
- Harel, D., “Statecharts: A Visual Formalism for Complex Systems,” 1987.
- Lee, E. A., Plato and the Nerd: The Creative Partnership of Humans and Technology, 2017.
- Henzinger, T. A., “The Theory of Hybrid Automata,” 1996.
- Alur, R., Principles of Cyber-Physical Systems, 2015.
- Woods, D. D., “Four Concepts for Resilience and the Implications for the Future of Resilience Engineering,” 2015.

- Beyer, B. et al., Site Reliability Engineering, 2016.
- Research on digital twins, runtime verification, event sourcing, sagas, and cyber-physical systems.

SECTION

v0.1 — 2026-07-27

- 完成十八篇地基論文第十七篇。
- 將程式執行定義為世界狀態差分的提出、驗證、授權、提交、觀測與核對。
- 建立可編譯世界八元結構。
- 提出可編譯不等於完全可預測。
- 建立世界編譯與世界提交函數。
- 區分預測差分與實際差分。
- 建立內部計算、權威數位、制度與物理四類世界差分。
- 建立世界差分合法性判定、守衛與世界版本檢查。
- 建立世界交易模型。
- 區分提交、完成與接受。
- 建立分階段提交、模擬、dry-run 與效果沙盒。
- 建立外部、物理與制度效果的分層提交。
- 將結果未知與部分成功正式納入世界模型。
- 區分回滾、補償與重建。
- 建立差分觀測、核對與世界差分日誌。
- 加入檔案、資料遷移、付款、部署、AI Agent、研究發布與物理設備案例。
- 提出二十四類失敗模式與十八項可證偽研究方向。

- 完成與 PU-2-06《後文本程式語言》的銜接。