WHITE PAPER · SUPPLIER & INFRASTRUCTURE STRATEGY

Designing for Leverage

Managing service providers without becoming dependent on them

A customer holds real negotiating power twice: when the provider is competing for the business, and when the contract is nearly over. What shifts the balance in between isn't the contract — it's dependency, accumulated one convenient decision at a time. This is how we designed against it across 60 countries.

BY **JACK NASH** 60 COUNTRIES · CADBURY & BT GLOBAL SERVICES ABOUT 13 MIN READ

THE ARGUMENT IN BRIEF

- A customer has real negotiating power at two moments: **when the provider is competing for the business, and when the contract is nearly over**. In between, the balance shifts.
- What shifts it is not the contract. It is **dependency** — services accumulated one convenient decision at a time until removing any of them means touching all of them.
- I came to call these **sticky services**. Each is defensible on its own merits. Collectively they become an exit barrier, and the provider never has to mention it.
- The response was to **commoditize our providers**: bounded responsibilities, deliberate separation between transport and security, and a blast radius small enough that changing one service changed one service.
- Leverage is not created at the negotiating table. **It is created in the architecture** — and its purpose is to keep the provider in selling mode for the whole term, not just at the ends.

PART I **The Two Moments of Power**

Introduction

Over years of managing large technology environments and global service-provider relationships, I noticed that a customer typically holds significant negotiating power during two periods of a

contract: when the provider is trying to win the business, and when the agreement is approaching expiration.

Between those two points, the balance of power tends to move toward the provider.

The reason is not usually the contract. Contracts are negotiated carefully, by people who are good at it, and they generally say what both parties intended. The reason is dependency — which accumulates quietly, after the contract is signed, through decisions that each look sensible at the time.

This paper describes how that accumulation happens, what it costs, and the architectural discipline I used to prevent it across a 60-country environment. The short version: providers were deliberately commoditized, responsibilities were bounded, and the organization retained the credible ability to move any single service without disturbing everything around it.

The conversation that changed my thinking

Cadbury was a major global customer of British Telecom Global Services. We operated across roughly 60 countries, including many developing markets where the local IT organizations worked with considerably smaller budgets than their counterparts in North America and Western Europe.

One recurring issue was the cost of Cisco equipment supplied through our global BT relationship. Our developing-country operations were struggling to justify the prices, and I raised it with our BT Global Services account lead. Eventually I asked him directly:

> “Don’t you want to sell to these countries?”
>
> My question

His answer was immediate.

> “I don’t need to. I already have a contract.”
>
> BT Global Services account lead

I did not like the answer. But the longer I sat with it, the more useful it became — because from where he sat, it was entirely correct.

BT had already won. The agreement was negotiated. The contract was signed. Services were deployed. The infrastructure depended on them. Why would a provider behave as though it were still competing for business that we had already structured in a way that made it difficult to take away?

He was not being cynical. He was describing the incentives we had built for him. And that reframed the question I had been asking.

> **The question stopped being *how do we negotiate a better contract?* and became *how do we design the relationship so the provider never stops competing?***

The two moments of customer power

Before a major contract is signed, the customer has leverage, and everyone behaves accordingly. Providers compete. Prices fall. Executives attend meetings. Exceptions get approved. Engineering resources materialize. Contract terms turn out to be negotiable after all.

Then the agreement is signed, and the economics begin to change almost immediately. The provider holds revenue under contract. The customer begins accumulating switching costs. As more of the environment comes to depend on that provider, replacing it becomes progressively harder and the conversation becomes progressively less urgent on their side.

Then, as expiration approaches, something entirely predictable happens. The customer becomes important again. Executive sponsors reappear. Pricing improves. New technologies are proposed. Account teams become remarkably responsive.

The provider has returned to selling mode — because the customer has once again acquired a credible ability to leave.

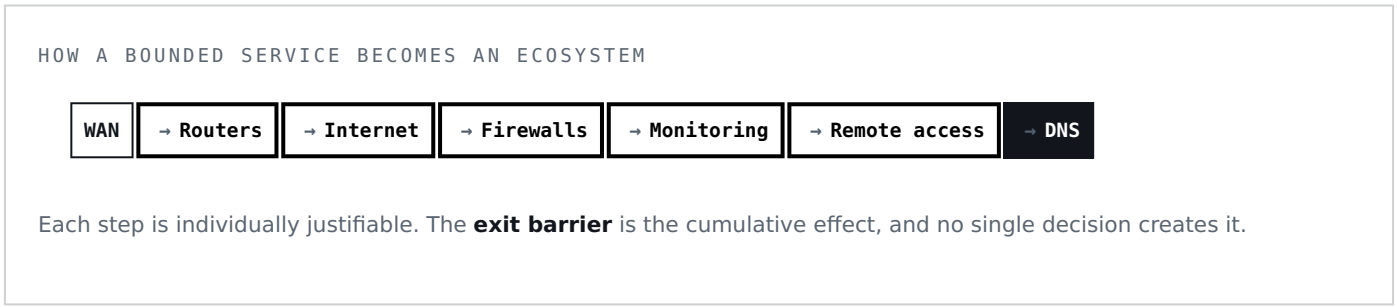
My objective was to stop that selling mode from disappearing in the years between the two.

PART II **How Dependency Accumulates**

Sticky services

The most effective way providers gain leverage is not through contract language. It is through the accumulation of adjacent services.

A provider begins by supplying WAN connectivity. Then managed routers. Then Internet access. Then firewalls. Then security monitoring. Then remote access. Then cloud connectivity. Then DNS and other supporting services.



Every one of those additions can be presented as a benefit, and usually is a real one. There may be bundled discounts. Simplified billing. Genuine operational efficiencies. A perfectly sound argument for having one provider accountable for the whole thing rather than three providers pointing at each other.

But every addition also makes the provider harder to remove. The services become sticky because pulling on one begins pulling on everything attached to it. What started as a set of convenient purchases has become an exit barrier — and at no point did anyone decide to build one.

> **Eventually the customer is no longer buying individual services. It is buying an ecosystem it cannot easily leave.**

Sticky services create commercial leverage

The danger is not technical complexity. It is the quiet transfer of negotiating power.

Suppose a provider supplies the global network and also hosts the enterprise firewalls protecting it. The customer becomes dissatisfied with the network service — pricing is no longer competitive, or quality has slipped, or the business now requires a different architecture.

Changing network providers should be a network decision. It no longer is. The customer must now work out what happens to its firewalls.

- Do they stay with the previous provider? *Can* they?
- How are they connected once the transport changes?
- Who owns the configurations?
- What happens to routing? To remote access?
- What other security services depend on them?

A decision that should have affected one component has acquired a much larger blast radius. That complexity benefits the incumbent, and it does so without anyone having to be difficult about it.

THE DEPENDENCY TRAP

The provider never has to say “you cannot leave.”

The architecture says it for them.

Limit the blast radius

This became one of the fundamental objectives behind my supplier architecture: a change in one service should affect as little of the surrounding environment as possible.

The reason is not tidiness. It is that the infrastructure has to move with the business, and businesses do not hold still. They reorganize. They acquire. They divest. They enter countries and leave them. Technology changes, pricing changes, requirements change.

A service-provider strategy that makes infrastructure difficult to change is therefore working against the business it exists to support.

Bounded providers
+ Changing the network changes the network
+ Security posture survives the transition
+ Migration is scoped and schedulable
– *Loses some bundled discount*

Accumulated provider
+ Single point of accountability
+ Simpler billing and governance
– *Changing the network changes five things*
– *Migration becomes a transformation project*

If I wanted to change a network provider, I wanted to change the network provider. I did not want that decision to automatically drag in the firewall architecture, the security services, the monitoring platform, and the remote-access solution.

Separation between providers created commercial and technical fault domains. The effect is the same as modularity in systems engineering: components can change without requiring the whole system to change with them.

One provider, one clearly defined responsibility

So I began setting rules about which services could be combined under a single provider. The clearest example was network and security.

If a provider supplied our local access or global network services, that provider could not also host our network firewalls.

There was nothing wrong with their firewall capability. Several of them were entirely competent at it, and in a purely technical evaluation they might well have won. The issue was leverage. Allowing one provider to control both transport and security would have made replacing that provider substantially harder, and would have put our security posture inside the blast radius of every commercial disagreement about connectivity.

I wanted each provider's responsibility to have clear boundaries. A provider could become exceptionally good at the service it was assigned. What it could not do was spread through the architecture until removing it became a transformation project.

This meant occasionally declining attractive bundled offerings — which is not a comfortable conversation to have with a finance team looking at the discount. A bundle can save money today and cost considerably more tomorrow in flexibility that no longer exists.

Keep the provider in selling mode

This changed the character of the relationships. Providers understood that winning one service did not automatically deliver the next one. They also understood that losing our confidence did not mean waiting three years for a contract expiration event before anything could happen. Their service was replaceable, and they knew it.

That mattered more than any clause we could have negotiated.

> **A provider that believes a customer cannot realistically leave behaves differently from one that knows the customer can.**

Selling mode is not primarily about price. It means continuing to demonstrate value. Remaining responsive. Solving problems rather than routing them. Competing for the next piece of work. Treating the existing relationship as something that still has to be earned.

What it prevents is the state I would describe as the tail wagging the dog — where the supplier's architecture, commercial model, or internal processes begin determining what the customer is

able to do. We were the customer. The infrastructure existed to support our business. Providers needed to fit into that architecture rather than the reverse.

Architecture creates negotiating power

The larger lesson was that leverage is not created exclusively at the negotiating table.

Procurement can negotiate aggressively. Legal can secure favorable language. Executives can escalate. All of that is worth doing. But none of it is as powerful as being able to say, credibly:

> **“If this no longer works for us, we can move it.”**

For that sentence to carry weight it has to be technically true. Which means avoiding unnecessary dependencies, maintaining documentation, controlling credentials and configurations, understanding integration points, separating infrastructure functions, and deliberately resisting services whose convenience is purchased with future portability.

Architecture therefore becomes part of the organization’s commercial strategy. Every technical dependency carries a potential commercial consequence, and the technical decision is usually made years before anyone experiences the commercial one.

Own the architecture

All of this rests on a single principle: the customer must own the architecture.

Providers can operate pieces of it. They can supply expertise, make recommendations, and manage infrastructure well. What they should not become is the only organization that understands how the environment works.

Once a supplier holds knowledge, access, or control that the customer cannot reproduce elsewhere, that knowledge is itself a sticky service — and it is the stickiest kind, because it does not appear on any invoice and no contract clause can dislodge it.

This matters most in multi-provider environments, which can become genuinely chaotic when nobody owns the whole. The answer to that chaos is not to consolidate everything under one provider. The answer is strong architectural ownership inside the customer organization.

> **We owned the architecture. Providers supplied components of it. That distinction did more work than any contract term.**

The exit test

Every significant provider decision should include an exit test. Before asking only how easily a service can be implemented, ask how easily it can be removed.

- Who owns the equipment, and who owns the configuration?
- Who controls the administrative credentials?
- Who controls addressing, routing, and DNS?
- Who owns the security policies?
- Could another provider assume this service, and can it be migrated independently?
- What else must change if this changes?
- How long would migration realistically take, and what business functions would it affect?
- What proprietary technologies or processes create lock-in?

A service that is easy to buy and extraordinarily difficult to remove may be far more expensive than its contract price suggests. The ability to exit is not a procurement concern to be handled later. It is an architectural requirement, and it is cheapest to satisfy before the service is deployed.

Consolidation is not always efficiency

Supplier consolidation is routinely presented as an efficiency strategy, and the advantages are real. Larger commitments produce discounts. Billing simplifies. Governance looks easier. Escalations involve fewer participants.

But consolidation also concentrates power, and the analysis usually stops before that appears on the page.

ASKED	How much will we save by giving this provider more of our business?
ALSO ASK	How much leverage are we giving away in exchange for that saving?

If consolidating five services saves ten percent and makes the supplier extraordinarily difficult to replace, the organization may have traded a visible short-term discount for a much larger long-term liability — one that will not be measured, because nobody prices the negotiation they were unable to have.

The price of a service and the cost of dependency are not the same number.

Strategic does not mean irreplaceable

None of this is an argument against strategic suppliers. Some providers will inevitably become important, through expertise, geographic reach, technology, or simply the quality of their people.

The mistake is confusing a strong strategic relationship with dependency.

A strategic provider should remain strategic because it continues to perform, continues to innovate, and continues to price competitively — because it keeps earning the business. It should not remain strategic because removing it has become too painful to contemplate. Those two situations look identical on an org chart and are completely different in a negotiation.

PART V **Dependency Management**

From vendor management to dependency management

Traditional vendor management concerns itself with contracts, service levels, performance reviews, escalations, and renewals. All of it matters, and all of it addresses only part of the problem.

The deeper discipline is dependency management: continuously understanding how much control the enterprise has surrendered to each provider, and whether that level of dependency is still acceptable.

- **Own the architecture.** Providers supply components of it.
- **Keep provider responsibilities clearly bounded.**
- **Avoid unnecessary sticky services**, however convenient each one appears.
- **Limit the blast radius** of any provider change.
- **Maintain technical and commercial portability.**
- **Keep configurations, credentials, documentation, and institutional knowledge** under customer control.
- **Evaluate exit complexity before purchasing**, not at renewal.
- **Do not confuse bundled pricing with long-term value.**
- **Maintain credible alternatives** throughout the term.
- **Keep providers in selling mode.**

The ability to move

The most important lesson I took from managing these relationships is that customer leverage rests on the ability to move.

Before a contract is signed, the provider knows the customer can choose someone else. Near renewal, the provider knows the customer can leave. The entire challenge is making sure the provider understands those alternatives still exist during the years in between — and that is not a negotiating skill. It is an architectural one.

At Cadbury we deliberately limited supplier dependencies and resisted the accumulation of sticky services. Providers were given clearly defined responsibilities. Infrastructure layers were separated where concentration would have created excessive dependency. The architecture remained ours. That let us make changes that affected one area rather than unnecessarily affecting many, and it meant our providers understood we retained the ability to move.

None of this is about adversarial supplier relationships. Quite the opposite — the best provider relationships are the ones where both organizations continue to see value in working together. But that relationship should remain a choice for both parties, and a customer should never arrive at the point where a provider can reasonably say that it doesn’t need to compete because it already has a contract.

The best defense against that sentence is not a tougher contract. It is an architecture that ensures there is always another option.

Own the architecture. Bound the provider.

Keep the ability to move.

A note on the account. This is a first-person recollection of work carried out during my time at Cadbury and in the global service-provider relationships of that period. Quoted remarks are recalled rather than transcribed. Company and product names are used descriptively and belong to their respective owners.