



研究附件

编辑日期：2026 年 10 月 18 日

再生产边界：衡量技术恢复能否独立于原供应商的可检验模型

The Reproduction Boundary: A Testable Model of Supplier-Independent Technical Recoverability

研究类型：概念性研究论文（Conceptual Research Paper），包含两项实际执行、范围有限的计算实验：对一个小型软件制品进行两次确定性重建（Deterministic Artifact Rebuild），以及在供应商缺失条件下检查虚构依赖节点的可达性（Supplier-Loss Dependency Closure）。研究不含生产系统样本、国家级估算或概率预测。

完成状态：已完成下列特定证据的审阅、概念模型设计、演示程序运行与输出核验，以及未来实地研究方案。尚未进行实地比较、沙特服务实验或实证财务可行性研究。

证据截点：2026 年 10 月 18 日。条件性应用区间：2026 – 2035。

中文摘要

问题：在讨论技术独立性时，源代码、数据和物理基础设施的所有权受到高度重视；但这些指标不足以说明：当原供应商的工具、许可、签名或知识不可获得时，一家机构能否重新生产可运行的服务，或对其进行修复。本文提出再生产边界（Reproduction Boundary）：从源资产走向安全且可接受功能的过程中，服务所有者无需依赖不可获得的资产，仍能完成的最后一个有记录的转换。它是特定功能、路径与情境的属性，并非国家的绝对等级。

证据：2024 年旅行者 1 号（Voyager 1）的软件修复、制造商破产后菲斯克 Ocean（Fisker Ocean）汽车的维护路径，以及 2023—2026 年 VanMoof 的知识转移，分别显示技术能力不能简单等同于地理位置、资产所有权或法律权利。可复现构建（Reproducible Builds）、SLSA 1.2、SPDX 与软件更新框架（The

Update Framework; TUF) 提供了保存信息、证明来源或保护发布流程的部分机制, 但任何一种机制都无法单独保证独立恢复运行。

已执行方法: 我们创建了一个小型 Python 演示程序和两个行为测试, 在两个独立临时路径中构建软件包, 并固定输入、文件顺序和时间戳。两份软件包逐字节一致, SHA-256 哈希相同。研究还检查了一棵由 26 个节点组成的虚构依赖树, 节点包括本地资产、供应商要求以及可推导组件。我们执行了 8 次恢复路径检查: 四条路径分别在供应商可用与不可用时各检查一次。在供应商不可用的条件下, 已存档镜像、托管源代码、源代码加本地构建工具三条路径分别缺少 3、5、3 个原始输入; 假定的可迁移核心路径没有缺失输入, 但仅保留八项功能中的六项, 并不声称与原系统完全一致。

结论: 这一检查以有限的计算结果表明, 保存大量文件或获得源代码, 并不必然意味着供应商离场后仍存在一条恢复特定功能的完整路径。它不能证明所提指标可以预测沙特的实际结果, 也不估算市场规模或恢复时间。因此, 我们建议开展一项跨产品独立实验, 比较四类能力: 材料保存、重新构建、行为修复, 以及获得授权的功能迁移。本文同时给出检验方案、证伪条件、安全边界和成本框架。

关键词: 技术独立 (Technical Independence); 再生产边界 (Reproduction Boundary); 功能可恢复性 (Functional Recoverability); 可复现构建 (Reproducible Builds); 软件物料清单 (SBOM); 构建链资产 (Build Chain Assets); 供应商依赖 (Supplier Dependence); 依赖闭包模拟 (Dependency Closure); 软件可继承性 (Software Inheritability)。

English abstract

Question. Does owning a service's data, source code, or compiled image establish the practical ability to rebuild, modify, validate, or replace its essential function when the original supplier is unavailable? We introduce a proposed *Reproduction Boundary*: the last independently executable and verifiable transition within a service's source-to-operation chain, conditional on specified functionality, lawful access, build tools, cryptographic trust, hardware interfaces, skills, and testing.

Completed work. An original small Python demonstration service was packaged twice at separate temporary locations using identical explicitly fixed archive metadata. Its two SHA-256 digests and raw bytes matched, and two basic functional checks passed. An entirely synthetic 26-node dependency graph and four alternative restoration targets were evaluated both with and without vendor primitives, producing eight deterministic route checks. Under the stipulated supplier-absent condition, the archived-image, source-escrow and local-toolchain routes had respectively 3, 5 and 3 unavailable primitive prerequisites and could not complete. The assumed portable-minimum route had none but retained only six of eight stipulated features. These results are properties of an invented graph and toy package; they are **not supplier-outage probabilities, measured Saudi systems, real security certifications, independent reproductions of production software, or restoration-time estimates.**

Contribution and next study. The framework separates byte-level reproducibility, behavioral acceptability, authorization to deploy, and economic attractiveness. It identifies the data and authorized technical experiments needed to learn whether an independently exercised reconstruction capability predicts shorter or

cheaper vendor-exit transitions than inventories, escrow commitments, or system redundancy alone. Proposed falsification and economic decision criteria are explicit; no causal estimate or validated national index is claimed.

Keywords: sovereign recoverability; independent rebuild; software supply chain; hermetic build; functional equivalence; digital maintenance; supplier disappearance; reproducible software; Saudi Arabia.

1. 科学问题与分析单元

核心问题是：在原供应商无法继续提供部分工具或运行权利时，什么条件能使一个主体在合法、技术可行且质量可验证的前提下，重新生产一项明确的运行功能？

分析单元不是整个“国家”，不是“数据中心”，也不是作为商品名称的“应用”。分析对象是：产品某个确定版本中的一项功能或一组明确功能，在已知平台、经授权设定的供应商缺失情境和公开验收标准下的恢复能力。简单报表服务可能可以重建，而依赖特定硬件或授权模型的部分却无法重建；不能用一个抹平差异的共同指标替代具体判断。

“数字主权”包含法律和政治维度，并非都能纳入工程设计。因此，本文把问题分为三组：

1. 材料是否存在？ 数据、源代码、二进制版本、文档、运行镜像、软件包和驱动程序。
2. 路径是否可用且正确？ 工具、使用权、合法拥有的签名密钥、可替代硬件、行为测试和人员经验。
3. 投资这条路径是否划算？ 保持并更新能力的成本、重新开发或购买替代品的成本、供应商缺失造成的预期损失，以及产品剩余寿命。

这些不是凭感觉授予的成熟度等级，而是分母不同的独立检验。

1.1 哪些证据会改变答案？

如果一项获准开展的研究发现，保存一份源代码并不能实际提高独立团队在供应商离场后构建有效版本的能力，那么开发合同的结构应当改变：机构的重点应从单独购买源代码，转向以可检验方式保存构建环境与测试，甚至开发可迁移核心。反过来，如果比较研究表明，对某类服务而言，维持独立仓库增加了故障与成本，却没有真正提高恢复速度，就没有理由普遍推广。

1.2 拟议假设

- H1（路径完整性）：对于依赖授权组件或难以替代组件的产品，在控制产品复杂度、规模及团队因素后，实际通过重新构建测试，比单纯保存源代码更能预测迁移与修复能力。
- H2（功能边界）：如果在供应商中断前就定义了独立于次要特性的核心功能，得到可接受替代品的机会将提高，但可用特性数量可能减少。

- H3（资产的时间有效性）：如果保存源代码的同时，没有维护软件包、编译器基础设施和硬件接口随时间更新的机制，源代码保存的效用会下降。
- H4（条件性回报）：并非每项服务都值得承担独立再生产的成本；价值取决于使用期限、定制深度、替代市场是否可用以及错误成本。

强反假设：购买现代替代产品并重新设计流程，可能比重建旧系统更有效；由拥有大量客户的供应商管理的商业依赖，也可能比维护不足的本地副本风险更低。实验应允许这两种结果出现，而不能把它们隔离为政治异议。

2. 将现实证据与推论分开审视

2.1 旅行者 1 号：物理距离极大，认知上的修复路径仍然存在

美国国家航空航天局（National Aeronautics and Space Administration; NASA）于 2024 年 4 月 22 日宣布，工程数据已经恢复为可理解状态。工程师此前诊断出飞行数据子系统（Flight Data Subsystem; FDS）部分内存损坏，并把部分代码及其指令重新分配到其他位置。探测器距地球超过 240 亿公里，信号单程约需 22.5 小时；首个可观察的实际成功发生在 4 月 20 日。NASA 随后在 5 月宣布两台仪器恢复科学数据，不能把两个阶段合并成“4 月即完全恢复”。[S01S02](#)

有证据支持的观察：在无法实际修理芯片的条件下，远程实施的部分软件修复取得成功。可能机制：团队知道如何解释内存、代码和内部连接，并且有权发送指令、接收验证证据。该事件不能证明：商业退出合同成功、另一艘探测器的构建不再依赖供应商，或修复成本符合普通政府服务可复制的水平。

2.2 菲斯克 Ocean：所有者成为维护产业的一部分

汽车制造商 Fisker 于 2024 年进入破产程序。菲斯克车主协会（Fisker Owners Association; FOA）在 2024 年 7 月 26 日提交破产案件的声明中称，维持服务需要获得源代码、维修工具、云服务、零配件与车辆通信。协会是争取协议的一方，并非中立的结果评估者。应用商店把 Undercurrent Inc. 列为 OceanLink Pro 的开发者和提供方，并在说明中称其为 FOA 会员的官方应用；页面所列功能会因车辆硬件和软件版本不同而变化。声明和开发者提供的产品页面都不能证明所有汽车已经停运，或其全部功能已独立恢复。

[S08S09S10](#)

有证据支持的观察：原厂之外出现了维护与软件项目，并有一个自称代表车主的开发者提供工具。可能解释：分散在用户和工程师之间的有限知识被组织成一种维修能力。推论的迁移边界：协会文书来自利益相关方；应用说明不是针对每辆车的独立测试；部分功能需要专用设备；恢复部分功能不表示全部故障或更新权问题已经解决。替代方案的安全性应由获准的专业人员进行实证测试；本文不提供进入车辆系统或绕过安全措施的方法。

2.3 VanMoof: 三种不同冲击与复合技术寿命

VanMoof 破产受托人公布的原始报告证实，阿姆斯特丹地区法院（District Court of Amsterdam）于 2023 年 7 月 17 日宣布其荷兰相关实体破产。同期，竞争对手 Cowboy 发布 Bikey 应用；上线说明重点介绍把 S3/X3 车型的自行车密钥保存在本地，以应对原厂服务不可用的情况，版本记录随后又写明加入了 A5/S5 支持。该页面不证明所有车型都获得完整、长期支持。[S03S04](#)

2024 年 2 月 27 日，新所有者宣布通过 VanMoof Archives 扩大多个世代自行车维修说明和技术图纸的开放范围，并说明部分资料此前仅限授权维修点使用。这些文档涉及机械与程序性维修，不意味着全部电子图纸、发布密钥或软件所有权已经开放。[S05S06](#)

2026 年，制造商又宣布旧车型所依赖的 2G 网络连接将发生变化，并把 4 月 1 日设为特定用户订阅防盗追踪延续服务的期限。同时明确表示，上锁、骑行及其他独立功能不会停止。这与 2023 年破产事件不同：即使品牌和商业活动恢复，老化的也是设备外部的通信基础设施。[S07](#)

这一序列构成对三种不同原因的概念性检验：商业实体中断、维修知识封闭、外部协议老化。不能用“故障”一词把三者合并，也不能把一项事件的结果归到另一项事件上。维修资料的扩大开放，也可以视为对“所有供应商在重组后都会阻止产品延续”这一看法的部分反例。

2.4 沙特证据：资产可以共享，但不能据此假定能够构建

沙特数字政府管理局（Digital Government Authority; DGA）把政府云软件仓库（Government Cloud Software Repository; GPaaS）描述为一个保存与交换政府自由和开源代码、促进共同开发的平台。房地产总局（Real Estate General Authority; REGA）网站也展示了开放软件要求和政府代码托管平台的作用。[S17S18](#)

这一观察的重要性在于，它指明了开展重新构建测试的现实起点，却不声称平台本身能够衡量独立恢复运行的能力，也不声称沙特所有软件都在其中。对迁移有效性作出判断前，必须针对明确样本开展公开或获准测试。

3. 科学基础：区分四种证据

3.1 长期保存（Digital Preservation）

软件遗产（Software Heritage）档案致力于收集公开开源资产及其开发历史，并通过可追踪引用提供访问。这一目标不同于提供完整生产环境镜像、保证旧版本使用的编译器长期存在，或赋予使用全部商业依赖项的许可。[S15](#)

归档可以挽救不可替代的原始资产，但单独的代码库镜像可能不含外部软件包、运行配置记录和数据库状态，也不能包含本来就不应托管的秘密值。部分内容还可能在法律上不可分发。因此应把资产分为：可保存并公开的材料；可在限制条件下保存的材料；以及不得在合法管控范围外保存或分发的资产。

3.2 构建产物一致性 (Reproducible Builds)

Reproducible Builds 项目把可复现构建定义为：在源代码、环境和指令相同的条件下，能够重新生成逐字节一致的产物。影响环境的细节包括依赖项版本、语言区域设置、时间戳和构建路径。[S11](#)

一致性可以证明：在给定假设和设置下，指定输入可以再次生成相同字节。一致性不能证明：软件无缺陷、编译器无恶意、数据能够导入、发布密钥使用合法，或主管机构会批准使用。它们属于不同种类的证据。

可重新构建性也不总以逐比特一致衡量。平台迁移或架构变化必然改变字节时，更合适的标准是可接受的行为等价 (Behavioral Equivalence)；它需要明确定义输入、输出、误差边界和测试。行为等价不应被描述为二进制一致。

3.3 来源与软件物料清单 (Provenance and SBOM)

软件包数据交换规范 (Software Package Data Exchange; SPDX) 定义了描述软件资产、关系和许可证的方法。软件制品供应链等级 (Supply-chain Levels for Software Artifacts; SLSA 1.2) 则以不同保证级别，提出追踪源代码来源与构建来源的要求。两者都不意味着第二方会自动取得签名设备、激活设施或处理器的权利和能力。[S12S13](#)

完整的组件记录是准备工作的重要证据，却可能在“影子依赖 (Shadow Dependency)”面前止步。影子依赖不会写在软件包清单里，例如远程服务许可、嵌入式硬件密钥、供应商掌握的测试集，或未记录的运行知识。这些只是可能机制的例子，并非对任何特定机构弱点的清单。

3.4 更新安全与信任基础设施 (Secure Software Update)

软件更新框架 (The Update Framework; TUF) 通过签名元数据、不同密钥权限和到期时间，降低更新发布密钥以及旧版或伪造版本带来的风险。最终，在特定硬件上运行新程序，仍受该硬件信任规则约束。[S14](#)

这里必须区分拥有有效签名与有权改变获准签名主体。良好的更新保护可能阻止后续安装未经授权的软件，即使设备所有者持有源代码。绕过保护不应被视为“恢复方案”；需要权利所有者授权、预先设计的合法更新机制，或可用的替代硬件。

3.5 对编译器的信任 (Trusting Trust)

肯·汤普森 (Ken Thompson) 在《Reflections on Trusting Trust》中说明：当生产工具本身值得怀疑时，只检查源代码可能不足以理解二进制制品携带的行为。戴维·惠勒 (David A. Wheeler) 后来提出多样化双重编译 (Diverse Double-Compiling; DDC)，在特定假设下检验某些编译器是否与目标源代码一致。

[S16S19](#)

这并不是要求每家机构重写所有编译器，而是说明“能否构建程序”可以拆成若干保证：工具是否保存；其源代码能否获得；验证方是否相互独立；编译器是否适配目标平台；是否具备与风险相称的验证证据。

4. 概念模型与符号

定义一项特定服务 s 、一项可接受功能 f ，以及供应商缺失情境 z 。其生产路径包含原始资产和派生阶段，构成制品依赖图（Artifact Dependency Graph; ADG）；部分资产可能是必要条件，另一些可能是可选项。没有对实际产品进行检查，就不能把某条依赖关系视为已证实事实。

基本变量：

- $A(v,z)$ ：在情境 z 下，一项原始资产、权利或能力 v 是否能合法且独立地获得；在简化确定性模型中取 0 或 1。
- $Anc(r)$ ：再生产路径 r 所依赖的全部原始要求；通过依赖关系追溯至根节点得到。
- $V(r)$ ：独立判断产物是否达到所要求的安全与行为水平，而不只是文件是否成功编译。
- $L(r)$ ：独立判断在现行协议下运行、发布该路径及保存其资产是否合法。
- $F(r)$ ：路径经实际验证可以提供的功能集合，而不是方案承诺的功能集合。

针对每条路径，提出以下初始构建条件：

$$P_r = \prod_{v \in Anc(r)} A(v, z)$$

$P_r = 1$ 表示在该情境下所有原始要求均可获得，并不意味着程序必然有效、安全或合法。完整的实际再生产还要求 $V(r)$ 与 $L(r)$ 通过验证，且功能 f 属于 $F(r)$ 。若独立替代路径成立，即使其他路径失败，仍可能存在一条有效路径。

在虚构示例中，根层缺失依赖的数量定义为：

$$B_r = \sum_{v \in Anc(r)} (1 - A(v, z))$$

这是阻碍数量，没有经济权重。一个不可替代的密钥可能比十个几分钟内即可从另一仓库恢复的文件更难处理。在没有定义资产及其边界时，不能仅凭 B_r 比较不同服务。

如果有真实时间数据，可以把功能恢复时间定义为：供应商宣布缺失之后，在替代环境中首次通过安全与使用测试所需的时间；实验结束前未成功的案例应视为右删失（Right-Censored），而不是臆造一个无限恢复时间。本文也不建议按保存文件数量计算国家百分比，因为这会混合不同性质的量，并掩盖单个“阻断性障碍”。

5. 已执行的第一项计算实验：从已知输入重复构建

5.1 设计

我们创建了 `analysis/demo_service.py`，其中的简短 Python 函数检查某个虚构标识符是否存在于本地集合中，输出包含请求标识符、布尔标记与模式版本。该函数不判断任何真实权利，也不连接外部数据。

`analysis/reproduce.py` 使用标准 `zipfile` 库打包三个固定元素：服务代码、一个小型 JSON 模式，以及记录参考构建时间的文件。我们对文件名排序，统一时间戳和元数据，然后在两个独立临时路径中构建软件包。两份文件的原始字节和 SHA-256 哈希均被直接比较；没有从互联网下载依赖项，也没有运行商业签名链。

5.2 实际输出

- 同一进程内、空间位置独立的构建次数：2。
- 字节级一致状态：True。
- 两次构建的哈希：36c763c61b7114cffe473a230d5d583ca9accef14ff60bdebfa3d78747fdc52。
- 小型服务正向与负向测试：均通过。
- 临时结果档案不是生产软件包，也未作为可安装更新发布；实验在一个名为 `data/toy_rebuild_verification.json` 的验证文件中记录了软件包大小、两次构建哈希和两个测试结果。

允许得出的科学结论很窄：在这个示例中，当输入和打包信息固定后，两次构建得到相同结果。我们不称其为经第三方验证的独立再生产；两次尝试使用相同代码和同一环境中的标准语言解释器。该实验也不能证明使用不同编译器、许可证、库和硬件的软件具有可复现性。

5.3 尚未在现实系统执行的失败对照

实验边界说明后续仍需开展：在真正独立的两个环境中构建；接收资产前验证其完整性；在测试环境中比较产物行为；测试软件包仓库中断；以及测试获准更新。以上均为拟议方案，并非已经得到的结果。

6. 第二项实验：供应商离场时要求树是否完整

6.1 虚构输入

我们创建 `data/reproduction_graph.json` 描述 26 个节点。其中包括服务源代码、数据导出及其模式、本地构建工具、软件包镜像等根资产，也包括专有编译器、开发工具包、激活、签名和固件等供应商要求，以及可从前序节点推导的结果节点。

这是一张人为构造的教学图，不代表任何特定企业、沙特设备类别，也不披露真实控制点或故障点。我们把原始组件的可获得性与完整运行功能是否成功分开；本实验预先假定各路径的测试和其他权利条件已经满足，程序不会评价或绕过这些条件。

数据定义四条路径：

1. `image_archive`：以保存的运行镜像部署，同时需要数据、配置，以及供应商提供的激活、签名与硬件条件。
2. `source_escrow`：从可用源代码构建，但使用原有编译器和开发工具包，以及原运行要求。
3. `source_with_local_tools`：使用本地工具和软件包构建源代码；部分激活、签名与原硬件要求仍然存在。
4. `portable_minimum`：重新设计部分功能，并以假定的替代组件、有效许可和合法本地签名进行组合；功能范围小于原系统。

我们在供应商可用与供应商不可用两种条件下分别检查每条路径。两种情境都没有概率权重，只是确定性的条件状态。

6.2 实际输出

实验路径	供应商可用时缺失根节点数	供应商缺失时缺失根节点数	第二种条件下能否完成	假定功能数，共 8 项
保存的镜像	0	3	否	8 项，有条件
托管源代码	0	5	否	8 项，有条件
源代码与本地工具	0	3	否	8 项，有条件
可迁移核心	0	0	在示例条件内为是	6

程序执行了 8 次评估，并不是在 8 个不同真实产品上做了 8 项实验。实验把各路径缺失要求的名称和输出记录在一个名为 `data/closure_results.csv` 的文件中。一项内部测试确认不可用根节点数量与上述结果一致。

这些数字表明，在一套仍要求供应商资产的方案中，源代码托管本身不足以替代这些资产；转向可迁移核心可以降低依赖，但会牺牲部分功能。这一结构性结果对依赖图的画法高度敏感：节点、使用权或可用替代项发生变化，判断也会变化。实验没有测量修复成本、迁移时间或恢复版本的安全性。

6.3 如果重新设计实验，会发生什么？

若假定供应商在离场前授予无条件运行权，并提供独立有效的更新密钥，镜像路径和源代码路径的部分缺失节点可能消失。若可迁移替代项依赖一份无权修改的源代码，即使文件树看似完整，该路径仍会失败。若功能标准要求全部八项功能，可迁移核心便不能在同一标准下算作成功，尽管可以称为“部分替代”。

因此，验收标准的精确定义不是研究装饰，而是在决定“我们恢复了什么”这句话的含义。

7. 经济维度：为什么不重新构建一切？

本模型不是不计成本地把所有产品都变成本地制造的路线图。决策需要比较重新构建选项（Rebuild Option）与购买替代品选项（Replacement Option），并计算保持某一选项开放的价值。企业或公共机构今天可以支付额外成本，取得某些权利和工具，以降低五年后更换供应商的复杂度；但如果软件寿命很短，且市场上已有成熟竞争替代品，这笔成本可能超过收益。

在不臆造国家级数字的前提下，建议区分六组成本：

1. 保持可构建性的成本：保存源代码与获准的软件材料、更新软件包镜像、记录版本、定期测试。
2. 失去专业化的成本：改用可迁移替代方案时，放弃针对特定任务优化的硬件或软件性能。
3. 知识成本：工程师理解旧系统并在不引入新故障的情况下改变行为所需的时间。
4. 权利与信任成本：许可证、获准的签名方式、更新维护和适当披露。
5. 停机或服务替换成本：已记录功能损失、生产延误与迁移费用。
6. 完整替代成本：购买新系统、转换数据、培训人员，并按可接受程序停用旧系统。

机构应把维持修改能力的成本与彻底淘汰整套技术的价格比较，而不是只与一张旧服务器镜像的价值比较。沉没成本（Sunk Costs）也必须与未来可避免成本分开；过去的投资回报不能成为继续使用无效产品的理由。

7.1 会削弱本文论点的情境

如果一种产品由多家供应商提供，采用标准数据格式；数据能在短时间内迁移并测试；功能不需要深度定制；更新持续且安全，那么维护一套独立重建工厂可能只是浪费。此时，实践中的主权体现为容易替换，而不是逐字节重造旧软件。

8. 拟议的沙特研究方案

8.1 所需设计与样本

现实检验需要机构批准，在沙特四个相关方向中选择非敏感服务或获准的测试版本：政府计算、经济技术服务、人工智能组件，以及部分商业数字基础设施产品。功能应按复杂度、硬件依赖、合同性质和预期寿命分类，而不是只按供应商名称分类。

一个独立于开发团队的团队，应检验自己无需供应商账户便能从获准资产包中生产出什么。研究人员不得接触真实用户数据或运行秘密；必要时应使用隔离实验室以及合成或匿名数据。研究不需要在在线关键系统上做实验。

8.2 四个可记录的实际阶段

第一阶段：保存源代码与原始资产。 团队清点依法可接收的源代码、依赖图、许可证、镜像、软件包和测试数据，并准确记录缺失内容，以及已知存在但禁止访问的内容。

第二阶段：复现构建。 团队尝试创建运行版本，或说明不能创建的原因，同时保留输入、命令、工具版本和哈希记录。若结果不同，应区分预期差异与可复现性失败。

第三阶段：修复一个预先选定的安全问题。 在测试环境中，团队负责修复一个合成缺陷或修改一项范围很窄的功能；尝试前固定测试集。通过测试不能被描述为全面安全证明。

第四阶段：迁移一项核心功能。 团队尝试在获准的替代平台上运行选定核心功能。当有可靠测量设备时，应记录功能数量、验收结果，以及延迟、成本与能耗差异。结果按服务计算，不能合并成一个暗示所有功能同质的国家指数。

8.3 测量什么，不测量什么？

拟议主要输出： 再生产是否成功及失败原因；从实验开始到首次有效构建的时间；到首次可接受修复的时间；通过测试的功能数量；所需法律条件；实际人力投入；以及路径各组成部分的成本。观察期内未能完成实验的案例，应记录为单独状态，而不是臆造一个估计的无限恢复时间。

比较条件： 只有源代码；只有二进制镜像；源代码加已归档构建包和驱动程序；现成的标准替代品。这些特征可能重叠，选择机制必须预先说明，以免最优秀团队总是分配到最容易的产品。

因果风险分析： 成功重建不能证明成功由归档造成。团队可能更熟练、组件更简单、文档更新，或供应商此前更合作。研究需要预先分层、重复实验和清晰的实施比较。如果数据不允许进行因果设计，论文只能描述相关关系。

8.4 停止或拒绝试点的条件

若测试需要中断生产系统、绕过制造商签名密钥、取得未经许可的资产，或收集敏感运行图，就必须排除该步骤。可以改用模拟或范围有限的获准程序。独立再生产不值得一破坏其本应保护的安全保证或权利为代价。

9. 面向 2035 年：国家能力的三条路径

以下不是经过计算的预测，而是与可观察信号相连的条件性情境（Conditional Scenarios）。

路径一：不可再造的资产大量积累。代码库、软件镜像和采购文档不断增加，却没有实际测试脱离供应商构建版本。危险信号包括：合同只要求保存源代码，不要求有效环境或测试工具；以及没有独立团队知道怎样发布新版本的软件。

路径二：无选择地普遍推行本地构建。机构试图重造所有采购产品，包括从充分国际竞争中能够更低价购买的产品，结果增加维护成本和延迟，并把专业知识分散到重要性不足以支撑投入的项目上。关键迹象是独立成本上升，修复率或迁移便利性却没有改善。

路径三：可继承维护经济。本地企业和实验室专门再生产经过选择的产品类别，开发代码迁移工具、字节验证、接口恢复、行为测试实验室和修改权文档。成功应以经证实的独立重建、维护收入和可检验的出口衡量，而不是以宣布了多少计划衡量。

9.1 监测机构应观察的六个信号

1. 实际接受测试的功能比例，而不是宣布连续性计划的项目数量。
2. 按产品类型区分的独立团队重建成功率与失败率。
3. 供应商支持实际中断时长，以及对更新测试版本能力的影响。
4. 拥有最新软件包镜像和可合法复用构建环境的产品比例。
5. 实际迁移与购买全新替代品的成本差额，并计入功能缩减。
6. 拥有付费客户购买迁移、验证与修复服务的沙特产品，而不只是已发表的实验室测试。

10. 有效性与推广边界

本研究仍有六项重要限制：

1. 缺少沙特生产数据：本文没有经过批准的沙特机构采购、构建或迁移样本，因此不能声称该指标证明了其脆弱性或强度。
2. 依赖树经过简化：现实依赖可能含有可选条件、循环、版本兼容、共同故障和复杂许可制度；已执行模型只是一棵简化的原始条件树及独立派生项。

3. 安全性尚未证明：构建要求可获得，不表示程序安全或能产生目标结果。演示测试非常小。
4. 权利差异：法律或合同可能禁止使用已保存的源代码、密钥或固件；本文没有对特定产品开展法律评估。
5. 隐性知识问题：依赖树不衡量工程师经验、培训时间或错误空间的广度。
6. 字节构建不同于迁移：把服务迁到另一类硬件会产生不同字节；战略上可能更成功，却无法只靠 SHA-256 一致性测试显示。

因此，我们不把本模型称为**“沙特主权指数”**，也不称为“可恢复性认证”。它是一组概念与定义、两项小型实验，以及一项拟议的现实研究计划。

11. 论点证伪测试

科学贡献需要明确什么证据会削弱它：

- 如果独立服务提供商能够持续地购买新替代品，并以比重建原系统更快、更低成本的方式安全运行，那么对所研究产品类别而言，机构保留构建能力的普遍价值将降低。
- 如果在计入团队经验、软件简单程度和供应商实力后，保留构建链的测试不再增加预测能力，就应修改“再生产边界是一项重要独立指标”的主张。
- 如果无法在不失去关键权利或特性的情况下定义“核心功能”，可迁移核心模型便不适合该服务，即使技术上可实现。
- 如果技术证据证明某项被描述为独占的依赖可以立即由可信且获准的替代项取代，就不能继续在图中把它保留为“阻断性障碍”。
- 如果大多数软件生命周期短且能够高效替换，那么保存通用工程能力，可能优于保存迅速过时的产品构建方案。

这就是前瞻性建议与封闭主张之间的区别：在投入数十亿之前，先问什么结果会迫使我们改变战略。

12. 复现协议与配套研究材料

原稿所述的原始实验文件结构：

文件	内容	边界说明
analysis/demo_service.py	逻辑完全公开的虚构服务	不含真实数据、客户或权利
analysis/reproduce.py	两次创建软件包并检查依赖树	只依赖 Python 标准库
data/reproduction_graph.json	26 个节点、四条路径、两种条件	获准使用的虚构输入，不代表沙特现实
data/closure_results.csv	八项详细输出及依赖缺失原因	没有概率权重的计算条件
data/toy_rebuild_verification.json	软件包大小、两次构建哈希与两个测试结果	本地检查，并非跨机构独立验证
analysis/README.md	执行步骤、假设与验证方法	不公开密钥或运行数据

本版配套材料：[研究材料](#)收录从所提供证据工作簿导出的 `closure_results.csv`、经审阅的来源登记表 `source_register.csv` 和事实案例簿 `news_casebook.md`；本语言研究论文 PDF 另行提供。

原稿把原始实验流程描述为：在工作路径运行 `python3 analysis/reproduce.py`，随后比较 CSV 与 JSON 两项输出。所述流程不假定 Python 标准库之外的软件依赖。第三方环境中的 ZIP 解释器或 Python 版本不同，可能使软件包文件出现差异；此时需要完整描述环境，之后才能判断概念本身是否失败。

临时目录中生成的归档文件不会作为即用软件发布；只保存验证结果，每次运行程序时重新生成软件包。

13. 研究结论

本文的结论不是“供应商都不可信”；供应商可能最擅长恢复运行，也最有能力维持创新。结论也不是“开源能解决一切”；开源工具同样会老化，专业知识也可能消失，复杂权利约束下的功能仍可能无法实现。

更准确的结论是：数字所有权可以与复活一项功能的工程能力彼此分离。一家机构可能保存了完整材料，却没有一条合法、经过测试的路径把材料连接到功能；一个物理资产很少的小团队，也可能保有一条有效修复路径。

两项小型实验表明，在已知环境中可以把一个软件包逐字节构建两次；在虚构依赖模型中，少数根节点缺失足以切断路径，即使其余材料都被保存。至于把这一原则扩展到沙特服务，能否在成本、时间、独立性和维护上产生实际回报，仍是一个需要独立实验的假设。

如果这项实验得以开展，我们获得的将不只是主权口号，而是可检验的知识：一个系统能重新制造什么、谁能制造、在哪种设备上制造，以及在失败时承担代价的人能够理解的成本、权利与安全条件。

原始参考资料与追踪标识

来源细节、出版与事件日期、推理边界和引用分支，汇集在证据包内的来源登记表（`source_register.csv`）与事实案例簿（`news_casebook.md`）中。相应标识如下：

1. S01 美国国家航空航天局（NASA）：*Voyager 1 Resumes Sending Engineering Updates to Earth*, 2024 年 4 月 22 日。
2. S02 美国国家航空航天局（NASA）：*Voyager 1 Resumes Sending Science Data from Two Instruments*, 2024 年 5 月 22 日。
3. S03 VanMoof 破产受托人：*Fourth Public Bankruptcy Report*, 2024 年 6 月 6 日；记录 2023 年 7 月 17 日事件。
4. S04 Cowboy SA: Bikey 应用原始页面；上线说明重点介绍 S3/X3，版本记录随后写明加入 A5/S5 支持，2023 年 7—8 月。
5. S05 VanMoof: 新所有者恢复经营及客户补偿计划声明，2024 年 7 月。
6. S06 VanMoof: *VanMoof Archives for Legacy Models*, 2024 年 2 月 27 日。
7. S07 VanMoof: *What's Happening with 2G Network Shutdown*, 2026 年说明。
8. S08 菲斯克车主协会（Fisker Owners Association）：2024 年 7 月 26 日向美国法院破产案件提交的车主协会声明，第 2 页。原始声明文本已被索引，但最终核验时文件服务器未响应直接打开请求。
9. S09 Undercurrent Inc.: OceanLink Pro 应用原始页面；页面说明称其为菲斯克车主协会（Fisker Owners Association; FOA）会员的官方应用。
10. S10 Reuters: 2024 年 10 月 11 日 Fisker 破产计划获批报道。
11. S11 Reproducible Builds: *Definitions*, 一致构建的技术定义。
12. S12 SLSA: *SLSA Specification Version 1.2*, 2025 年 11 月 24 日获批，并在证据截点时保持为已批准版本。
13. S13 SPDX: *Specifications*, 软件组件及其许可证清单。
14. S14 The Update Framework: *Overview*, 软件更新安全。
15. S15 Software Heritage: *FAQ*, 长期保存目标。
16. S16 肯·汤普森（Ken Thompson）：*Reflections on Trusting Trust*, 1984 年。
17. S17 沙特数字政府管理局（Digital Government Authority）：*Government Cloud Software Repository (GPaaS)*。
18. S18 沙特房地产总局（Real Estate General Authority）：*Free and Open-Source Government Software*, 未注明发布日期的官方机构页面；页面内容包含截至 2025 年 5 月 12 日的具日期产品。
19. S19 戴维·惠勒（David A. Wheeler）：*Fully Countering Trusting Trust through Diverse Double-Compiling*, 2010 年。

引述说明：正文只使用对已发表观点的归属式概述，没有把虚构对话归于任何工程师、骑行者或研究人员。利益相关方的陈述与应用页面均未被当作对全部功能的独立技术审查。