Technology Obsolescence in a 25-Year Concession: Keeping the Contract Investible

A Lifecycle Framework for Upgrade, Benchmarking, Interoperability and Change

Authored by Chennakeshav Adya, Independent Researcher

September 2026

Abstract

Public-private partnerships combine long contractual terms with technologies whose useful lives, standards, suppliers, economics and security conditions can change rapidly. A concession signed around a particular platform can become operationally brittle, commercially expensive or strategically stranded long before expiry. This paper develops a lifecycle framework for keeping a 25-year concession investible while protecting public value. It separates predictable refresh from disruptive change; translates service needs into outcome-based specifications and modular architecture; uses technology roadmaps, asset registers, interoperability, data portability and supplier-continuity controls; and structures upgrade bands, benchmarking, market testing, variation, cost recovery and benefit sharing. The framework also connects cyber resilience, intellectual property, licensing, energy and compute economics, demand substitution, financing, termination and handback. Five figures and five tables provide an obsolescence taxonomy, architecture-control map, upgrade economics model, change waterfall and technology-investibility certificate. Eight frequently asked questions and forty primary or authoritative references support practical application. Numerical values and scores are illustrative analytical scenarios. Project conclusions require verified technical, operational, cybersecurity, legal, procurement, competition, intellectual-property, tax, accounting, fiscal and financing evidence and advice.

JEL Classification: G31, H54, L86, O31, O32

Keywords: technology obsolescence, public-private partnership, concession, technology refresh, interoperability, benchmarking, lifecycle investment, contract change

1. Define the enduring service outcome

The project team should state the user need, essential service, performance envelope, capacity, accessibility, resilience and public-policy objective. The required output is a technology-neutral service charter. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [1][2].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that the contract can preserve a product while losing the public outcome. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

2. Separate technology from service

The project team should distinguish equipment, software, data, interfaces, communications, operating process and service output. The required output is a layered service-technology map. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [3][4].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that an equipment refresh can be confused with a change in contracted service. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

3. Create the technology baseline

The project team should inventory assets, versions, vendors, licences, support dates, dependencies, security status and replacement cost. The required output is an evidence-dated technology register. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [5][6].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that parties can dispute what was included and serviceable at financial close. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

4. Classify obsolescence risk

The project team should separate wear-out, support withdrawal, standards change, security exposure, economic inefficiency and disruptive substitution. The required output is an obsolescence taxonomy. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [7][8].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that a single technology risk label can conceal very different control and funding needs. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

5. Map lifecycle and contract horizons

The project team should compare useful life, support life, licence term, debt tenor, concession term and handback horizon. The required output is a horizon-mismatch map. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [9][10].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that short-lived components can sit inside long-lived financing without funded replacement. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

6. Design outcome-based specifications

The project team should set measurable outputs and constraints while allowing solution choice and innovation. The required output is an output specification. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [1][11].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that prescriptive requirements can lock both parties into procurement-era technology. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

7. Retain necessary architecture constraints

The project team should define safety, security, interoperability, data, accessibility, resilience and regulatory boundaries. The required output is an architecture guardrail schedule. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [12][13].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that technology neutrality can permit incompatible or weakly governed solutions. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

8. Use modular and open interfaces

The project team should separate replaceable components through documented APIs, protocols, data models and physical interfaces. The required output is a modularity and interface plan. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [14][15].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that monolithic architecture can make a modest refresh require full-system replacement. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

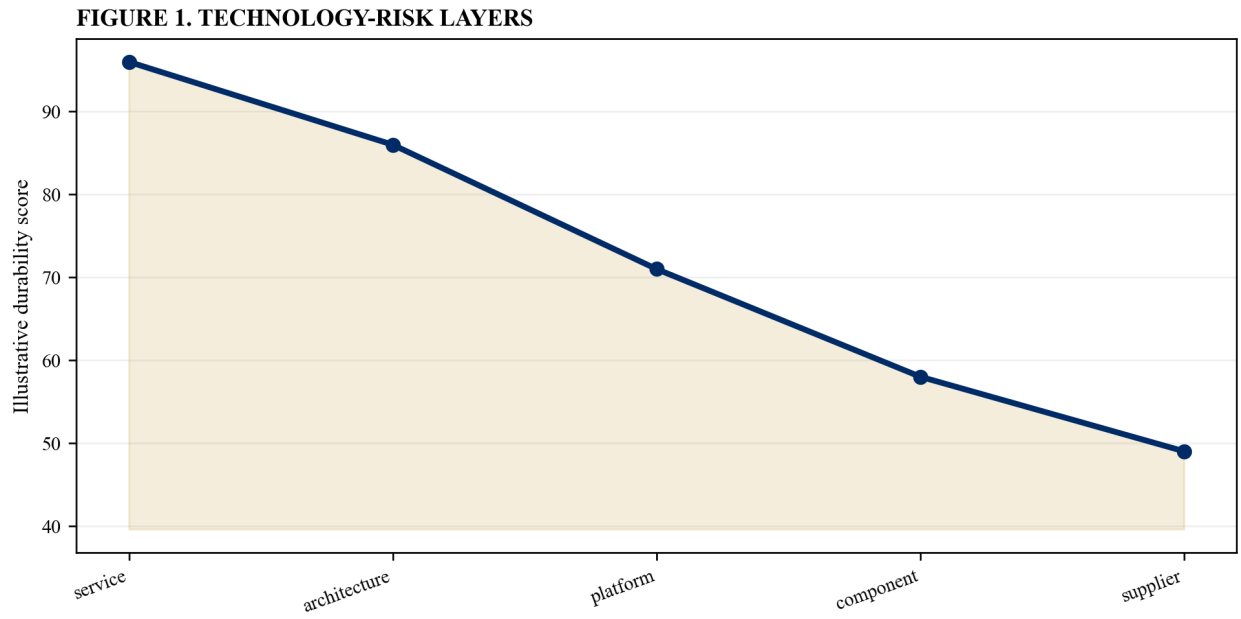
Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

Table 1. Obsolescence taxonomy

Type	Typical trigger	Primary response
wear-out	condition	planned refresh
support withdrawal	vendor date	migration
standards change	external rule	compatibility upgrade
disruption	market shift	shared review

Illustrative analytical structure; verified project and jurisdiction evidence governs.

Figure 1. Technology-risk layers



Illustrative analytical scenario; verified project evidence should replace values.

9. Protect interoperability

The project team should test conformance, backward compatibility, integration, data exchange and independent component substitution. The required output is an interoperability assurance plan. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [16][17].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that nominal standards compliance can fail in actual multi-vendor operation. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

10. Build the technology roadmap

The project team should forecast service growth, standards, vendor support, security requirements and plausible substitution. The required output is a rolling technology roadmap. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [18][19].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that the project can treat change as exceptional until multiple upgrades become urgent together. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

11. Define ordinary refresh obligations

The project team should allocate planned replacement, patching, supported versions, consumables and routine efficiency improvements. The required output is a baseline refresh schedule. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [5][20].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that ordinary lifecycle cost can be reclassified as a compensable contract change. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

12. Set minimum supported-state rules

The project team should require maintained versions, security updates, vendor support or documented equivalent capability. The required output is a supported-state standard. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [6][21].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that a functioning system can remain exposed, unsupported and unfinanceable. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

13. Create upgrade bands

The project team should divide minor, medium and major changes by cost, risk, service impact and approval route. The required output is an upgrade-band matrix. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [22][23].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that every improvement can enter a slow bespoke variation process. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

14. Cap predictable technology exposure

The project team should define included refresh allowances, cost caps, exceptions and indexation for foreseeable change. The required output is a capped upgrade obligation. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [3][24].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that unbounded private exposure can inflate bids or deter finance. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

15. Price repeatable change

The project team should establish unit rates, margins, open-book rules, competition thresholds and validity periods. The required output is a pre-priced change schedule. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [22][25].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that small upgrades can accumulate opaque transaction costs and excessive margins. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

16. Benchmark technology cost and performance

The project team should compare total cost, service levels, energy use, availability and support with credible market references. The required output is a benchmarking protocol. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [26][27].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that headline hardware price can conceal migration, integration, licensing and operating cost. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

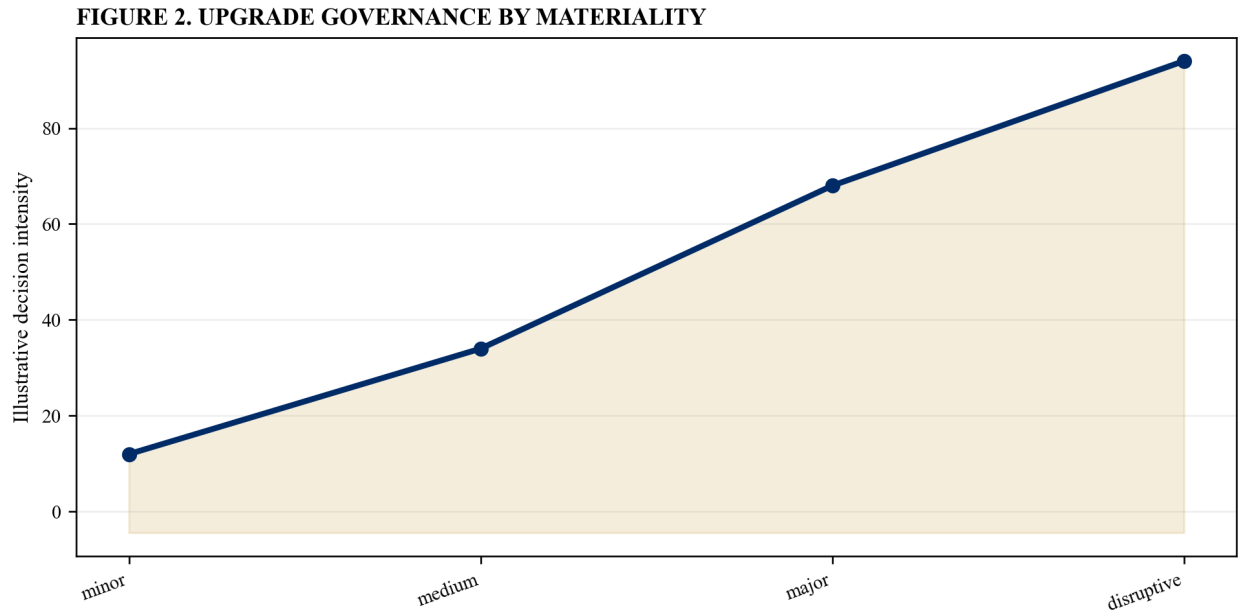
Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

Table 2. Upgrade bands

Band	Commercial treatment	Governance
minor	included baseline	operator notice
medium	unit rates or cap	joint approval
major	open-book or competition	authority approval
disruptive	economic-balance review	expert process

Illustrative analytical structure; verified project and jurisdiction evidence governs.

Figure 2. Upgrade governance by materiality



Illustrative analytical scenario; verified project evidence should replace values.

17. Use market testing where contestable

The project team should recompete separable services or components with transition protections and objective evaluation. The required output is a market-testing plan. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [25][28].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that incumbency and integration dependence can make periodic competition artificial. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

18. Govern authority-initiated change

The project team should define request, impact assessment, option appraisal, affordability, approval, delivery and acceptance. The required output is an authority-change protocol. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [22][29].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that policy-led upgrades can proceed without an agreed price or service consequence. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

19. Govern operator-initiated innovation

The project team should require a business case, compatibility evidence, service protection, savings calculation and approval route. The required output is an innovation proposal protocol. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [3][30].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that useful improvements can be delayed or private changes can transfer hidden risk. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

20. Share innovation benefits

The project team should allocate verified capex, operating savings, revenue, performance gain and residual value. The required output is a benefit-sharing waterfall. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [3][31].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that the party funding innovation can lack incentive when another party captures the gain. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

21. Allocate disruptive substitution risk

The project team should test whether technology changes demand, comparative cost, regulation or the relevance of the underlying service. The required output is a disruption-risk allocation. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [2][3].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that an efficient asset can become economically stranded through change outside the project. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

22. Define economic-balance safeguards

The project team should set thresholds, evidence and remedies where unforeseeable technology change causes material uncompensated impact. The required output is an economic-balance mechanism. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [3][29].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that minor market evolution can trigger opportunistic renegotiation or severe disruption can have no workable route. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

23. Control supplier concentration

The project team should map critical vendors, subcontractors, proprietary dependencies, financial health, support commitments and substitutes. The required output is a supplier-continuity register. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [32][33].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that a concession can depend on one vendor whose incentives and solvency are outside project control. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

24. Secure source and configuration continuity

The project team should use escrow, repository access, documentation, keys, build processes, configuration baselines and step-in rights. The required output is a technology-continuity package. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [34][35].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that the authority or replacement operator can own assets without the means to operate them. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

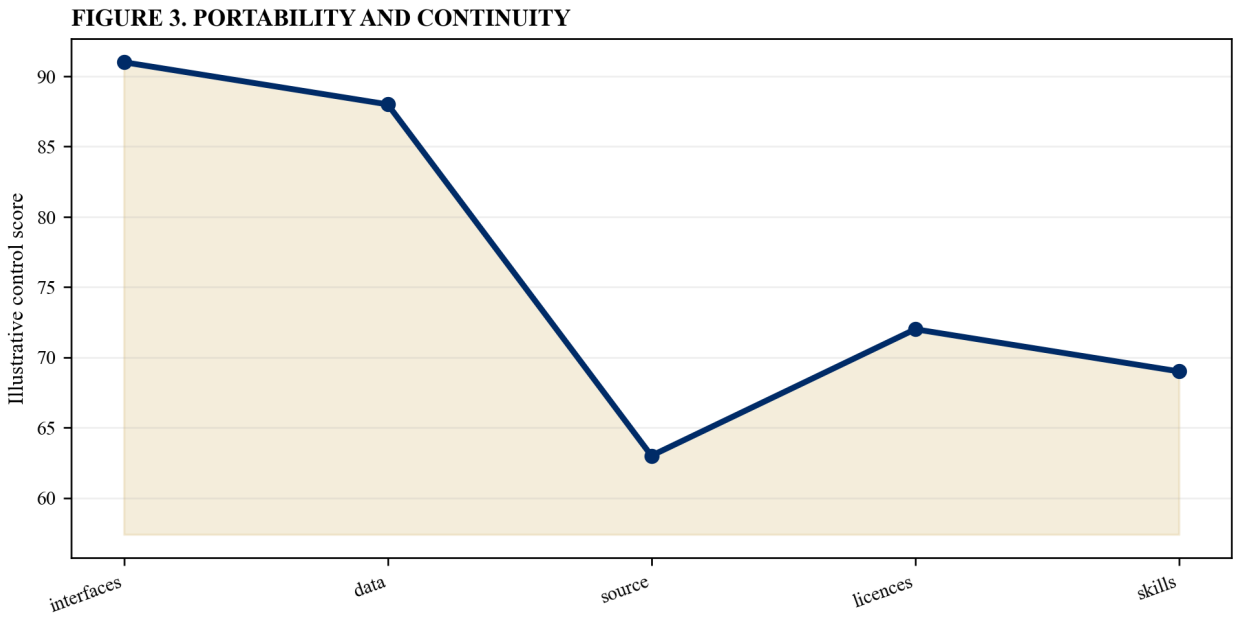
Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

Table 3. Continuity controls

Dependency	Control	Fallback
software	source and build access	escrow release
vendor	support and substitution	step-in
licence	transfer and survival	transition right
data	portable format	verified export

Illustrative analytical structure; verified project and jurisdiction evidence governs.

Figure 3. Portability and continuity



Illustrative analytical scenario; verified project evidence should replace values.

25. Govern software and licence rights

The project team should define term, users, environments, renewal, audit, transfer, termination and post-expiry operation. The required output is a licence-rights matrix. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [5][36].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that licence restrictions can block scale, transition, refinancing or handback. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

26. Protect data ownership and portability

The project team should define data classes, control, access, format, lineage, retention, export and deletion. The required output is a data-rights schedule. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [14][37].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that operational dependence can persist after contract rights end because usable data cannot move. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

27. Integrate cybersecurity lifecycle

The project team should require secure design, patching, identity, monitoring, vulnerability management, incident response and recovery. The required output is a cyber-lifecycle plan. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [21][38].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that technology refresh can introduce unmanaged exposure or unsupported systems can accumulate critical risk. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

28. Control artificial intelligence and automation

The project team should define approved uses, data provenance, model performance, human oversight, audit, update and fallback. The required output is an AI assurance schedule. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [39][40].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that adaptive systems can change behaviour without a traceable contractual decision. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

29. Model energy and compute economics

The project team should forecast power, cooling, connectivity, cloud, edge, storage and processing requirements. The required output is a technology operating-cost model. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [18][27].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that a technically current solution can become economically obsolete through changing unit cost. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

30. Build the integrated lifecycle model

The project team should connect refresh capex, maintenance, licences, efficiency, downtime, change cost and residual value. The required output is a technology-adjusted financial model. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [9][24].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that technical roadmaps and financing assumptions can use inconsistent replacement cycles. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

31. Stress correlated technology failure

The project team should combine vendor exit, cyber incident, support expiry, component shortage, migration delay and demand shift. The required output is a compound technology stress. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [32][38].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that single-variable sensitivities can understate simultaneous service and liquidity pressure. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

32. Fund refresh and major upgrades

The project team should compare lifecycle reserve, included opex, capex facility, authority payment, refinancing and shared investment. The required output is a technology-funding waterfall. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [24][31].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that necessary upgrades can lack cash even when contract responsibility appears clear. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

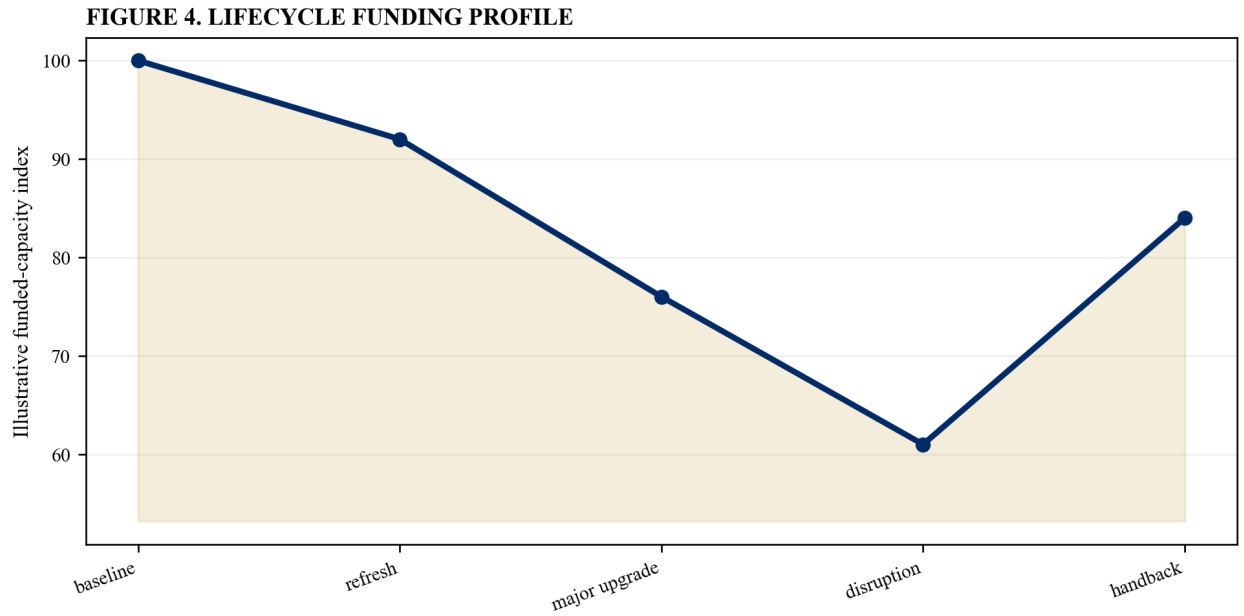
Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

Table 4. Illustrative lifecycle funding

Requirement	Share of lifecycle cost	Funding route
routine refresh	28	service payment
major upgrade	34	reserve and facility
authority change	23	change payment
disruption reserve	15	shared contingent layer

Illustrative analytical structure; verified project and jurisdiction evidence governs.

Figure 4. Lifecycle funding profile



Illustrative analytical scenario; verified project evidence should replace values.

33. Align debt with technology risk

The project team should test debt tenor, amortisation, reserve, covenant, insurance, step-in and residual-value assumptions. The required output is a technology-adjusted debt case. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [9][33].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that long debt can depend on short-lived technology and concentrated suppliers. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

34. Maintain a controlled document set

The project team should keep contracts, variations, specifications, registers, interfaces, licences, tests and acceptance records current. The required output is a shared technology contract record. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [4][28].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that parties can manage different versions of the operating bargain. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

35. Monitor leading indicators

The project team should track support horizon, patch lag, incidents, cost variance, performance gap, vendor health and substitution signals. The required output is a technology early-warning dashboard. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [6][19].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that intervention can begin only after service failure or emergency procurement. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

36. Define escalation and dispute routes

The project team should assign expert determination, technical panel, commercial negotiation, mediation and formal dispute paths. The required output is a technology dispute ladder. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [2][29].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that general dispute procedures can be too slow for time-critical upgrades. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

37. Plan transition and exit assistance

The project team should require inventories, documentation, data export, knowledge transfer, licences, suppliers and parallel operation. The required output is an exit-assistance plan. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [10][37].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that competition at expiry can fail because the incumbent controls operational continuity. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

38. Set technology handback condition

The project team should define supported state, remaining life, security, interoperability, documentation, licences and accepted technical debt. The required output is a technology handback standard. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [10][12].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that the authority can inherit assets that meet physical condition tests while being digitally obsolete. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

39. Rehearse replacement and recovery

The project team should test component substitution, vendor failure, data restoration, manual fallback and operator transition. The required output is a technology continuity exercise. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [34][38].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that contractual step-in rights can prove unusable during a real failure. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

40. Issue the technology-investibility certificate

The project team should reconcile service outcomes, architecture, lifecycle, upgrades, cost, rights, cyber, finance and handback. The required output is an auditable technology-investibility certificate. Record the accountable party, evidence, assumption, contractual right, service and cash-flow consequence, approval and review date [1][3].

Test the proposed treatment against user needs, asset and software inventories, vendor evidence, standards, architecture, interoperability, cybersecurity, competition, affordability, lender requirements and applicable law. Preserve the source, version, date, dependency and reason for every material judgement.

The principal risk is that approval can rest on isolated technical schedules without a coherent lifecycle bargain. Quantify effects on service quality, availability, lifecycle cost, demand, revenue, migration, operating resilience, debt-service coverage, equity returns, public liabilities and residual value. Compare the chosen response with credible refresh, modularity, procurement, transfer, reserve and shared-risk alternatives.

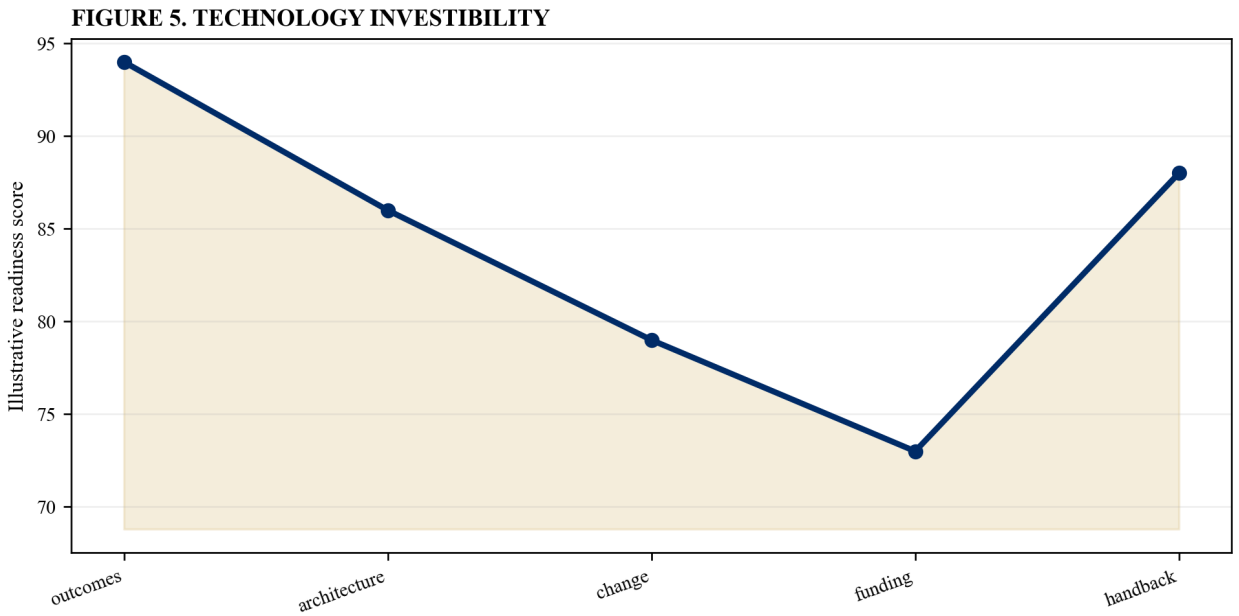
Translate the conclusion into output specifications, architecture constraints, lifecycle obligations, change rights, price and benefit rules, funding, monitoring, dispute, exit and handback provisions. Refresh it when service needs, standards, supplier support, security threats, technology economics or market alternatives change.

Table 5. Technology-investibility certificate

Decision	Evidence	Owner
service	outcome charter	authority
architecture	interfaces and standards	technical
lifecycle	roadmap and funding	operator and finance
exit	portable handback	joint

Illustrative analytical structure; verified project and jurisdiction evidence governs.

Figure 5. Technology investibility



Illustrative analytical scenario; verified project evidence should replace values.

References

[1] World Bank Group, PPP Reference Guide Version 3, https://ppp.worldbank.org/sites/default/files/2024-08/PPP%20Reference%20Guide%20Version%203.pdf

[2] World Bank PPP Resource Center, Managing Technological Change, https://ppp.worldbank.org/node/7616

[3] World Bank Group, PPP Contracts in an Age of Disruption, https://ppp.worldbank.org/sites/default/files/2024-04/10028%20-%20PPP%20Contracts%20in%20An%20Age%20of%20Disruption%20%28October%202023%29.pdf

[4] UK National Infrastructure and Service Transformation Authority, PFI Document Stocktake, https://www.gov.uk/government/publications/pfi-contract-management-document-stocktake/nista-pfi-contract-management-guidance-document-stocktake

[5] ISO, ISO IEC 19770-1 IT Asset Management, https://committee.iso.org/sites/jtc1sc7/home/projects/flagship-standards/isoiec-19770-12017.html

[6] ISO, ISO IEC TS 19770-10:2025, https://www.iso.org/standard/86588.html

[7] World Bank PPP Resource Center, Changing Technological Environment, https://ppp.worldbank.org/changing-technological-environment

[8] OECD, Digital Transformation of Public Procurement, https://www.oecd.org/en/publications/digital-transformation-of-public-procurement_79651651-en.html

[9] ISO, ISO 55001 Asset Management, https://www.iso.org/standard/83054.html

[10] World Bank PPP Resource Center, Contract Expiry and Asset Handover, https://ppp.worldbank.org/contract-expiry-and-asset-handover

[11] World Bank PPP Resource Center, Structuring the PPP Contract, https://ppp.worldbank.org/structuring-ppp-contract

[12] ISO, ISO IEC IEEE 42010 Architecture Description, https://www.iso.org/standard/74393.html

[13] NIST, Cybersecurity Framework 2.0, https://www.nist.gov/cyberframework

[14] European Commission, European Interoperability Framework, https://ec.europa.eu/isa2/eif_en/

[15] Open Group, TOGAF Standard, https://www.opengroup.org/togaf

[16] ISO, ISO IEC 19941 Cloud Interoperability and Portability, https://www.iso.org/standard/66639.html

[17] OECD, Digital Government Interoperability, https://www.oecd.org/en/topics/digital-government.html

[18] OECD, Going Digital Toolkit, https://goingdigital.oecd.org/

[19] World Bank, Digital Development, https://www.worldbank.org/en/topic/digitaldevelopment

[20] ISO IEC, Software Lifecycle Processes, https://www.iso.org/standard/63712.html

[21] NIST, Secure Software Development Framework, https://csrc.nist.gov/Projects/ssdf

[22] UK Treasury, PFI Change Protocol Principles, https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/759768/pfi_change_protocol_principles.pdf

[23] UK NISTA, PFI Contract Management Strategy, https://www.gov.uk/government/publications/pfi-contract-management-strategy/nista-pfi-contract-management-guidance-contract-strategy

[24] World Bank PPP Resource Center, Government Support, https://ppp.worldbank.org/government-support

[25] UK Cabinet Office, Sourcing Playbook, https://www.gov.uk/government/publications/the-sourcing-playbook

[26] UK NISTA, PFI Benchmarking and Market Testing Guidance, https://www.gov.uk/government/collections/pfi-contract-management-guidance

[27] European Commission, Life Cycle Costing, https://green-business.ec.europa.eu/green-public-procurement/life-cycle-costing_en

[28] World Bank PPP Resource Center, Managing PPP Contracts, https://ppp.worldbank.org/managing-ppp-contracts

[29] World Bank PPP Resource Center, Renegotiation, https://ppp.worldbank.org/renegotiation

[30] OECD, Public Procurement for Innovation, https://www.oecd.org/gov/public-procurement/innovation/

[31] World Bank PPP Resource Center, Translating Risk Allocation into Contract Structure, https://ppp.worldbank.org/translating-risk-allocation-contract-structure

[32] OECD, Managing Risks in Public Procurement, https://www.oecd.org/gov/public-procurement/risk-management/

[33] ISO, ISO 22301 Business Continuity, https://www.iso.org/standard/75106.html

[34] UK National Cyber Security Centre, Supply Chain Security Guidance, https://www.ncsc.gov.uk/collection/supply-chain-security

[35] ISO, ISO IEC 27001 Information Security, https://www.iso.org/standard/27001

[36] World Intellectual Property Organization, Software and Intellectual Property, https://www.wipo.int/software/en/

[37] OECD, Enhancing Access to and Sharing of Data, https://www.oecd.org/en/publications/enhancing-access-to-and-sharing-of-data_276aaca8-en.html

[38] NIST, Guide for Cybersecurity Event Recovery, https://csrc.nist.gov/publications/detail/sp/800-184/final

[39] OECD, AI Principles, https://oecd.ai/en/ai-principles

[40] NIST, AI Risk Management Framework, https://www.nist.gov/itl/ai-risk-management-framework

About the Author

Authored by Chennakeshav Adya, Independent Researcher

Chennakeshav (CK) is a corporate finance and investment banking executive with 25+ years of global experience in deal origination, structuring and execution across M&A, growth capital and corporate strategy. He has led value-creation mandates for founders, corporates and funds; bridging the boardroom view to hands-on execution and close.

His career spans Morgan Stanley, HSBC, Lloyds Banking Group, EWEC, ADQ portfolio companies and Emirates Growth Fund, across TMT, real estate, fintech, deeptech, cleantech, infrastructure and energy. He has partnered with C-suite leaders, private equity and venture funds, sovereign wealth funds and family offices to finance complex fund raises and scale-up ventures, and has led M&A due diligence, post-merger integration and business-transformation initiatives to create value.

At Matchpoint Partners he is Managing Partner, leading the firm's corporate finance, M&A and capital-raising practice. He holds an MBA from London Business School, an engineering degree from VTU and a Master of Laws (LLM, in progress) from UCL London.

An active start-up mentor, CK mentors at Techstars, DIFC FinTech Hive, Startup Grind, Founder Institute and IN5, serves as Entrepreneur Mentor in Residence (EMiR) at London Business School, and judges the Entrepreneurship World Cup.