Open Source Back-End Services: A Migration Guide for Directory, Storage, Mail, and Real-Time Communication

Stéfane Fermigier

Draft working paper v1.0, 2026-08-30. Guide 1 of 2. The companion volume, `workstation-wp.md`, covers end-user applications and the managed Linux desktop. Product statuses are stated as of the date given in §1.4; entries marked † were moving at that date and need re-checking before a procurement decision.

Abstract

This guide addresses a question that most organisations now face in some form: what would it take to run the collaboration back end (directory, file services, mail and calendar, document collaboration, and real-time communication) on open source software, and what does that cost. It adapts three public studies, two commissioned by the Swiss Federal Chancellery in 2024 and one by the German state of Schleswig-Holstein in 2022, from their original public-administration setting to organisations generally, and updates their product assessments.

The guide departs from its sources in three ways. It treats **supplier depth** as a first-class selection criterion alongside function and maturity, because a single-vendor product replaces the identity of a dependency without changing its structure, and because only a plural supplier base turns an open licence into an actual ability to change your mind. It separates **directory of record, authentication, and device policy** into three procurement decisions instead of looking for one product that covers all three. And it treats **exit cost rather than licence cost as the object**, following the Schleswig-Holstein strategy's observation that the cost of client access licences is best understood as the price of a dependency.

It also covers what the source studies leave out, which is most of the engineering. Product selection occupies §§4 to 11, including the AI assistance question that postdates every source here and that supplies the mechanism behind the price escalation the economics rest on. §12 takes the hosting decision, on which most of the sovereignty turns and which a guide discussing only software would miss. The work then occupies §§13 to 19: reference architectures and sizing, data-migration mechanics, coexistence patterns, backup and continuity, security baselines, operations, and what to put in a contract. A migration fails in those chapters, and succeeds in them.

The headline finding is uncomfortable for both the enthusiast and the sceptic. Credible open-source options now exist in every domain examined, and in two of them (groupware with native Outlook compatibility, and integrated sovereign suites) the position improved materially after the source studies were written. No source in the corpus supports a claim of cost savings, and the most rigorous of them concludes cost parity over a long horizon. The case for migrating rests on exit cost, price-escalation exposure, and optionality, all of which are real and none of which is a discount.

Executive summary

For readers who will approve this work without executing it. Everything below is developed and sourced in the sections referenced.

The decision is not binary

There are three defensible positions, and most organisations that believe they are choosing between the first and the last should be choosing the middle one.

Position	What you do	What it buys	What it costs
Stay and negotiate	Nothing structural. Use the threat of the other two.	Nothing, unless the threat is credible. An incumbent can tell.	Cheap, and it decays: leverage you never demonstrate stops being believed.
Build credible optionality	Federate identity, move to open formats, run a working reserve (§16.5). No full migration.	A real exit threat, the knowledge of your own dependencies, and continuity cover. Most of §2.6′s benefit.	A fraction of a migration. A few months.
Migrate	Replace the stack, domain by domain, over years.	The above, plus the licence line and the escalation exposure removed.	See below. Investment-shaped: worse before better.

The middle position is under-used and is the right recommendation for a large share of organisations. It is also the correct first phase of the third position, so choosing it costs nothing if you later go further.

What migration costs, in one number

The model in §21 refuses to guess what you pay today, and instead derives the figure that needs no guess: **the incumbent cost per seat per year at which migration breaks even.**

Estate	3 years	5 years	10 years
250 seats	401	269	170
2,500 seats	293	229	182
25,000 seats	207	160	126

EUR per seat per year, back-end stack, undiscounted. If your blended per-seat cost is above the figure, migration is cheaper over that horizon; below it, the incumbent is. **These figures run on placeholder effort, day-rate and productivity parameters, so they demonstrate a method and do not state a cost.** Two adjustments move them in opposite directions and both belong in a real case: letting the incumbent's price rise at the measured rate for a locked base lowers the ten-year figure by about a third, and discounting the comparison raises it again, by a tenth at four percent (§21.5).

Two things follow. **Migration cost is close to independent of estate size**, so the fixed cost of doing it at all dominates, and small organisations are the ones for whom the arithmetic is hardest. And **the three-year comparison usually favours the incumbent**, correctly, because the investment phase sits inside that window (§2.3). If your board evaluates on three years, present the ten-year column beside it and explain why.

How long

Indicative, and derived in §20.7. No completion deadline for user migration, and a planned programme regardless:

- **Foundation, 6 to 12 months.** Inventory, identity federation, open formats, the reserve. Reversible; delivers value alone.
- **Substitution, 12 to 30 months.** Files, editing, mail and groupware. The bulk of the work and the disruption.
- **Consolidation, 12 to 24 months.** Real-time communication, telephony, print, and decommissioning the sources.

Total three to five years for a mid-sized estate, against four to seven observed in the two documented field cases. Telephony and secure print come last, and both are integration projects (§9.5, §11).

The five things that actually decide it

1. **Your incumbent agreement's renewal date and exit terms.** Frequently a harder constraint than any technology. Reducing seats mid-term often saves nothing (§21.4).
2. **Whether you keep Outlook.** It determines the groupware choice and the sequencing of the largest domain (§7.1).
3. **Where you will run it.** Open-source software on infrastructure you were trying to become independent of changes the licence bill and little else (§12).
4. **Whether you can staff a platform capability**, or will buy the stack operated. Both are legitimate; pretending is not (§18.4).
5. **Whether the sponsorship outlasts a change of leadership.** The only risk rated catastrophic in the source corpus, and the mechanism that ended the best-known programme in this field (§22.1).

What remains hard

Four problems remain hard: tenant-wide AI assistance grounded on your own content and permissions (§10.3); secure pull-printing with badge release (§11); enterprise telephony as a whole (§9.5); and calendar free/busy across the boundary during coexistence (§7.4). None is a reason not to start; each is a reason to sequence around it.

What no source supports

A claim of savings. Three of the six source documents contain no cost figures at all, and the most rigorous concludes cost parity over a long horizon with savings only after several years (§21.1). The case rests on exit cost, escalation exposure, and optionality. Presenting it as a discount invites a comparison it will lose.

---

1. Scope and use

1.1 What this guide covers

The server-side services that a Microsoft-centred organisation typically runs: Active Directory or Entra ID; Windows file services, SharePoint, and OneDrive; Exchange or Exchange Online; SharePoint's and Office Online's collaborative editing; and Teams or Skype for Business for chat, meetings, and telephony. Printing is covered briefly because it recurs as an unsolved case.

The companion guide covers what runs on the user's machine. The split follows the sources, which decompose the problem into three independent steps: back-end services; client applications; and the client operating system. That decomposition is the single most useful inheritance from the Swiss studies, because each step can be taken alone and each delivers value alone.

1.2 Who it is for

Any organisation with a managed estate, from a few hundred seats upward. The source studies are public-sector documents and carry public-sector assumptions: procurement law constrains product selection, parliamentary cycles constrain project horizons, and the buyer is unusually able to tolerate long payback. Those assumptions are stated where they matter and set aside where they do not. A private organisation can name a product without running a tender, and generally cannot wait seven years for a payback, which changes the sequencing advice in §20 considerably.

Three organisation profiles recur in the guide and are used throughout §13 and §21:

- **Small.** Around 250 seats, one or two sites, a small internal team, most operations outsourced or bought as managed services.
- **Medium.** Around 2,500 seats, several sites, a dedicated infrastructure team, mixed internal and external operations.

- **Large.** Around 25,000 seats, many sites, specialist teams per domain, an existing platform capability.

These are reference points and not thresholds. The break that matters is whether the organisation can run a platform capability at all, which is discussed in §18.4; seat count is secondary.

1.3 How to read the recommendations

Verdicts run on a four-point scale:

- **Ready.** Deployable now for the stated role at organisational scale, with a supplier market behind it.
- **Viable with conditions.** Works, subject to the named conditions. The conditions are the point; read them.
- **Emerging.** Promising, and a candidate for a pilot. Not yet a safe primary bet for a production estate.
- **Not yet a replacement.** Does not cover the role. Named so you know it was considered.

These verdicts are about fitness for a role in a managed estate. They are not quality judgements about the software, and a project can be excellent and still be Emerging here.

1.4 State of information

Product assessments are current to **January 2026** and were re-checked against the source studies. A verification pass against vendor sources was outstanding when this draft was written, and claims that were moving at that date carry a † marker. Prices throughout are the published list prices recorded in the February 2024 Swiss study, reproduced with that provenance and that date. They are stale by construction and are used to demonstrate a method; they do not state a cost.

Sizing figures in §13 are planning heuristics for first-pass capacity estimates. They are engineering rules of thumb offered to start a conversation with a supplier, and none of them substitutes for a load test against your own data.

2. What you are buying out of

2.1 The dependency is not the licence fee

The most economically literate passage in the corpus is the Schleswig-Holstein strategy's treatment of client access licences, which reframes the whole exercise.

A CAL is charged for the use of a service that has already been licensed for its operation, so that "many services have to be licensed twice" [Staatskanzlei Schleswig-Holstein 2024]. CALs are usually acquired for a whole infrastructure and not to measured need, because metering them at scale costs more than the licences save. The strategy calls them "an artificial block of costs that shows how dependent one is on a particular supplier's products."

> A supplier will only dispense with an artificial product such as access licences if the customer enters a stronger bond of a different kind. It is understandable that the loss of this type of licence makes a switch to cloud infrastructures all the more attractive. [Staatskanzlei Schleswig-Holstein 2024]

Relief from an artificial cost is being sold in exchange for a deeper dependency. The same document makes the parallel point about bundling: a package priced below the sum of its parts increases dependence, "because 'the little bit left' is included in the package 'free of charge' and there is no need to look for an alternative to it." A bundle suppresses the search for alternatives in adjacent categories, which is a lock-in mechanism operating on procurement attention.

2.2 What the price series shows

Schleswig-Holstein records that German federal expenditure on Microsoft products rose "more than fourfold" over roughly a decade, and observes that this "cannot be explained solely by the addition of new services and workplaces" [Staatskanzlei Schleswig-Holstein

2024]. The diagnosis offered is structural: the buyer sits in a monopolistic market "characterized by technological dependence; there is no real freedom of choice, and the cost of switching is very high."

That is the exposure a migration addresses. It is exposure to the rate of change of price under conditions where the buyer cannot credibly threaten to leave. Migrating does not principally reduce this year's bill. It restores the threat.

2.3 The four-phase cost shape

Schleswig-Holstein's cost argument, stated in words and not in figures, is the most useful mental model in the corpus and matches what the German feasibility study concluded independently.

Investment. A multi-vendor strategy "requires higher expenditure in the short term", in training, new technical processes, operational readiness, and organisational adjustment. In this phase it "is more expensive than 'keeping the old infrastructure going'".

Plateau. Licence costs lose their significance because long-term support and development agreements replace them, and, decisively, "it is always possible to change software, developers, consultants and support."

Crossover. The incumbent model "begins to lose its short-term cost advantage, and finally it even becomes more expensive", as licence costs rise disproportionately.

Shock. When a technology discontinuity arrives (the strategy names video conferencing during the pandemic, and VoIP and instant messaging from 2010), the multi-vendor model needs fresh capital immediately, while the incumbent's costs arrive later and bring a new dependency with them, one that "enable[s] ever greater price rises in the future."

The conclusion is careful, and improving on it would weaken it:

> The objective of OSS is not to eliminate licence costs. The use of OSS helps to achieve the target of breaking out of the vicious circle of price increases. [Staatskanzlei Schleswig-Holstein 2024]

2.4 Open source as the instrument

The same document separates the mechanism from the ideology in a way this guide adopts throughout:

> As a first step, the monopolies of the IT infrastructure have to be broken. Transformation of the IT system and implementation of a multi-vendor strategy bring IT management back into an active role. … The use of open-source solutions is not absolutely essential at first. [Staatskanzlei Schleswig-Holstein 2024]

The ability to change supplier does the work. Open licensing and open standards are how that ability is secured and kept, which is why §3 weights supplier depth and exit path so heavily. An open licence on a product that only one company can realistically operate delivers less freedom of action than the licence suggests.

2.5 What open source does not fix

Schleswig-Holstein is unusually frank about this. Open source "never eliminates [dependency] completely." Development "may be stopped unexpectedly." And:

> It has to be admitted that the licence payments for proprietary software cover precisely this risk. They are used for financing development teams that produce the latest versions and update them. On the other hand, as a licensee one has no influence on the direction such development takes. [Staatskanzlei Schleswig-Holstein 2024]

Two cases in this guide's own domain make the point concretely; they are different failure modes that a single word would blur.

Kopano is enclosure. The groupware suite the Swiss study still listed as a live alternative in February 2024 reached end of life on 31 March 2025, and its successor, Kopano Cloud, is not offered under an open-source licence. An organisation that selected it on openness grounds lost that basis without being consulted, and there is no fork to move to.

ownCloud is something else, and calling it enclosure would be wrong. Acquired by Kiteworks in 2023, its core Infinite Scale engineering team left in January 2025 and forked the product (§6.3). The acquirer has since moved in the opposite direction from enclosure: in May 2026 it established an open-source program office under the ownCloud brand, relicensed over a hundred projects to Apache-2.0, retired the contributor licence agreement in favour of a developer certificate of origin, and published a governance charter. The **people** moved, and with them the community's centre of gravity. That is key-person and team risk operating on a codebase, and it is at least as common as enclosure while being much harder to see coming from a licence or a balance sheet.

ONLYOFFICE is a third mode again: licence overreach. The software is published under the AGPLv3. Around May 2021 its publisher added terms under section 7 of that licence requiring that the product logo be retained on redistribution, while separately declining to grant any trademark rights in that logo. In April 2026 both the Free Software Foundation and the Software Freedom Conservancy published analyses concluding that the logo requirement exceeds what section 7(b) permits, that it therefore constitutes a "further restriction" which recipients may remove, and that the combination with the withheld trademark rights produces a contradiction: a fork must keep a mark it has no licence to use. §8.3 sets out the dispute and its practical scope.

The failure mode here is neither enclosure nor departure. The licence remained open the whole time. It compromised the **practical usability of the freedoms it grants**, through terms buried in a licence file and enforced aggressively when someone relied on them. The Conservancy's warning is the part that generalises: an organisation exercising a right it clearly holds may still face public condemnation and possible litigation, and the prospect of that is itself the deterrent.

Three modes, then, and none of them is detectable from a licence badge: **enclosure** (the licence changes), **departure** (the people leave), and **overreach** (the licence stays and its terms are stretched). C4 and C5 exist to detect all three in advance, and all three are more common failure modes than any technical one.

The counterpart obligation follows: budget that was licence fees becomes "development orders or contracts for support". An organisation that migrates and then funds nothing has moved its risk without reducing it. The German study makes the same point from the operational side, warning of "die Gefahr des 'ungewollten' Erstellens eines 'Forks'" where local adaptations drift away from upstream and their maintenance becomes a permanent internal cost [Kern et al. 2022].

2.6 The three things you are actually acquiring

Migration business cases usually claim the wrong benefit:

1. **A credible exit threat**, which disciplines the price of everything you continue to buy, including from the incumbent if you keep part of the estate.
2. **Architectural knowledge of your own systems.** An organisation that has migrated its identity layer knows what depends on it. Most organisations do not, and discover it during an incident.
3. **The ability to fix things yourself or pay someone to.** This is worth little until the day it is worth everything, which is the argument for the sovereign reserve in §16.5.

Cost reduction, if it arrives, arrives late and is a consequence of the first item, never an independent benefit.

Where this frame comes from. Nothing in §2 is new economics, and saying so locates it for a reader who wants to check it. The mechanism is switching costs, whose canonical treatment is Klemperer (1995) with the field survey in Farrell and Klemperer (2007): where leaving is costly, an incumbent can price above the competitive level up to the cost of leaving, and a buyer who looks only at this year's invoice systematically under-weights that ceiling. Shapiro and Varian (1998) states the same result for practitioners. The regulatory record now

measures it: the UK competition authority's cloud market investigation found the two largest providers earning returns sustained above their cost of capital while fewer than one percent of customers switched provider in a year (CMA 2025), which is what a priced switching cost looks like from outside the firm. The horizon argument of §21.5 belongs to a second literature, the treatment of irreversible investment under uncertainty in Dixit and Pindyck (1994), which is also where the discounting of §21.5 gets its vocabulary: a migration buys an option, pays for it up front, and should be valued accordingly.

3. The evaluation framework

The source studies score products on function and maturity, with the Swiss studies adding a mechanical mapping from OpenSSF Best Practices badge tiers onto a SWOT grid. That badge mapping is weak evidence: it measures a project's declared process hygiene, and in the Swiss backend study it produced the result that Samba's 97 percent passing score appears as a *weakness* while a project with no badge at all appears only as a *threat*. This guide uses five criteria instead.

C1. Functional coverage. What proportion of the incumbent's used functions the option covers, assessed against the functions actually in use, which is a different quantity from the functions installed. §20.2 develops this distinction, which the French study makes better than anyone.

C2. Maturity at scale. Evidence of operation at a comparable seat count, in a comparable configuration, by someone who will say so publicly.

C3. Manageability. Whether the option can be provisioned, configured, monitored, backed up, and upgraded by automation, and whether it integrates with the identity layer. An option that requires manual care per instance fails at estate scale regardless of its features.

C4. Supplier depth. How many organisations can competently support, operate, and if necessary modify this software, independently of each other. This is the criterion the source studies omit and the one most predictive of whether an open licence yields real freedom of action.

C4 has a sub-question that a licence check cannot answer: **is this open source, or is it source that has been published?** The two look identical in a repository listing and differ completely in what they let you do. The test is whether development actually happens in the open:

- Are there merged contributions from people the vendor does not employ, recently and routinely?
- Is the public repository the one the developers work in, or a mirror of an internal one?
- Is there a public issue tracker where the vendor's own engineers argue about design, or only a bug inbox?
- Has anyone outside the vendor become a maintainer?
- Are there documented, honoured interfaces for third parties to integrate against, and does the vendor accept work on them?

A project failing most of these can be perfectly good software under an impeccable licence, and it still concentrates all capability in one organisation. A fork of it starts with the code and none of the people who understand it, which is the difference between a fork you could actually run and a fork you could merely publish. Score C4 on the answers. The licence will not tell you.

Apply this evenly. Several products recommended in this guide are single-vendor, and the question is whether anyone else *could* write it. One company writing most of the code is common and often fine.

C5. Exit path. What it would cost to leave *this* option later. Standard protocols and documented formats reduce it; proprietary extensions, undocumented schemas, and single-

implementation protocols raise it. Applying the criterion to the replacement as well as to the incumbent is the discipline that prevents a migration from simply relocating the problem.

C5 also covers the exit path of last resort, which is **forking**. That backstop is worth less than it appears if the licence carries additional terms that make an independent build hazardous to publish, so the check is concrete: **read the licence file itself.** An SPDX identifier in a repository description tells you what the project calls its licence. Section 7 of the (A)GPL family permits certain additional terms, those terms live in the text, and they are occasionally both material and introduced without announcement. §8.3 works through a case where they were. As the analysis of that case puts it, "licensed under AGPLv3" tells you less than you may think.

3.1 Control, and how to weigh it

Each option's table entry records where control sits. Control has three components that often point in different directions, and collapsing them into a single flag loses the information that matters:

- **Legal domicile** of the entity holding copyright, trademark, and the commercial relationship.
- **Development locus**: where the people who actually write the code are, and who employs them.
- **Funding dependency**: what revenue or grant keeps the project alive, and who controls it.

A project can be domiciled in the EU with development concentrated elsewhere, or stewarded from outside Europe with a genuinely distributed contributor base, or European in every respect and dependent on a single customer. These are different exposures and they merit different responses.

Jurisdiction is one input among these five criteria and should not be given lexicographic priority. The channels that reliably determine whether an organisation can change its mind are economic: rent extraction, switching cost, and the concentration of capability in one supplier. Legal-access questions also represent the channel where the mapping from law to operational consequence is least direct.

The third component is the one that moves. Funding determines development locus over time, and development locus determines effective control regardless of where copyright sits. The companion guide's §4.3 works through the clearest case: a rendering engine that began as European free software and whose stewardship migrated with two decades of sustained investment.

3.2 Supplier depth and the European supplier base

A note on how to read C4, because it is easy to misread.

Several of the strongest options in this guide are single-vendor products from European companies. Nextcloud, Univention, Open-Xchange, OpenProject, and grommunio are all in this position: an open licence, a real community, and one company that employs nearly all the core developers and sells the enterprise subscription.

C4 is not an argument against choosing them. It is an argument for the conditions that let a European supplier base grow into something an organisation can rely on: open standards at the interfaces, data in documented formats, and procurement that builds a second source. Interoperability lets European suppliers compete for the work and to be replaced by other European suppliers when they underperform, and a market with that property can carry serious estates. Where an integrator ecosystem already exists around a product, C4 is satisfied even though one company writes most of the code, and this guide records that explicitly where it applies.

The practical test is a question to ask every candidate supplier directly: *if we ended our relationship with you in three years, name the organisations that could take this over, and tell us what we would have to hand them.* A good answer names competitors. Grade the answer.

3.3 A scoring template

For organisations that need an auditable selection record, the five criteria can be scored 1 to 5 with the weights below as a starting point. The weights matter less than declaring them before scoring, without which the exercise ratifies a decision already taken.

Criterion	Suggested weight	Score 1	Score 3	Score 5
C1 Functional coverage	25%	Major gaps in daily use	Gaps in occasional use	Covers used functions
C2 Maturity at scale	20%	No comparable deployment	Comparable, undisclosed	Named public reference at your scale
C3 Manageability	20%	Manual per instance	Scriptable	Declarative, IdP-integrated, upgradeable in place
C4 Supplier depth	20%	One supplier only	Vendor plus partners	Several mutually independent suppliers
C5 Exit path	15%	Undocumented store	Documented export	Standard protocol, several implementations

Two rules keep the exercise from ratifying a decision already taken. Score C2 on evidence you can name, and record "no evidence" as 1 rather than 3. And score C5 against the specific data you would have to move, which §14 enumerates per domain.

Worked example: the mail domain. A template that is never applied teaches nothing, so here it is run against §7.2′s options, twice, under two different requirements. The scores are judgement offered to demonstrate the method and are not findings.

Option	C1	C2	C3	C4	C5	A: Outlook required	B: web-first
Postfix + Dovecot + SOGo	1 / 3	5	4	5	5	**3.80**	**4.30**
Open-Xchange	2 / 5	4	4	3	4	3.30	**4.05**
grommunio	5 / 4	2	4	2	4	**3.45**	3.20
Zimbra	2 / 4	4	3	2	3	2.75	3.25

C1 carries two values because it is the only criterion the requirement changes; the rest are scored once. Three things fall out, and the third is the reason to do this at all.

The ranking inverts. grommunio is the answer under scenario A and last under scenario B. Nothing about the products changed between the columns. This is why §7.3 opens by asking whether you are keeping Outlook: it is not a preference to be traded off against other criteria, it is the question that determines which comparison you are running.

The assembly route scores better than its reputation, on both columns, because C4 and C5 reward exactly what it has. That is the same conclusion §7.3 reaches in prose, arrived at independently, which is mild evidence the weights are not obviously wrong.

Naive weighted scoring gets scenario A wrong. The top scorer under A is the option that cannot meet the requirement, because a 25 percent weight on functional coverage is not enough to overcome a strong all-rounder. The fix is structural: **express a hard requirement as a gate applied before scoring, never as a heavy weight on a criterion.** Under A the gate is native MAPI support, only one option passes it, and the score's job is then to say how comfortable you should be with the forced choice. Here it says: not very, and the thin C2 and C4 are the specific things to manage. That is a more useful output than a ranking.

A weighted sum is a device for making a judgement auditable, and it will faithfully produce a defensible-looking wrong answer whenever a requirement has been encoded as a preference. Check the gates before trusting the arithmetic.

3.4 Re-validating these tables

The five criteria assess a product at a moment. The tables in §§4 to 11 are a snapshot, they are the fastest-ageing content in this guide, and re-running the assessment takes four distinct checks that are easy to collapse into one and should not be.

Liveness. Is it still developed, still released, still supported? The failure mode is quiet: a product carried forward from an older assessment because it was good when that assessment was written. Kopano is the worked example in §2.5, listed as a live alternative in a February 2024 study and end-of-lifed the following year, on 31 March 2025, per the vendor's own end-of-life notice.

Standing. Is it still among the leading options in its domain, or has it been displaced? This is a different question from liveness and it is the one most often skipped, because a maintained project with a working website gives every appearance of being the answer. A product can be alive, competently run, and no longer what a well-informed buyer would choose. Useful signals: whether the domain's newest credible entrant positions itself against this product or ignores it; whether independent integrators still propose it unprompted; whether release cadence is steady or lengthening; whether the contributor base is broadening or narrowing; and whether the integrated suites in §4 selected it, since their assemblers run this evaluation professionally and their choices are a cheap second opinion.

Control. Has the steward, the licence, the funding, or the development locus moved, per §3.1? ownCloud in §6.3 is the worked example. This check has no natural trigger, because nothing about the software changes on the day the company is sold.

Coverage. What is missing from the list entirely? This is the check the other three structurally cannot perform, because they iterate over entries that already exist. A list assembled in one year cannot contain the entrant of the next, and several of the strongest options in this guide (§4′s suites, grommunio in §7, openCloud in §6) are absent from the 2024 study it adapts. The question to ask is **"if I were assembling this list today, with no prior list in front of me, what would be on it"**.

The fourth check is the one that most changes an answer and the one almost never run, because revalidation naturally takes the existing list as its work item. An assessment that only revalidates its own entries converges on an increasingly confident description of an increasingly obsolete market.

Readers should assume this guide needs all four applied to it. §1.4 states the date its assessments were made, and the † markers indicate where the position was already moving at that date.

4. Integrated suites: the option that did not exist in 2024

The most consequential development since the source studies is that assembling this stack component by component is no longer the only route. Integrated, publicly-governed suites now package the components below with the identity, provisioning, and single-sign-on plumbing already done.

4.1 The offerings

openDesk (ZenDiS, Germany), version **1.16.1** as of 30 June 2026, bundles calendar, chat, contacts, documents, email, file storage, identity and access management, a knowledge base, notes, projects, tasks, and video calls. ZenDiS is a publicly-funded limited liability company of the German federal government which took over development in January 2024; the suite was launched by the Federal Ministry of the Interior. It is the reference implementation of the German "Souveräner Arbeitsplatz" programme, and the Swiss backend study already recommended coordinating with it [Standtke and Tiede 2024a]. Note that the public site does not enumerate the upstream project behind each component, so an evaluation should establish that mapping explicitly: it determines what you are actually adopting and what your exit path is per component.†

La Suite numérique (DINUM, France) takes a different shape, offering separately usable services developed in the open by the French state: **Tchap** (messaging), **Visio** (video, hosted in France), **Messagerie**, **Fichiers**, **Docs** (a collaborative editor), **Grist** (collaborative spreadsheets), and **FranceTransfert** (large file transfer). It is built on open foundations including Matrix, LiveKit and Grist Core.

Its adoption figures are the strongest C2 evidence anywhere in this guide: **more than 500,000 agents monthly across 15 ministries**, with Tchap alone reporting 600,000. Both figures are operator-reported, published by the programme itself and not independently audited, which is the normal standing of adoption numbers in this field and is worth knowing when the figure is quoted back at you. For a reader weighing whether any of this works at scale, that is the number to carry into the room.

dPhoenixSuite (Dataport, Germany) is the offering the German feasibility study names as its intended collaboration platform [Kern et al. 2022].

4.2 What a suite changes

A suite converts an integration problem into a deployment problem. The components in §§5 to 9 have to be wired together: a shared identity source, consistent provisioning and deprovisioning, single sign-on across web interfaces, a shared address book, file storage reachable from the editor, meeting links that work from the chat client. Each of those integrations is small; there are perhaps thirty of them; and together they are the largest hidden line in §21′s cost model.

The suite assembler has done that work, keeps it working across component upgrades, and tests the combination. That is the product.

4.3 What a suite costs

Concentration, in three specific forms.

Release cadence. The components' release cadences pass to the assembler. You upgrade when the suite upgrades, and a security fix in one component reaches you through the suite's pipeline.

Component choice. The suite has already chosen your groupware and your chat system. If its groupware choice is web-first and you need native Outlook (§7.1), the suite is not for you until it changes, and you have little influence on that.

A new single supplier. The C4 question applies to the suite as a whole. Ask it of the assembler as well as of each component: who else could operate this suite for us, and what would we hand them.

4.4 How to choose

Situation	Recommendation
Under ~2,500 seats, no platform capability	Evaluate a suite first; assembling this stack yourself is the larger risk
Any size, needs native Outlook	Component route, or wait; check the suite's groupware choice first
Large, existing platform capability, opinionated architecture	Component route; read the suite's choices as a well-researched prior
Public sector in DE or FR	Evaluate the national offering first, on procurement and support grounds as much as technical ones

Verdict: evaluate first. For most organisations below a few thousand seats, an integrated suite is now the sensible default starting point, with component-level assembly reserved for the domains where the suite's choice does not fit.

5. Identity and directory

5.1 Split the problem into three

The Swiss study treats "Verzeichnisdienst" as one thing and recommends one product for it. That framing costs it accuracy. Active Directory occupies three roles that are separately replaceable and increasingly separately supplied:

1. **The directory of record**: the authoritative store of users, groups, and machines.
2. **Authentication and federation**: proving identity to applications, over OIDC, SAML, or Kerberos.
3. **Device join and policy**: machine identity and configuration enforcement.

Most successful designs use different software for each, and the third is largely a client-side question handled in the companion guide. Organisations that look for one product to do all three end up concluding that nothing replaces AD.

5.2 Options

Option	Steward and control	Licence / governance	Supplier depth	Verdict
Samba AD DC	Samba Team; distributed, with major funding from SerNet (DE) and Catalyst (NZ)	GPLv3, community-governed	Broad; many independent integrators	Ready for the directory role
Univention Corporate Server	Univention GmbH (DE, Bremen)	AGPL, single-vendor with partner network	Moderate; vendor plus partners	Ready; the packaged directory and platform
Univention Nubus	Univention GmbH (DE, Bremen)	AGPL, single-vendor	Moderate	Ready for the IAM role; the identity layer inside openDesk
Keycloak	Red Hat / IBM (US), CNCF project; contributor base broad	Apache-2.0, foundation-governed	Broad	Ready for federation
FreeIPA	Red Hat / IBM (US)	GPLv3	Moderate	Ready for Linux-majority estates
Zitadel	Zitadel (CH, St. Gallen)	Apache-2.0, single-vendor	Thin	Emerging†
authentik	authentik Security (US-registered; distributed development)	MIT core, open-core boundary	Thin	Emerging†
OpenLDAP / 389 DS	Community / Red Hat	OpenLDAP licence / GPL	Broad	Ready as a component

Default: Keycloak for federation in every case, and it is the first thing to deploy (§5.5). For the directory itself, Samba where the estate is Windows-centric and you want the protocol unchanged; Univention where you want that packaged and managed; FreeIPA where the estate is Linux-majority. These are three different jobs, per §5.1.

5.3 Samba as a domain controller

Samba is the only open-source implementation of the AD domain-controller protocols themselves, which makes its exit path the best in the table: it speaks the protocol your estate already speaks, so adopting it changes the server without changing the clients. The Swiss study rates it two stars and calls it "eher eine 'Low-level-Komponente'" managed through configuration files [Standtke and Tiede 2024a]. That is accurate about the raw project and

misleading about the option, because products like UCS add the management layer on top of Samba. The German study notes the same relationship from the other side, observing that Samba "ist bereits optimal für die Bereitstellung von SMB-Freigaben für Linux-Clients ausgelegt" [Kern et al. 2022].

What works: domain controller role, Kerberos and NTLM authentication, LDAP directory, DNS integration, replication between Samba DCs, and joining Windows and Linux clients. Existing Windows management tools (the RSAT consoles) attach to a Samba domain and work for everyday user and group administration.

What to check before committing:

- **Forest and domain functional levels.** Samba supports a bounded set. Verify the level your estate requires and whether anything in it depends on a higher one.†
- **Schema extensions.** Applications that extended the AD schema (Exchange being the classic case) need those extensions present or removed. Enumerate them before migrating; this is the most common late discovery.
- **Group Policy.** Samba stores and serves GPOs, and the Windows editors work against it. The client-side application on Linux is a separate question, covered in the companion guide's §9.
- **Replication topology.** Multi-site replication with site links and scheduling needs designing, and cannot be inherited.
- **Trusts.** If you hold trusts with other forests, test them early; this is a common gap.†

5.4 Univention, and the packaged route

Univention Corporate Server is the Swiss study's recommendation and remains the strongest packaged choice for an organisation that wants AD-like central management without assembling it. UCS is a Debian-based server distribution combining an LDAP directory, Samba for AD compatibility, an identity management layer, a web administration interface, and an app catalogue that installs and wires integrated components.

Nubus is a different product, and this guide previously conflated them. Nubus is Univention's identity and access management offering: central IAM, single sign-on, and a self-service portal. It is the identity layer inside openDesk. UCS continues to be developed and sold alongside it, together with UCS@school, and Univention publishes no recommendation of one over the other for new deployments. The distinction maps cleanly onto §5.1: **UCS occupies the directory-of-record and platform role; Nubus occupies the authentication and federation role.** An organisation may want one, the other, or both, and asking which of the two is "the successor" is the wrong question.

The Swiss study reports very high acceptance from both users and administrators, with the observation that end users "werden kaum einen Unterschied bemerken", and notes the REST interface as the integration path for third-party components [Standtke and Tiede 2024a]. It also records the AD Takeover function, which migrates an existing AD domain and spares you a parallel build, and that is the feature that most changes the shape of the project.

The C4 caveat applies and is stated in §3.2: one company, a partner network, and an app catalogue whose integrations are that company's work.

5.5 The federation layer

Keycloak is the federation layer, and most estates need one whether or not they replace the directory. This is the single highest-leverage move in the whole guide, and the one to make first: putting applications behind a standard OIDC or SAML broker decouples them from the directory underneath, which makes every subsequent identity decision reversible.

Sequenced properly it looks like this. Keycloak federates to the existing Active Directory as its user store. Applications are migrated one at a time from direct LDAP binds or Windows-integrated authentication to OIDC against Keycloak. When enough applications

have moved, the directory underneath Keycloak can be changed with the applications none the wiser. The broker makes the directory replaceable.

An organisation that does nothing else in this guide should do this.

5.6 Entra ID

Entra ID has no drop-in replacement, and any claim to the contrary should be treated sceptically. The replacement is a pattern: a directory of record, a federation broker, and SCIM provisioning to the applications that need it.

The hard parts, which should be scoped explicitly, never discovered:

- **Conditional access.** Policy evaluated per sign-in against device compliance, location, risk, and application sensitivity. Keycloak can express a good deal of this through authentication flows and step-up authentication, and the device-compliance signal is the piece that has no direct equivalent because it comes from the device-management platform.
- **Risk-based authentication.** Identity-protection signals derived from a global telemetry estate. This is not reproducible. Substitute stronger deterministic controls (phishing-resistant authentication factors, tighter session lifetimes) in place of pretending to replicate the signal.
- **Cross-tenant collaboration.** Guest access into your services for external identities. Federation gives you this for partners who can federate; for the rest a self-service invited-guest flow is a build.
- **Application provisioning.** SCIM connectors to SaaS applications. Keycloak plus a provisioning layer covers the common targets; the long tail is per-application work.

Plan for authentication factors at the same time. If the estate currently authenticates with anything phishable, the migration is the moment to change it, and doing both at once is cheaper than doing them in sequence.

6. File services, sync, and sharing

6.1 Two jobs

The most common architectural error in this domain is to migrate SMB file shares into a sync-and-share product because the sync-and-share product is the modern-looking one. These are different jobs:

- **Windows File Services** provides network shares with ACLs and protocol-level file locking. Its direct replacement is **Samba**, and the migration is close to invisible to users because the protocol does not change.
- **OneDrive and SharePoint** provide personal sync, external sharing, and web-based document collaboration. Their replacement is a sync-and-share platform.

Doing both at once, through one product, multiplies the disruption and the risk. The German study reaches the same conclusion from the client side, noting that the drive-letter concept has to be abandoned on Linux and that shares are reachable through the file manager or an automounter, while a *separate* server-side sync service is needed because Windows Work Folders has no Linux client [Kern et al. 2022].

6.2 Options

Option	Steward and control	Licence / governance	Supplier depth	Verdict
Samba (file server)	Samba Team; distributed	GPLv3, community	Broad	Ready
Nextcloud	Nextcloud GmbH (DE, Stuttgart)	AGPL, single-vendor open core	Moderate; large partner network	Ready for sync and share

Option	Steward and control	Licence / governance	Supplier depth	Verdict
openCloud	Heinlein Group (DE, Berlin); forked by the former ownCloud Infinite Scale team	Apache-2.0	Thin, growing	Emerging; 5.1.0 rolling, 4.0.3 production
ownCloud	Kiteworks (US) since 2023; OSPO and Apache-2.0 relicensing from May 2026	Apache-2.0 (migrating)	Moderate	Viable with conditions
Seafile	Seafile Ltd (CN) with Seafile GmbH (DE)	GPL / proprietary pro edition	Thin in Europe	Viable with conditions
XWiki / CryptPad	XWiki SAS (FR, Paris)	LGPL / AGPL	Thin	Viable for confidential documents
Ceph	Ceph Foundation (Linux Foundation); IBM-led	LGPL	Broad	Ready as storage substrate
Garage	Deuxfleurs (FR)	AGPL	Thin	Emerging

Default: Samba for the file server and Nextcloud for sync, share, and external collaboration. Two products because they are two jobs (§6.1). openCloud is the second source to pilot in parallel with it.

6.3 What to choose

Samba for the file server, first and cheaply. It is the lowest-disruption change available anywhere in this guide, and it removes a Windows Server dependency and its CALs without touching a single user's workflow. Deployed against existing storage it is a protocol head, and no storage migration, which keeps the two changes separate.

Nextcloud for sync, share, and external collaboration. It is the most widely deployed option, has the deepest partner network in Europe, and integrates the collaborative editing of §8. The Swiss study rates it four stars and records the caveat that acceptance is very high for file sync while other functions "hat jedoch noch an vielen Stellen Verbesserungspotenzial" [Standtke and Tiede 2024a].

Performance at large scale is a design question and not a blocker, and it turns on four things, because deployments go wrong on each of them: the database (which carries the file metadata and becomes the bottleneck before the storage does); a memory cache and file-locking backend; the storage backend, where object storage scales better than a filesystem past a few million files; and background job scheduling, which must be a system timer in place of the default web-triggered mechanism at any serious size.

openCloud matters for the C4 reason more than a functional one. Its emergence gives this domain a second credible European option, which is exactly the market structure §3.2 argues for. It is a fork of ownCloud Infinite Scale by the engineering team that built it, who left in January 2025, and it is stewarded by the Heinlein Group in Berlin. It is young enough that a large estate should pilot it in parallel before committing to it alone.

One disclosure: **the Heinlein Group also stewards OpenTalk** (§9.2). Two of this guide's European recommendations therefore share a single steward. Neither recommendation changes on that account. An organisation adopting both is making a single relationship with one company, which simplifies procurement and also concentrates the relationship. Both are facts, and neither one is hidden by the separate table rows.

The **ownCloud** case is instructive and is frequently told wrong, including in an earlier draft of this guide. The acquisition in 2023 did move control outside Europe. What followed was **not** enclosure: from May 2026 the acquirer established an open-source program office, relicensed over a hundred projects to Apache-2.0, retired the contributor licence agreement in favour of a developer certificate of origin, and published a governance charter. The engineering team left and forked. The lesson is the one in §2.5: steward risk arrives through people at least as often as through licences, and a licence audit would not have predicted this.

6.4 The permission model mismatch

This is the part of a file migration that generates the surprises, and §14.3 covers the mechanics. The design point belongs here: Windows ACLs and POSIX permissions are different models, and Samba bridges them at a cost.

Windows ACLs support per-principal allow and deny entries, fine-grained rights beyond read, write, and execute, and inheritance flags that control propagation to children. POSIX permissions carry three classes and three bits. POSIX ACLs extend this but still lack deny entries and Windows' inheritance semantics.

Samba's answer is to store Windows security descriptors in extended attributes, preserving fidelity for SMB clients, with a mapping to the underlying POSIX permissions for local access. The practical consequences: the filesystem must support extended attributes and be mounted to permit them; backup tooling must preserve them, and much of it does not by default; and any deny-based rule needs review, since deny entries are frequently a workaround for a group structure that should be fixed, and transcribing it carries the workaround forward.

Treat a file migration as an opportunity to rebuild the group model. The alternative is to carry a decade of accreted exceptions into the new system and lose the audit trail that explained them.

7. Mail, calendar, and groupware

This is the hardest domain, and the one where the source studies are most out of date.

7.1 The Outlook problem, and why it changed

The German study states the structural constraint:

> das lizenzrechtliche Verbot innerhalb der USA, das von Microsoft patentierte ActiveSync-Protokoll in Open-Source-Anwendungen zu implementieren. Aus diesem Grund gibt es keine Open-Source-E-Mail-/Groupware-Client-Applikation, die man gegen einen MS-Exchange-Dienst verwenden kann. [Kern et al. 2022]

The consequence drives their whole sequencing: replacing Outlook forces you to replace Exchange, so the two decisions are welded together and the interim answer is Outlook on the Web. The Swiss study reaches the same place from the other direction, recommending Open-Xchange and accepting a move to web clients.

That welding has loosened. **grommunio** implements the protocols native Outlook speaks, MAPI/HTTP and RPC-over-HTTP, alongside EWS, Exchange ActiveSync, CalDAV, CardDAV, and IMAP. The vendor's description of the client side is the operative claim: Outlook for Windows connects natively, "nothing needs to be installed on the client, nor are there connectors in between". The server can therefore be replaced while the client stays. Its lineage runs through Zarafa and Kopano, which is why the Swiss study's "Kopano" row is better read today as pointing at grommunio. Kopano itself should be treated as closed: the collaboration suite reached end of life on 31 March 2025 and its successor is a proprietary cloud service, which is the enclosure case in §2.5.

If that holds, the sequencing options invert. Instead of one large simultaneous change of server and client, an organisation can replace Exchange first while users keep Outlook, and change the client later as a separate, smaller project. For a large estate this is a materially

different risk profile, and it is the single most important thing to verify before planning this domain.

7.2 Options

Option	Steward and control	Licence / governance	Supplier depth	Verdict
grommunio	grommunio GmbH (AT, Vienna)	Open source (GitHub, Codeberg) plus commercial subscription	Moderate; 200+ certified partners claimed	Ready; the only native-Outlook path
Open-Xchange	Open-Xchange AG (DE, Cologne)	AGPL	Moderate; large telco and ISP base	Ready for a web-first estate
Carbonio	Zextras (IT, Trento)	Community edition open; open core	Thin	Emerging†
Zimbra	Zimbra Inc. (US)	Source available; official binaries licence-gated from 10.1	Moderate; open path runs through third-party rebuilders	Viable with conditions
SOGo	Alinto (FR) since the Inverse acquisition	GPL	Thin	Viable as a groupware layer†
EGroupware	EGroupware GmbH (DE)	GPL / open core	Thin	Viable for smaller estates
Stalwart	Stalwart Labs	AGPL	Thin	Emerging as an MTA/IMAP/JMAP stack
Postfix + Dovecot (+ SOGo)	Community	IBM Public Licence / MIT / GPL	Very broad	Ready as the assembly route

Default: settled by a gate applied before any ranking (§3.3). Keeping native Outlook makes grommunio the only candidate. Otherwise Open-Xchange for the packaged route, or Postfix with Dovecot and a groupware layer where supplier depth and exit path matter more than integrated calendaring.

7.3 What to choose

The decision turns on one question asked first: **are you keeping Outlook?**

If yes, **grommunio** is effectively the only path.

Its core is gromox, currently at 3.8, and the company claims deployments from six users to over 100,000 and more than 200 certified partners. A partner-programme count is a weaker signal than independent capability, so read it as evidence that a market exists rather than as proof of depth, and apply §19.4′s test by asking which of those partners could take the work over.

Make **scaling to your mailbox count an explicit, measured pilot objective** with a defined pass mark. The vendor claim of 100,000+ is a claim, and verifying it against your own data is ordinary diligence. Track one roadmap item if you are also modernising authentication: OAuth2 for Outlook, IMAP and POP3 is listed as planned rather than shipped, which interacts with §17.4.

If no, **Open-Xchange** is the mature choice, and both the Swiss study and the Schleswig-Holstein strategy select it. The Swiss study's assessment is unusually strong: "höchstwahrscheinlich die beste frei verfügbare Alternative zu Exchange", ahead of the field on features, ergonomics, performance, and developer responsiveness [Standtke and Tiede

2024a]. Its reference deployments in the study are substantial C2 evidence: a large German city education authority, a European research organisation with published migration notes, and several municipal administrations. Schleswig-Holstein pairs it with Thunderbird as a supplementary client and, notably, sells the change to staff on "considerably larger mailboxes for all users" instead of on sovereignty [Staatskanzlei Schleswig-Holstein 2024]. That tactic is discussed in §20.5; copy it.

The loose version of **Zimbra**'s position is unfair to it. The source remains available and maintained in public. The change is that **official binaries require a licence key from version 10.1**, so the freely-deployable path now runs through third-party rebuilders publishing community builds with security patches on their own cadence. That is a C4 finding rather than a licensing scandal: the open route exists and depends on parties other than the vendor, which is a different risk from the one Zimbra is usually accused of, and arguably a worse one to carry unknowingly. The Swiss study rates it four stars and records a satisfied reference at the Swiss Federal Court, roughly CHF 25 per user per year, administered by one part-time person [Standtke and Tiede 2024a]. **Carbonio** is the European entry in the same lineage and deserves evaluation on that basis.†

The **assembly route** (Postfix, Dovecot, and a groupware layer) deserves more respect than the Swiss study gives it. Its C4 and C5 scores are the best available: these are the most widely operated mail components in existence, every competent mail supplier knows them, and the data sits in Maildir and standard protocols. Its cost is that groupware features (shared calendars, delegation, resource booking, free/busy) come from the layer above and are less integrated. For an organisation whose calendar requirements are modest, this route is undervalued.

7.4 The part that actually hurts

Neither the products nor the mailbox migration is the hard part. **Free/busy federation between the incumbent and the replacement during coexistence** is the hard part, and hybrid mail projects fail on it. The Swiss study identifies this as the central challenge and then declines to solve it, noting only that the study "zeigt erste Möglichkeiten theoretisch auf" for interoperability topics including free/busy, calendar sharing, mail routing, and cross-world meeting invitations [Standtke and Tiede 2024a].

Plan this before choosing a product. §15.3 sets out the workable patterns.

7.5 The features that get forgotten

Enumerate these against your estate before selecting, because each is a potential blocker and none appears in a feature comparison:

- **Shared and functional mailboxes**, and the delegation model over them. Who can send-as, send-on-behalf, and see what.
- **Resource booking**: rooms and equipment, with capacity, location, approval workflows, and recurring bookings.
- **Free/busy visibility rules**, internal and external, and what detail is exposed to whom.
- **Server-side rules and out-of-office**, including whether rules carry over in a migration at all. Usually they do not.
- **Retention, journaling, and legal hold**, which are compliance obligations before they are features, and which frequently sit in a third-party archive product that also needs replacing.
- **Distribution lists and dynamic groups**, and whether membership is directory-derived.
- **Mobile access** and the device-management interaction.
- **Signatures**, centrally applied, which is invariably someone's project.
- **Large mailbox and attachment limits**, the usual source of the promised user benefit.

7.6 Mobile access

The forgotten domain, and one users notice within a day. Three separate questions hide behind "does it work on phones".

Mobile access to the services. Mail, calendar and contacts reach a phone over the incumbent's mobile protocol, or over the open standards, or through a vendor's own application. The open-standard path works, and most of the options in §7.2 offer it; the vendor-application path varies in polish, and the gap with the incumbent sits there. Nextcloud, Element, Open-Xchange and grommunio all ship mobile clients, and their quality is the thing to test in the pilot on real devices, which reading cannot substitute for.†

Mobile device management. This is a separate estate with its own procurement, and the migration should integrate with it rather than absorb it. The Swiss study treats the existing management platform as a fixed integration constraint and not as a replacement target, which is the right call [Standtke and Tiede 2024a]. Open-source management of current mobile operating systems is thin, because the platforms expose management through vendor-controlled channels, and an organisation that tries to replace its mobile management as part of this programme is adding a hard problem to a full one.

Enrolment and conditional access. If device compliance currently gates access to mail, that signal comes from the management platform and has to be re-plumbed to the new identity layer. §5.6 flags this as one of the hard parts of leaving the incumbent identity service, and it bites first on mobile.

Recommendation: treat mobile device management as out of scope and integrate with it; treat mobile *access* as a pilot acceptance criterion with named test cases per §23; and get real devices into the pilot from week one, because mobile complaints arrive faster and louder than any other kind.

8. Collaborative editing

8.1 A correction to the sources

The Swiss study lists "Nextcloud Office" and "Collabora" as separate alternatives with separate ratings. They are the same engine: Nextcloud Office is Collabora Online packaged and integrated by Nextcloud. Treating them as independent options overstates the number of choices in this domain, which is a small one.

There are three real engines: **LibreOffice Technology** (which underlies both Collabora Online and LibreOffice itself), **ONLYOFFICE Docs**, and the newer **CRDT-based editors** built for the web, with no desktop-suite ancestry.

8.2 Options

Option	Steward and control	Licence / governance	Supplier depth	Verdict
Collabora Online	Collabora Productivity (UK, Cambridge); the largest LibreOffice code contributor	MPL-2.0	Moderate	Ready
LibreOffice / TDF	The Document Foundation (DE, Berlin)	MPL-2.0, foundation-governed	Broad	Ready (desktop; see companion guide)
ONLYOFFICE Docs	Ascensio System SIA (Riga, LV), held via Ascensio System Ltd (UK) by ONLYOFFICE	AGPLv3 plus disputed §7 additional terms	Moderate	Viable with conditions; see §8.3

Option	Steward and control	Licence / governance	Supplier depth	Verdict
	Capital Group Pte Ltd (Singapore)			
Euro-Office	European vendors, following a 2026 fork of ONLYOFFICE	AGPLv3, trademarks removed	Thin, new	Emerging†
Docs (La Suite / ZenDiS)	DINUM (FR) and ZenDiS (DE), jointly	MIT/AGPL†	Thin	Emerging†
CryptPad	XWiki SAS (FR)	AGPL	Thin	Viable for confidential work
Etherpad / HedgeDoc	Community	Apache-2.0 / AGPL	Broad	Ready for lightweight cases

Default: Collabora Online, paired with desktop LibreOffice for offline work. Choose ONLYOFFICE instead where OOXML fidelity is the binding constraint and §8.3′s control fact pattern is acceptable to you.

8.3 What to choose

Collabora Online for an organisation whose documents are Office documents, which is nearly all of them. Its fidelity comes from sharing an engine with LibreOffice, so the browser and the desktop render the same file the same way, which matters more than any individual feature. Pair it with desktop LibreOffice for offline work, as the Swiss study recommends [Standtke and Tiede 2024a].

ONLYOFFICE has the best OOXML fidelity of the three and runs much of its rendering in the browser, which the Swiss study notes as a performance advantage [Standtke and Tiede 2024a]. The same study records two limitations: OpenDocument files could at that time be read but not written, and a threat entry concerning the enforceability of digital-sovereignty commitments given the company's domicile.

The control question deserves the full chain, because an earlier draft of this guide recorded only the bottom of it, which is exactly the error §3.1 warns against. The intellectual property sits with **Ascensio System SIA in Riga, Latvia**, wholly owned by **Ascensio System Limited in the United Kingdom**, wholly owned by **ONLYOFFICE Capital Group Pte. Ltd. in Singapore**, a structure established in a 2023 reorganisation. Development is distributed across offices including Riga, Singapore, London, Dallas, Belgrade, Yerevan, Tashkent and Shanghai, with roughly 133 staff. The chief executive holds Russian and Turkish citizenship and has been resident in Istanbul since 2023.

"EU-domiciled" is therefore true of one entity and misleading about the whole. Whether that matters is an organisation's own judgement against its own threat model, and this guide does not make it. The guide insists only that the judgement be made against the chain instead of against its last link.

The licensing dispute matters more to most readers than the jurisdiction does, because it bears directly on C5. The sequence, as far as it can be established from public sources:

- Around **May 2021**, terms were added under section 7 of the AGPLv3 in the project's licence file, without a change to the public licence label. One requires that "you must retain the original Product logo when distributing the program", stated as a section 7(b) obligation. Another, under section 7(e), declines to grant any rights in the trademarks.
- On **27 March 2026**, Nextcloud and IONOS announced **Euro-Office**, a fork of ONLYOFFICE with the Ascensio trademarks removed. Days later Ascensio suspended its partnership with Nextcloud and posted a legal notice claiming the fork breached its

licence terms, its chief executive stating that Euro-Office "removed all references to our brand/attribute as required by our license".

- On **15 April 2026** the Free Software Foundation published an analysis holding that the logo obligation "is not included in Sec. 7(b) of the (A)GPLv3 … and is therefore considered a further restriction", which recipients are entitled to remove [Free Software Foundation 2026].
- On **16 April 2026** the Software Freedom Conservancy published a longer analysis describing the arrangement as badgeware, noting that section 7(b) permits only "specified reasonable legal notices or author attributions" and that "logos, advertising, and similar are not on the list" [Software Freedom Conservancy 2026].

The structural objection is that the two terms combine into something neither achieves alone: a fork must retain a mark it has no licence to display, which makes forking impossible to do compliantly, and section 10 of the licence forbids imposing further restrictions on the rights granted.

Ascensio's position is that each term is independently permitted by section 7, that they form an indivisible whole, and that the Foundation's 2026 analysis is an opinion which cannot amend a licence whose binding text dates from 2007. A licence steward's later commentary is indeed not an amendment. It is also true that the steward's reading of its own instrument is the best available evidence of what its terms were drafted to mean, and the Conservancy's account of the drafting history is that section 10 exists to prevent exactly this class of entrapment. The dispute is unresolved, and the Conservancy characterises Ascensio's reply to the criticism as "unfair, inaccurate, aggressive, and community-unfriendly".

What this means in practice depends entirely on what you intend to do. An organisation that deploys ONLYOFFICE unmodified and does not redistribute it is largely unaffected; the software works and the licence permits the use. An organisation that intends to modify, rebrand, embed in its own distribution, or offer it as a differently-branded service is in the contested zone, against a publisher that has demonstrated willingness to issue legal notices. Since this guide treats forkability as the backstop that makes every other guarantee credible (§3, C5), that is a material markdown on C5, which carries more weight than any other criterion for long-run freedom of action.

Scored on this guide's own criteria, the result is not close. C1 remains strong: the OOXML fidelity is the best of the three engines. It lands badly on C4 and C5. On C4, the question in §3 is whether anyone else could take it on, and a fork that must either retain a mark it cannot license or invite a legal notice is not a credible second source. On C5, forkability is the backstop that makes every other guarantee in this guide credible, and it is the specific thing in contention. An organisation selecting ONLYOFFICE is therefore choosing on C1 against C4 and C5, which is a legitimate trade to make knowingly and a poor one to make by accident.

Three consequences. **The default does not change**, and is reinforced: Collabora Online is MPL-2.0 with no comparable dispute, and it was already the recommendation. Any organisation relying on the **Nextcloud plus ONLYOFFICE integration** should note the partnership suspension and plan on that combination no longer being a supported path.† And **Euro-Office** is listed in §8.2 for coverage rather than recommendation: it is weeks old at the time of writing, its backers are substantial, and whether it acquires an independent community is exactly the C4 question asked of a fork.†

A note on how far this generalises, because it is easy to over-read. The conduct at issue concerns trademark and attribution terms and the manner of their enforcement. It does not bear on the software's quality, and it does not automatically transfer to every other single-vendor product in this guide. It does justify applying C4′s open-development test (§3) to this product specifically and carefully, since a publisher willing to litigate the boundaries of

its licence is one whose development model, contribution policy, and integration interfaces should be established before you depend on them.

Watch **Docs**, the jointly-developed French and German editor, for the governance reason more than the functional one.† Two states sharing one codebase is the arrangement most likely to produce a component that neither can capture, which is the structural answer to §3.2′s problem.

8.4 Sizing and the concurrency question

Collaborative editing is the component whose resource profile surprises people, because cost scales with *concurrently open documents* and not with users. Each open document holds a rendering process with the document in memory; a spreadsheet with heavy computation holds considerably more than a text document.

The planning approach that works: estimate peak concurrent documents as a fraction of active users (a low single-digit percentage is a common starting point for general office populations, higher where the organisation works document-centrically), size memory per document from a measured sample of your own worst documents, and load test before committing. The failure mode is a shared pool exhausted by a handful of large spreadsheets, which degrades editing for everyone.

Both Collabora and ONLYOFFICE scale horizontally behind a load balancer with document affinity, so the architecture is straightforward once the per-document cost is measured.

9. Real-time communication

The Swiss study's verdict was blunt: "Im Bereich Enterprise Communication gibt es zurzeit kein Produkt, das Skype for Business 'out of the box' ablösen kann" [Standtke and Tiede 2024a]. That remains true of the whole bundle. It is no longer true of the parts.

9.1 Decompose the bundle

Teams and Skype for Business bundle four separable things: persistent chat, video meetings, enterprise telephony, and presence and federation. They have different maturity levels in open source and different replacement costs, and the bundle makes the problem look intractable.

9.2 Options

Option	Steward and control	Licence / governance	Supplier depth	Verdict
Matrix (protocol)	Matrix.org Foundation (UK)	Apache-2.0, open standard	Multiple server and client implementations	Ready as the protocol
Element	Element / New Vector (UK), with EU entities	AGPL	Moderate; plus Famedly, Nordeck (DE)	Ready for chat
Synapse / conduwuit / Dendrite	Element and community	AGPL / Apache-2.0	Moderate	Ready; multiple implementations is the point
OpenTalk	Heinlein Group (DE, Berlin)	EUPL-1.2	Thin	Ready; v25.2.0, stable; built for this buyer
Jitsi Meet	8x8 Inc. (US) since 2018	Apache-2.0	Broad	Ready; steward outside Europe
BigBlueButton	BigBlueButton Inc. (CA)	LGPL	Moderate	Ready for teaching contexts
Nextcloud Talk	Nextcloud GmbH (DE)	AGPL	Moderate	Viable for small meetings

Option	Steward and control	Licence / governance	Supplier depth	Verdict
Asterisk / FreeSWITCH / Kamailio	Sangoma (US) / community / community	GPL / MPL / GPL	Broad	Ready as engines only

Default: Matrix with Element for chat, chosen for federation and the plurality of server implementations (§9.3). For video, OpenTalk where European stewardship is weighted, Jitsi where deployment simplicity is, BigBlueButton for teaching. For telephony, no default: it is an integration project (§9.5).

9.3 Chat

Matrix, and the reason is federation. Matrix is the only option here that is a standard with several independent server implementations, which makes it the strongest C5 entry in the whole guide: the exit path from one Matrix server is another Matrix server. Its public-sector deployment record is substantial, and the Swiss study already listed the German armed forces messenger, the German health telematics messenger, and the French government's Tchap [Standtke and Tiede 2024a]. Element is the mature implementation; Famedly and Nordeck are the European alternatives C4 asks about.

Four operational notes matter at scale. The reference server's resource use is dominated by room state resolution, so a small number of very large rooms with many federated participants costs far more than many small rooms. Media storage grows without bound unless retention is configured, and it is frequently the largest single store in the deployment. Federation is a policy decision as much as a technical one: an open federation exposes your users to the wider network, and most organisations want an allowlist. Bridges to other networks exist and are the most common source of operational trouble; treat each bridge as a service with its own lifecycle.

9.4 Video

Pick on jurisdiction and scale, because the technology is settled. Jitsi is the most widely deployed and easiest to self-host. 8x8 acquired it from Atlassian in 2018, it remains Apache-2.0, the team stayed intact, and no narrowing of openness is evident since; the American steward is a fact to weigh and not a defect, and the guide's earlier phrasing implied a recent change that did not happen. **OpenTalk** is the European-stewarded option built specifically for this buyer, at v25.2.0 under EUPL-1.2, stewarded by the Heinlein Group, with the Thuringian state administration among its named public references. BigBlueButton is stronger than either for structured teaching, with breakout rooms, polling, and a whiteboard designed for classroom use.

One disclosure repeated from §6.3 because it spans two sections: the Heinlein Group stewards **both OpenTalk and openCloud**. An organisation choosing both is concentrating supplier risk that the separate tables do not show.

The Swiss study's field evidence for Jitsi is good: the Swiss Federal Court runs it successfully with a custom extension for dial-in by telephone number, with high acceptance from administrators and users [Standtke and Tiede 2024a].

Sizing is bandwidth-bound rather than CPU-bound in the selective-forwarding architecture all of these use. The media router forwards streams without mixing them, so the constraint is network egress at the server and downstream capacity at each participant. Recording changes this completely, because it reintroduces mixing and transcoding, and a recording-heavy deployment needs separate capacity planning. Plan for a media server per site or region if participants are geographically spread, since a single central media path adds latency to every stream.

9.5 Telephony

This is an integration project, and calling it anything else causes failures. Asterisk, FreeSWITCH, and Kamailio are engines. Replacing enterprise telephony means rebuilding attendant consoles, call queues, hunt groups, executive-assistant call handling, emergency location, voicemail, and PSTN breakout, against hardware and carrier arrangements that are usually contractually fixed. The Swiss study rates Asterisk one star for this reason and recommends waiting [Standtke and Tiede 2024a].

Schleswig-Holstein's answer is instructive: they are building **OSKAR**, an open-source telephony architecture, as a named multi-year programme with a softphone as the standard endpoint, replacing an incumbent proprietary telephony service progressively from the end of 2024 [Staatskanzlei Schleswig-Holstein 2024]. That is the correct shape: a programme with its own budget, timeline, and supplier, sequenced after everything else.

The dependencies that fix the timeline are usually contractual rather than technical: SIP trunk contracts and their notice periods; number ranges and porting; emergency-call location obligations, which are regulated and non-negotiable; handset estate and its replacement cycle; and any integration into contact-centre or business systems.

9.6 Presence and federation

The forgotten fourth component. Presence within one system is straightforward. Presence *across* the boundary during coexistence generally is not. Users will have two presence states during the transition and will find it annoying. §15 treats this as a coexistence cost to be declared instead of solved.

10. AI assistance and the Copilot question

This section carries more † markers than any other, because it is the fastest-moving area in the guide. Omitting it would be worse: it is the first question most executives ask, and it is also the mechanism behind the price series §2.2 uses as its central evidence.

10.1 Why this is an economics question before it is a technology question

§2.1 sets out the bundling mechanism: a package priced below the sum of its parts increases dependence, "because 'the little bit left' is included in the package 'free of charge' and there is no need to look for an alternative to it" [Staatskanzlei Schleswig-Holstein 2024]. Generative AI assistance is the current instance of that mechanism, and the largest one to date. It arrives as a per-seat addition or as a reason for a tier upgrade, it is priced against the productivity it claims rather than against its cost, and it attaches to the identity and content layers, which are the hardest parts of the estate to move.

An organisation that adds a tenant-wide AI assistant to an incumbent suite has, in the terms of §2.6, reduced its exit threat. The assistant is grounded on the content and permission graph of the incumbent platform, so its value grows with the amount of content held there and disappears on the day that content moves. That is a switching cost by construction and should be recorded as one in any business case.

This does not make adopting it wrong. It makes it a decision with a term that belongs on the ledger.

10.2 What the incumbent assistant actually does

Decomposing it makes the replacement question tractable, because the parts have very different open-source positions:

1. **Drafting and rewriting** in documents and mail: generate, summarise, change tone, translate.
2. **Meeting capture**: transcription, summary, action extraction.
3. **Retrieval over the tenant**: answering questions grounded in the organisation's own documents, mail, and chat, respecting each user's permissions.
4. **Task automation**: chaining actions across the suite.

Items 1 and 2 are largely model capability plus a user interface. Item 3 is the hard one, and item 4 depends on it.

10.3 The open-source position, by layer

Layer	Options	Position
Models (open weights)	Mistral (FR), and the open-weight families generally; European public efforts including OpenEuroLLM†	Ready for drafting, summarisation, translation
Serving	vLLM, llama.cpp, Ollama, and managed inference from European providers	Ready
Transcription	Whisper-family models and their derivatives	Ready
Retrieval and grounding	Vector stores plus retrieval pipelines, assembled	Emerging; the permission model is the hard part
Integration into the suite	Nextcloud's assistant features, Open-Xchange AI features, OpenTalk and Element integrations†	Emerging†, moving quickly

Drafting, rewriting, translation, and summarisation are the substitutable parts, and open-weight models running on modest infrastructure cover a large share of everyday use. For an organisation whose actual usage is "help me write this paragraph" and "summarise this thread", the gap is much smaller than the marketing implies.

Meeting transcription and summarisation are likewise well served, and self-hosting carries a specific advantage: meeting audio is among the most sensitive content an organisation produces, and processing it on infrastructure you control is an easier conversation with a works council or a regulator than processing it anywhere else.

Tenant-wide grounded retrieval is the incumbent's one real advantage. Answering "what did we decide about the Hamburg contract" requires an index over mail, documents, and chat that enforces each user's permissions at query time, stays current, and spans systems. Building that across an assembled open-source stack is an engineering programme, and any supplier who says otherwise should be asked to demonstrate it against your permission model with your data. That is the gap; test it before arguing about it.

10.4 What to do

Do not let AI be the reason not to migrate, and do not pretend the gap is zero. Both positions are common and both are wrong. The defensible sequence:

1. **Measure actual usage** before assuming the requirement. Assistant licences are frequently bought broadly and used narrowly, and the usage telemetry is available to you. This is §18.2′s function-level analysis applied to a new category, and it often changes the requirement substantially.
2. **Budget inference explicitly.** It is a running cost with a different shape from a subscription: capital or hourly compute, scaling with use. §21.4 lists it among the costs the model does not carry.
3. **Decide build, buy, or European API.** Self-hosted open-weight inference maximises control and demands capability per §18.4. A European inference provider is the middle path, and most organisations will land there. Both keep the model layer replaceable, which is the §3 criterion that matters.
4. **Keep the assistant separable from the content platform.** This is the whole lesson of §10.1. An assistant that reads your documents through a documented interface can be changed; one that is welded to the platform cannot, and it welds the platform in place too.

5. **Pilot grounded retrieval** against real content and real permissions, and accept the answer.

11. Print

Both Swiss studies and the German study reach the same conclusion, and nothing since has changed it. **CUPS with IPP** is a mature and complete substrate for printing itself, and the German study recommends moving the whole print infrastructure to it, with a CUPS-plus-Samba arrangement for mixed estates described as "die technisch leichter wartbare Lösung": Linux clients print directly to the central CUPS server, and Windows clients send jobs to a Samba service on the same host which forwards them [Kern et al. 2022].

The gap is the layer above: secure pull-printing with badge release, per-user accounting, scan-to-workflow, and the multifunction-device panel integration that goes with them. These depend on device firmware and on vendor-specific middleware. The Swiss backend study names printing as one of its two exceptions where no equivalent exists, and the frontend study puts the secure-print client out of scope on hardware-dependency grounds [Standtke and Tiede 2024a, 2024b]. The German study observes that multifunction devices generally lack Linux support for anything beyond printing, and that full-feature panel interfaces come from the manufacturer and "kommen daher im Linux-Kontext eher selten vor" [Kern et al. 2022].

One hardening note from the same source, easily missed: the local CUPS service on each client "ist je nach Distribution in seiner Standardkonfiguration zu offen konfiguriert" and needs closing down.

Verdict: not yet a replacement at the secure-release level. Keep the print stack longest, move to CUPS where the secure-release requirement is absent, and treat printing as a constraint on sequencing and not as a target. Where secure release is mandatory, the realistic options are to retain the existing middleware against a CUPS backend if it supports one, or to accept that this workflow remains on the incumbent stack.

12. Where to run it: the hosting decision

A guide about sovereignty that discusses only software would miss where most of the sovereignty is actually won or lost. Open-source software running on infrastructure controlled by the entity you were trying to become independent of leaves the jurisdictional exposure largely intact and changes mainly the licence bill.

This decision also determines cost, staffing, and the §18.4 capability question more than any product choice in §§5 to 11, and it should be taken ahead of the reference architecture in §13.

12.1 The four options

Option	What you control	Staffing demand	Exit cost from *this* choice
Own datacentre	Everything, including physical access	Highest: facilities, hardware, platform, application	Low for software; high for the facility itself
European cloud or hosting provider	Software, data, configuration; not the metal	High on platform and application	Low, if the workloads are portable and the data is exportable
Managed private cloud from a European integrator	Data and configuration; the supplier operates	Lowest: you specify and govern	Moderate; depends entirely on §19′s contract terms

Option	What you control	Staffing demand	Exit cost from *this* choice
Hyperscaler	Software and data; the platform is theirs	Moderate	Highest, and it accrues through managed services and egress

12.2 The trap

Running the open-source stack on a hyperscaler is a coherent choice for some organisations and a trap for others, and the difference lies in what the migration was for.

If the objective is to escape per-seat licence escalation and regain the ability to change application supplier, hosting on a hyperscaler achieves it. The applications are portable, the data is in documented formats, and the exit threat in §2.6 is restored at the application layer.

If the objective includes jurisdictional exposure, it does not. The infrastructure operator retains the legal relationship, the operational access, and the commercial leverage, and the organisation has moved its dependency down a layer while reporting that it has removed it. Worse, the dependency it moves to is the one with the highest exit cost, because managed services and data gravity accumulate unseen.

The failure mode to name explicitly, because it is common: a programme justified on sovereignty grounds, executed as a software migration, and landed on infrastructure that leaves the original exposure unchanged. Everyone involved can defend their own decision, and the sum of those decisions defeats the programme's purpose.

12.3 Two corrections, in opposite directions

Two corrections in opposite directions, both following §3.1′s insistence on decomposing control.

Do not overclaim for on-premise. An own datacentre running hardware designed and manufactured elsewhere, on firmware you cannot audit, with a support contract from a foreign vendor, is not free of dependency. It has a different dependency profile with different failure modes, and some of them (facilities, power, physical security, staff retention) are worse. The Schleswig-Holstein strategy is careful about this, noting that dependency is never eliminated completely because "the very size of the IT infrastructure always leads to a certain degree of technological dependence" [Staatskanzlei Schleswig-Holstein 2024].

Do not underclaim for European providers. The channel that most reliably determines whether you can change your mind is economic: switching cost and supplier concentration, per §3.1. A European provider running portable workloads on documented interfaces scores well on that channel whatever its size, and the market of such providers is deep enough that no single one is indispensable.

12.4 How to choose

The question that resolves it is **what specifically are you trying to be independent of, and by when**. Three common answers and where they lead:

- **Price escalation and supplier lock-in.** Any of the four works. Choose on cost and capability, and put §19′s exit terms in the contract.
- **Jurisdictional exposure for specific data.** The first three work; the fourth does not, for that data. Note that this is usually a subset of the estate, which permits a split: sensitive workloads on controlled infrastructure, the rest wherever it is cheapest.
- **Continuity under disruption.** This points at §16.5′s sovereign reserve as a distinct, independently-hosted capability, and the hosting choice for the reserve can differ from the choice for the main estate.

Organisations frequently treat hosting as one decision when it is several; hence the second answer's split. Classify the data first, then host each class where its requirements point. A guide that insisted on one answer for the whole estate would be imposing a cost most organisations have no reason to pay.

12.5 What it does to the rest of the guide

The hosting choice propagates. **Managed private cloud** removes most of §18′s operations burden and makes §19′s contract the primary control, which suits organisations that fail §18.4′s capability test. **European cloud or own datacentre** means §13 to §18 are yours to execute and staff. **Hyperscaler** makes §13′s reference architectures easier and §12.2′s caveat decisive.

For most organisations below the medium profile in §1.2, managed private cloud from a European supplier is the pragmatic answer, and it preserves the benefits of §2.6 provided the contract secures exit. Saying so is more useful than implying that everyone should build a platform team.

13. Reference architectures and sizing

The source studies stop at product selection. This section covers what the deployment looks like, because the architecture determines the operating cost that §21 tries to estimate.

13.1 The layers

All designs in this section share the same five layers, and naming them separately keeps them replaceable:

1. **Identity**: directory of record, federation broker, provisioning.
2. **Data**: databases, object or file storage, caches, search indices.
3. **Application**: the components of §§5 to 9.
4. **Edge**: reverse proxy, TLS termination, load balancing, and the media path for real-time services.
5. **Operations**: monitoring, logging, backup, configuration management, secret management.

The German study's rule about interchangeability applies to all of them: design the interfaces "so generisch wie möglich" from the beginning and avoid hard dependencies, particularly between the workplace and the infrastructure [Kern et al. 2022]. In practice this means components talk to the identity layer through OIDC and to storage through documented interfaces, and nothing reaches into another component's database.

13.2 Small: around 250 seats

A consolidated design. Two or three physical hosts or their virtual equivalents, with a hypervisor or container platform beneath.

- Identity, mail, files, and collaboration each in their own virtual machine or container, on shared hosts.
- One database instance serving several applications, with separate databases and credentials per application.
- A single reverse proxy handling TLS for all web components.
- Storage on the hypervisor's storage, backed up off-box.
- Real-time communication as a managed service from a European supplier unless there is a specific reason to self-host, because the media path is the least rewarding thing to operate at this size.

Availability comes from fast restore, and not from redundancy. A tested restore path with a documented recovery time is worth more at this size than a half-understood cluster, and in clustering a database, small deployments most often introduce the outage they were trying to prevent.

At this size the recommendation is §4: evaluate a suite, or buy this stack operated by a supplier. An organisation of 250 seats that self-assembles seven components is taking on a platform capability it will struggle to staff, and §18.4 explains why.

13.3 Medium: around 2,500 seats

Separation of concerns, with redundancy where an outage is unacceptable.

- Identity in a redundant pair, in different failure domains. This is the first thing to make redundant, because everything else authenticates against it.
- Database on a replicated pair with automated failover, and here pay for expertise, since database failover is the component that most rewards it.
- Mail split into edge (inbound and outbound MTAs, filtering) and store, which lets the edge scale and be patched independently.
- File storage on a dedicated storage system or an object store, with the sync-and-share application stateless in front of it.
- Collaborative editing horizontally scaled behind the load balancer with document affinity, per §8.4.
- Chat with the homeserver, its database, and media storage separated, since media dominates growth.
- Video with a media server per site if sites are geographically distributed.
- A monitoring and logging stack that exists before the first service goes live, per §18.1.

13.4 Large: around 25,000 seats

Platform-based, with the applications as tenants of an internal platform.

- A container orchestration platform, which Nubus, openDesk, and the container-native component lines all target.†
- Identity as a highly available service with regional presence, and provisioning automated end to end.
- Storage as a service layer (object storage with a documented API) in place of per-application storage decisions.
- Per-component horizontal scaling with capacity managed against measured demand.
- Multi-site media paths for real-time services.
- A full operations layer: observability, policy enforcement, secret management, automated certificate lifecycle, and a change process.

The German study's warning applies with force at this size: provisioning the lifecycle-management infrastructure and its satellite systems is "sehr kostenintensiv", so the product choice has to be made carefully with long-term extensibility in view, and the disadvantage of a generic design is a longer lead time before rollout can begin [Kern et al. 2022]. The regional structure of the estate drives this: they cite "die sehr starke Untergliederung des schleswig-holsteinischen Landesnetzes in viele unterschiedliche Standorte" as the reason satellites are needed at all.

13.5 Planning heuristics

First-pass figures for a capacity conversation. Every one of them should be replaced by a measurement against your own data before anything is ordered.

Quantity	Planning heuristic	What actually determines it
Mailbox storage	Current mean mailbox size, times seats, times growth over the planning horizon, times a factor for indexes and overhead	Whether attachments are deduplicated; retention policy; whether the archive migrates too
Mail throughput	Peak messages per second, measured from current logs, with headroom of several times mean	Bulk senders, notification systems, and the morning peak, which is sharper than most people expect
File storage	Current used capacity plus growth, plus versions, plus trash retention	Version retention policy, which is the single largest multiplier and is usually set by default
File metadata	File count drives the database, independent of size	Sync clients re-scanning; deep directory trees

Quantity	Planning heuristic	What actually determines it
Collaborative editing	Peak concurrent documents, memory per document from a measured sample	Large spreadsheets; §8.4
Chat	Active users, room count and size, media retention	Large federated rooms; media dominates storage
Video	Concurrent streams times bitrate, at the egress	Recording, which changes the profile entirely
Identity	Authentications per second at peak, session lifetime	Session lifetime is the lever; short sessions multiply load

Two systematic errors recur. **The database is under-provisioned** more often than anything else, because it is invisible in the feature comparison. And **retention defaults are treated as decisions already made**, when version and trash retention frequently multiply file storage by a factor that was never chosen.

13.6 What actually needs high availability

Ranked, because budget for redundancy is finite:

1. **Identity.** Everything else depends on it, and its outage is a total outage.
2. **Mail receipt.** Mail that is not accepted is lost or bounced; a queue at the edge buys time for the store to recover.
3. **File access**, where it carries operational work.
4. **Chat**, where it has replaced mail for coordination.
5. **Video**, where a meeting can move to a fallback.
6. **Collaborative editing**, where the document is safe in storage and editing can wait.

Borrow the German study's approach to availability requirements: quality goals were reduced to three, of which one was 24/7 availability, stated as a requirement before any product was considered [Standtke and Tiede 2024a]. Declaring the target per service, before selection, prevents the common outcome where every service is engineered to the highest requirement any of them has.

14. Data migration mechanics

The part that consumes the schedule. Each subsection lists what moves, what does not, and where the surprises are.

14.1 The general rules

Migrate the data, retire the workflow. Server-side rules, custom forms, workflow automations, and macros usually have no equivalent and should be inventoried, triaged, and mostly not recreated. The French study's function-level analysis (§20.2) is the tool for deciding which few need rebuilding.

Leave the archive where it is, if you can. Both German-language sources bound the conversion problem by leaving existing documents in their existing formats and applying the new format only to new material [Kern et al. 2022; Staatskanzlei Schleswig-Holstein 2024]. The same logic applies to mail archives and old file shares: a read-only historical store on the old system, with a documented retention end date, is frequently cheaper and less risky than migrating data that will never be opened.

Migrate in waves by organisational unit, per §15.2, and make each wave reversible until it is confirmed.

Measure twice. Every migration tool undercounts. Reconcile item counts and byte counts on both sides, per user, and treat unexplained differences as blocking.

14.2 Mail, calendar, and contacts

What moves cleanly. Mail messages and folder structures move over IMAP with well-understood tooling; `imapsync` and its equivalents handle incremental synchronisation, and a low-downtime cutover rests on that possibility. Run it repeatedly over days, then a final pass during a short freeze. Calendars and contacts move as iCalendar and vCard, or over CalDAV and CardDAV where both sides support it.

What moves with care.

- **Recurring events with exceptions.** A weekly meeting with three moved instances and one cancellation is the standard test case, and iCalendar implementations diverge on it. Test it explicitly.
- **Meeting organiser identity and attendee responses.** After migration the organiser's address may have changed, which can break updates to existing meetings.
- **Delegation and shared mailbox permissions.** These are stored in the source system's own model and generally have to be re-created from an inventory rather than migrated.
- **Distribution lists.** Directory-derived lists move with the directory; manually maintained ones need extraction.
- **Personal archives** held in local files on user machines, which are invisible to a server-side migration and are frequently large. This is a client-side task and is covered in the companion guide.

What does not move. Server-side rules; out-of-office history; retention and hold state; journaling configuration; and anything held in a third-party archive product, which is a migration of its own and should be scoped separately.

The cutover mechanics. Mail routing does the actual switching. The pattern that works is to point the public mail exchanger at a routing layer that can deliver to either system per recipient, migrate users in waves, and update the routing table as each wave completes. This keeps a single external address space throughout and makes rollback per user rather than per estate.

14.3 Files and shares

What moves. File content moves with any competent copy tool. The metadata is the problem.

Permissions. §6.4 sets out the model mismatch. The mechanics: extract the existing ACLs to a readable inventory before copying anything; map security identifiers to the new directory's identities, which requires the identity migration to have happened first; and decide per share whether to transcribe the ACL or rebuild it from a corrected group model. Transcription is faster and carries the accumulated exceptions forward. Rebuilding is slower and is usually right for shares that matter.

Other metadata that is silently lost unless specifically handled: creation and modification timestamps; ownership; alternate data streams; long paths beyond the legacy limit; and characters legal on one filesystem and not the other. Each of these produces a small number of failures across a large estate, which is why per-user reconciliation counts matter.

SharePoint sites deserve their own warning. A site is files plus lists plus views plus workflows plus permissions plus customisation. The files move. The lists move as data with their structure needing rebuilding. The workflows and customisations do not move at all. Inventory sites by what they actually are: most are file shares with a web interface and migrate as file shares; a minority are applications, and those are application migrations that happen to be hosted in SharePoint.

14.4 Chat and meeting history

Chat history from a proprietary system is effectively non-portable, and attempting it consumes effort disproportionate to the value.

The realistic approach is a policy decision more than a technical one. Decide the retention obligation for historical chat; if there is a legal obligation, export to an archive format for

compliance search and keep it out of the new system; if there is not, set an end date on read-only access to the old system and start the new one empty. Recorded meetings and their transcripts are files and migrate as files, subject to the retention question.

Users complain about this once and adapt quickly, provided the decision is announced in advance with the reason, per §15.4.

14.5 A migration runbook skeleton
Per wave, per service. The value lies in having it written before the first wave, more than in its detail:

1. **Pre-flight.** Source inventory extracted and counted. Target capacity confirmed. Identities present in the target directory. Users notified with dates and expected changes.
2. **Bulk sync.** Repeated incremental synchronisation until the delta is small. No user impact.
3. **Freeze.** A short, announced window in which the source is read-only for the wave's users.
4. **Final delta and switch.** Last synchronisation pass; routing or client configuration updated.
5. **Reconcile.** Counts compared per user. Discrepancies investigated before sign-off.
6. **Confirm.** A defined period during which rollback remains possible, with the criteria for invoking it written down in advance.
7. **Decommission.** The source data for that wave marked for retention or deletion per policy, on a scheduled date.

Steps 5 and 6 are the ones under time pressure to skip, and they are the ones that prevent a silent data-loss incident from being discovered months later.

15. Coexistence patterns

Everything above assumes a period, probably measured in years, during which both stacks run. Coexistence absorbs the effort, and the source studies underweight it.

15.1 Federate identity first
Put both stacks behind one authentication broker before migrating any service, per §5.5. This makes every later move smaller and reversible. The provisioning direction matters: pick one system as the source of truth for identity and provision outward from it, since two authoritative directories converge on divergence.

15.2 Migrate by organisational unit
Calendar and chat interactions concentrate within teams. Moving a whole unit at once keeps the awkward boundary at the edges of collaboration instead of through the middle of it. Moving a service across everyone at once puts the boundary everywhere.

Select early waves for tolerance and influence, per §20.4; simplicity alone is the wrong criterion. The IT department itself is the correct first wave, and the French evidence in §22 explains why.

15.3 The calendar boundary
The specific hard problem from §7.4, with the realistic options:

Pattern	How it works	Cost
Unit-at-a-time with declared degradation	Free/busy works within each system; cross-boundary lookups fail or show as busy	Users at the boundary lose detail; simplest and most common
Shared external calendar publication	Both systems publish free/busy to a common location that each can read	Delay in updates; partial fidelity; some engineering

Pattern	How it works	Cost
Bidirectional federation	Direct free/busy interoperation between the systems	Expensive, fragile, and rarely completed; treat proposals sceptically
Short coexistence	Compress the transition so the gap is tolerable	Requires migration capacity; often the cheapest overall

The pattern that fails is a bidirectional federation attempted mid-migration by a team that also has to deliver the migration. If the coexistence period is long and the calendar boundary matters, buy that integration from someone whose product it is, and hold them to a working demonstration against both systems before signing.

15.4 Declare what will be worse

Cross-boundary free/busy, some file fidelity, presence, and chat history will degrade during coexistence. An organisation that publishes this in advance, with an end date, spends its credibility once. One that lets users discover it spends it continuously.

The publication should be specific: what breaks, for whom, when it starts, when it ends, and what to do instead in the meantime.

15.5 Standardise formats before moving systems

Both German-language sources make document-format conversion a prerequisite instead of a consequence. Schleswig-Holstein names the ODF switch as a milestone in its own right, and the German study specifies that existing documents stay in their current format while new documents are written in the open one [Staatskanzlei Schleswig-Holstein 2024; Kern et al. 2022]. This bounds the conversion work, which is otherwise unbounded, and it can begin years before any server moves.

16. Backup, continuity, and the sovereign reserve

A migration transfers responsibility for continuity from a supplier to you. This is the least glamorous chapter and the one whose omission does the most damage.

16.1 What changes

Under a hosted incumbent, retention, replication, and a large part of recovery are the supplier's problem, bounded by their service description. Self-hosting moves all of it to you, including the parts that were invisible: geographic replication, point-in-time recovery, immutable retention against ransomware, and the restore testing that proves any of it works.

Budget for this explicitly. It is a common omission in migration business cases and it is not small.

16.2 What to protect, per service

Service	What must be recoverable	Notes
Identity	Directory contents, schema, credentials or their derivation, federation config	Recovery order matters: identity restores first
Mail	Message stores and their indexes, plus configuration	Index rebuild time can dominate recovery time
Files	Content, permissions metadata, and version history	Extended attributes must be preserved; verify the tooling does
Sync and share	Application database *consistently with* the file store	The commonest failure: database and storage backed up at different moments
Collaborative editing	Nothing stateful, if configured correctly	Documents live in the file store

Service	What must be recoverable	Notes
Chat	Database and media store, consistently	Media is the bulk
Configuration	The whole configuration-management repository and secrets	Rebuild is impossible without it

The consistency requirement in rows four and seven is the one that invalidates backups without anyone noticing. An application database backed up at one instant and an object store at another produce a restore in which the database references files that do not exist. Either quiesce both, or use storage-level snapshots taken together, and test the restore.

16.3 The disciplines

Three copies, two media types, one off-site, in the classical formulation, with the modern addition that at least one copy must be **immutable** for its retention period. Ransomware that reaches the backup system converts a recoverable incident into an unrecoverable one, and immutability is the control that prevents it. The German study's reference environment uses vaulted snapshots replicated between data centres, with backup data held on a technologically different system from the primary [Kern et al. 2022].

Restore testing on a schedule, with the result recorded. An untested backup is a belief. A tested one is a control. Test at three levels: a single item, a single service, and a full-environment rebuild from configuration plus data.

Recovery objectives declared per service, per §13.6′s ranking, and engineered to, never assumed.

16.4 Rebuild from configuration

The property to engineer for is that the environment can be rebuilt from a configuration repository plus a data restore, with no undocumented state. A credible disaster recovery rests on that property, and so does the ability to replace a supplier: a rebuildable environment is the handover artefact §3.2′s test asks about.

It requires that everything is in the repository: infrastructure definitions, application configuration, and the procedures. Secrets sit in a secret store with their own recovery path, which needs testing separately and is frequently the single point of failure in an otherwise good design.

16.5 The sovereign reserve

The Swiss study closes with a recommendation that went largely unnoticed:

> es ist ratsam, Open-Source-Lösungen als eine Art «stille, souveräne Reserve» für Notfallsituationen zu entwickeln [Standtke and Tiede 2024a]

A quiet sovereign reserve, developed now because "die Systeme und das notwendige Know-how im Ernstfall nicht so schnell aufgebaut werden können". The argument is a business-continuity argument and not an ideological one: the capability to run independent mail, conferencing, and collaboration infrastructure cannot be acquired at the moment it is needed.

For an organisation not ready for a full migration, this is the minimum viable version, and it is a defensible programme in its own right:

- A small, independently operated stack covering mail, chat, conferencing, and document collaboration for a defined critical group.
- Kept current, exercised periodically, and integrated into continuity planning as a documented fallback.
- Staffed by people who work on it regularly enough to retain the knowledge.
- Sized for the crisis population, never the whole organisation.

It costs a fraction of a migration, it produces the skills a later migration would need, and it is justifiable on continuity grounds alone without any argument about sovereignty. It is also the cheapest available answer to §2.6′s third item.

17. Security, hardening, and compliance

17.1 The shift
Self-hosting moves the security boundary. Under a hosted incumbent, patching the service, hardening the platform, and monitoring for compromise are the supplier's obligations. Afterwards they are yours, and the attack surface is public: mail, web, and real-time services must be reachable from outside.

This is the argument that will be made against the migration internally, and it deserves an answer. The obligations are known, the controls are standard, and the capability is staffable, provided it is budgeted from the start; added after an incident, it is too late.

17.2 Baselines
Adopt a published baseline. Writing one produces a weaker result. The German study derived its requirements from the national IT baseline protection catalogue alongside the existing platform's requirements [Kern et al. 2022], which is the right pattern: an external baseline plus the organisation's own additions, with deviations recorded and justified.

The layers to cover: host hardening; container or hypervisor configuration; network segmentation between edge, application, and data layers; and per-service configuration.

17.3 Mail deserves its own paragraph
Mail is the service where self-hosting most often goes wrong in a way that is visible to the outside world.

Outbound: sender authentication records published and correct, message signing configured with a key rotation plan, and a policy record moved to enforcement only after monitoring reports confirm it is safe. Reputation matters and takes time to build; plan the sending-IP transition; one step will not do.

Inbound: transport security enforcement, filtering, and rate limiting. Filtering quality is a real difference from a large hosted service, which has visibility across an enormous corpus. Expect to buy filtering as a service or to invest in tuning, and measure the false-positive rate, because a spam filter that quarantines legitimate mail destroys confidence in the migration faster than any outage.

17.4 Identity and secrets
Phishing-resistant authentication factors for administrative access at minimum, and for everyone if the migration provides the occasion. Administrative access separated from ordinary accounts. Session lifetimes need choosing, since §13.5 notes they drive identity load as well as risk.

Secrets belong in a secret store with rotation and audit, never in the configuration repository. Certificate issuance and renewal automated end to end, because manual certificate renewal is the most reliably recurring self-inflicted outage in this class of system.

17.5 Logging and detection
Centralised logging that remains available when the host that produced it is lost, with retention set to the compliance requirement and coverage of authentication events, administrative actions, and mail flow at minimum. Detection content matters more than log volume: a small number of well-chosen alerts that someone actually responds to beats exhaustive collection that goes unread.

17.6 Compliance
Scope these with someone competent, and do not assume they are covered:

- **NIS2** for entities in scope, which imposes governance, risk-management, and incident-reporting obligations on the estate.†
- **The Cyber Resilience Act**, and its dates are close enough to matter to a programme being planned now. **Reporting obligations apply from 11 September 2026**; the remaining obligations from **11 December 2027**. Article 24 sets the duties of an "open-source software steward": maintain a cybersecurity policy covering secure development and vulnerability handling, cooperate with market surveillance authorities, and report actively exploited vulnerabilities and severe incidents, with an early warning inside **24 hours**, full notification inside 72 hours, and a final report within 14 days for a vulnerability or one month for an incident. Article 64(10) exempts stewards from administrative fines, which manufacturers do not enjoy. The practical question for a migrating organisation is which of those two roles it occupies, and §19.3′s decision to fund or contribute upstream can move it across that line.
- **Data protection**, where self-hosting typically simplifies the transfer analysis and does not remove the obligations.
- **Sector-specific retention and audit obligations**, which usually bind mail hardest and which §14.2 flags as frequently living in a separate archive product.

A migration is a good moment to discover that a compliance obligation was being met by a supplier's feature that had never been documented.

17.7 The comparison that will be made

Someone will say that the incumbent's security team is larger than your entire IT department. They will be right, and the argument still does not go where they think.

What is true in the incumbent's favour. The incumbent brings threat intelligence derived from a global telemetry corpus that cannot be reproduced; a dedicated response capability at a scale few organisations fund; automatic patching of the service layer, removing a class of failure that §18.2 makes your problem; physical security and supply-chain assurance at the datacentre; and the risk-based authentication signals §5.6 already concedes are not reproducible.

What is true in the other direction, and is usually left out. Your estate is a smaller and less interesting target than a platform hosting a large share of an economy. A compromise of a shared platform can include you without any failure on your part, and tenancy in a large target is itself an exposure. You control your own patching window and your own blast radius. Your data is not aggregated with anyone else's. And the incumbent's own services have had material security incidents, so the comparison is between two imperfect operators rather than between risk and safety.

The analytic point that resolves it. The security objection is an argument about **who operates the service**, and not about which software it runs. Open-source software operated by a competent European provider under a contract with security obligations (§19.2) sits on the same side of that argument as the incumbent. The objection is therefore answered by §12 and not here.

Three practical consequences:

For an organisation that fails §18.4′s capability test, the objection is **correct**, and the right response is §12.1′s managed private cloud, and not staying with the incumbent. Conflating "we should not self-operate" with "we should not migrate" is the most common error this argument produces.

For an organisation that will self-operate, budget the security capability explicitly and early, per §21.4. An estate that acquires the obligation without the budget will validate the objection within two years.

For everyone, note that the migration also *removes* security exposures: tenant-wide assistants processing all content (§10.1), telemetry leaving the estate, and the concentration risk

of a single supplier holding identity, content, and communications together. Those belong in the comparison and are routinely omitted from it.

18. Operations

18.1 Observability before production
The monitoring, logging, and alerting stack should exist before the first production service, since retrofitting it means operating blind through the riskiest period. Instrument the four layers of §13.1 separately: identity availability and authentication latency; database health and replication lag; per-application health and queue depths; and edge availability from outside the network.

The single most valuable check is an end-to-end synthetic test per service, run from outside, that exercises the real path including authentication. Component-level green with user-facing red is the normal failure mode.

18.2 Patching and upgrade discipline
The German study's rule from the client side applies to the back end as well: **keep the delta between upstream packages and your own as small as possible**, and ensure collaboration with the supplier is contractually possible [Kern et al. 2022]. Every local modification is a permanent maintenance cost and a step toward the accidental fork that §2.5 warns about.

Practical form: a staging environment that matches production closely enough to be predictive; a defined patch cadence with an emergency path for security fixes; upgrades rehearsed in staging with the restore path tested; and version currency treated as an operational metric, since an estate becomes unsupportable by falling behind, without any single decision to that effect.

18.3 Change and capacity
A change process proportionate to the estate, with the emphasis on reversibility more than on approval. Capacity reviewed against the §13.5 quantities on a schedule, because the growth terms (file versions, chat media, mailbox size) compound out of sight.

18.4 The capability question
The break that determines whether an organisation should self-host is whether it can staff a platform capability, and seat count is a poor proxy for it. The test is whether the organisation can sustain, for each of the five layers of §13.1, at least two people who can operate it and one who can change it, through holidays, illness, and resignation.

An organisation that cannot should buy the stack operated by a supplier, which is a legitimate outcome that preserves most of the benefit in §2.6, since the exit threat comes from the software being replaceable and not from operating it yourself. The German study reaches the same place from the supply side, observing that the vendor concept dissolves into distributors, communities, and enterprise support partners, and that support for general system components should rest on IT generalists who are independent of any manufacturer [Kern et al. 2022].

Its personnel warning applies equally to the transition from project to operations: role and job descriptions need re-examining and rewriting, and the organisational structure must be able to absorb a rapid increase in personnel demand at the moment the platform ships in volume [Kern et al. 2022].

18.5 The staffing model
§18.4 gives the coverage rule: for each layer, two people who can operate it and one who can change it. This section gives the shape that rule implies, without inventing headcount figures the corpus does not support.

Capability	Small (~250)	Medium (~2,500)	Large (~25,000)
Platform and infrastructure	Bought	Shared post	Dedicated team
Identity and access	Bought	Dedicated post	Dedicated team
Mail and groupware	Bought	Dedicated post	Dedicated team
Collaboration and storage	Bought	Shared post	Dedicated posts
Real-time communication	Bought	Shared post	Dedicated post
Security operations	Bought or shared	Dedicated post	Dedicated team
Service desk	Existing, retrained	Existing, retrained	Existing, retrained
Supplier and community management	Part of a role	Part of a role	Dedicated post

"Bought" means the capability comes from a supplier under §19, which is the normal and correct answer at the small profile and is why §12.5 recommends managed hosting there.

Four things the German study adds that are easy to miss and expensive to discover late [Kern et al. 2022]:

Role and job descriptions have to be rewritten, and this feeds both internal development and external recruitment. It is an HR workstream with its own lead time, and it usually has no owner.

The transition from project to line organisation needs preparing early, because the moment of highest personnel demand arrives when the platform ships in volume, and the receiving structure has to exist before then.

A supplier and community management function is a named role. Treated as a residual duty, it goes undone. The study recommends creating it specifically, because cooperating with upstream projects is a capability most organisations lack and because §19.3′s funding obligation needs an owner.

Training may not be purchasable in your language, with tailored programmes and certification paths available from few providers. That is a procurement lead-time item, and certification paths matter for recruitment as well as for training.

Key-person and attrition risk. A migration concentrates new and marketable skills in a small group at exactly the moment the organisation depends on them. Two mitigations: the coverage rule above, applied and audited, never assumed; and planned rotation of the pilot and early-wave work through more of the team than strictly needs it, which is the same mechanism §20.3 recommends for a different reason.

19. Procurement and contracting

The contract either preserves exit cost or gives it away, and it is the chapter most often left to people who were not in the technical evaluation.

19.1 Tender for support

The German study's approach is instructive and transfers beyond its legal setting. Because the software is freely available, what is actually being procured is support, integration, and development. They therefore tender the services and let the product follow, with award criteria including the contribution to digital sovereignty, integrability into existing technical and staff processes, enterprise support quality, and "Planbarkeit, Verlässlichkeit und die Möglichkeit, technische Anforderungen einzubringen" [Kern et al. 2022].

That last criterion is the one to weight, since it is the difference between buying software and having a supplier.

19.2 What to put in the contract

- **Exit assistance**, specified as a deliverable with a defined duration, scope, and price, invoked at your discretion. Without this, the exit path in C5 is theoretical.

- **Data export** in documented formats, on demand, at no incremental charge, with a tested procedure, never a promise.
- **Named second sources.** Ask the supplier to identify who else could take over, per §3.2, and record the answer in the contract's context. A supplier unwilling to answer has told you something.
- **Upstream contribution commitments**, so that fixes you pay for reach the public codebase and do not become a private fork you own forever. This is the contractual form of the French study's argument for developing with the community, which "assure le maintien de la fonctionnalité dans les versions futures du logiciel" [Lecrivain et al. 2022].
- **Support obligations** expressed as response and resolution targets by severity, with escalation into the upstream project defined.
- **Security obligations**: vulnerability notification, patch timelines, and a coordinated disclosure position.
- **No metering traps.** Verify that the pricing model cannot become per-seat by another name, and that development work is charged as work rather than as capacity.
- **Escrow** where a single supplier is critical, noting that escrow is worth little without the build pipeline and documentation to use it.

19.3 Funding upstream

§2.5 established the obligation. In practice it takes three forms, and an organisation should actively choose among them, since defaulting to none is the common outcome: a support subscription with the vendor, which is the simplest; funded development of specific features through a supplier, which is the French study's recommended route for remediation; and direct contribution or membership in the project's governance, which is the only form that buys influence over direction.

Schleswig-Holstein's formulation ties this to industrial policy: funds "used to finance development orders or contracts for support instead of licence fees", with the regional digital economy brought in, so that "the creation of digital sovereignty through the use of OSS solutions is linked to industrial policy for the digital economy" [Staatskanzlei Schleswig-Holstein 2024]. For a private organisation the equivalent argument is supply-chain resilience: the supplier base you will need in five years is the one you fund now.

19.4 Reference checks, and what to ask

C2 in §3 asks for "a named public reference at your scale". This is the criterion organisations most often skip, and the one most predictive of whether a deployment will behave as promised.

Ask the wrong questions and you learn nothing. "Are you satisfied" produces a useless answer, because the reference was selected by the supplier for their answer to it. The questions that produce information:

- What did you underestimate, and by how much?
- What broke in the first six months that you had not anticipated?
- What is your service-desk volume now against before, per hundred seats?
- Which features did you expect to use and do not?
- Show us a runbook, or an incident review.
- Who else did you evaluate, and what decided it?
- If you were starting again, what would you sequence differently?
- What have you had to fund upstream, and what did that cost?

Named public references from the source studies. These are recorded in the corpus and can be approached directly. Their value is that they were published by the deploying organisations or by an independent study rather than assembled as a sales list.

Deployment	Scale and role	Source
Swiss Federal Court, Zimbra	~600 users, ~CHF 25/user/yr, one part-time administrator	Standtke and Tiede (2024a)
Swiss Federal Court, Jitsi	Production video with a custom telephone dial-in extension	Standtke and Tiede (2024a)
Berlin education authority, Open-Xchange	~33,000 teachers	Standtke and Tiede (2024a)
CERN, Open-Xchange	Migration to a FOSS-focused mail service, with published migration notes	Standtke and Tiede (2024a)
Cologne schools, Univention	261 schools with central identity management	Standtke and Tiede (2024a)
Basel schools; Gummersbach city administration; Barcelona	Municipal and education deployments	Standtke and Tiede (2024a)
ETH Zurich physics department, Matrix	Departmental chat infrastructure	Standtke and Tiede (2024a)
gematik and the German armed forces, Matrix	National health telematics messenger; forces messenger	Standtke and Tiede (2024a)
French government, Tchap (Matrix)	Government messaging service	Standtke and Tiede (2024a)
Schleswig-Holstein	~25,000 workstations, published strategy and feasibility study	Kern et al. (2022); Staatskanzlei Schleswig-Holstein (2024)
French Gendarmerie; Fontaine and Échirolles	Linux desktop, field-interviewed post-migration	Lecrivain et al. (2022)
La Suite numérique (DINUM, France)	**500,000+ agents monthly across 15 ministries**; Tchap alone 600,000	lasuite.numerique.gouv.fr, 2026
openDesk (ZenDiS, Germany)	Federally-governed suite, v1.16.1 at 30 June 2026	opendesk.eu, 2026
Thuringian state administration, OpenTalk	Video conferencing for a Land administration	opentalk.eu, 2026

The negative reference is the valuable one, and you will have to ask for it. Suppliers do not volunteer the deployment that went badly. Ask directly what it was and what happened to it; a supplier claiming none is either new or not answering. Note also that the table above lists CERN against one product, and that a reference is evidence about a **product at a scale**, never about an organisation: the same institution can be a good reference for one component and a cautionary one for another.

Be sceptical of thin evidence in both directions, including in this guide. A single line in a comparison table, with no methodology behind it and no way to tell a product limitation from a pilot misconfiguration, is not a finding. It is a prompt to go and ask.

The last three rows carry the most weight for a sceptical reader, because they are recent, public, and large. An organisation being told that open-source collaboration does not work at scale has a ready answer in a national administration running it for half a million people across fifteen ministries. A single datapoint like that, with a number attached, is worth more than a page of satisfied testimonials, and suppliers do not volunteer them. Ask directly for the deployment that went badly and what happened to it. A supplier who claims none is either new or not answering.

20. Migration method and change management

20.1 The sequencing doctrine

Two of the sources arrive independently at the same rule, and it is the most transferable finding in the corpus.

> sodass die Umstellung des Betriebssystems schließlich nur noch ein “kleiner” Schritt zu einer Open-Source-basierten IT-Landschaft … darstellt. [Kern et al. 2022]

Move every layer that can move while the incumbent platform is still in place. Schleswig-Holstein states the rationale for the office suite explicitly: LibreOffice “can be run on terminals based on either Windows or Linux and is therefore the ideal office communication environment”, and it is installed “parallel to the existing Microsoft Office package” [Staatskanzlei Schleswig-Holstein 2024]. Each layer that moves early is one fewer variable in the layer that moves late.

For the back end the ordering that follows is: identity federation, then file services, then collaborative editing and sync, then mail and groupware, then chat and video, then telephony and print. This is roughly ascending in coupling to user behaviour and in coupling to hardware.

20.2 Inventory by function

Every source makes inventory a hard precondition. The French study calls it “préalable incontournable”; Schleswig-Holstein operationalised it as a licence-management system introduced in 2020, before migration, giving per-department needs “correct to the day” and used explicitly to size future migrations; the German study had to commission a cataloguing project because central IT did not know what applications the agencies ran [Lecrivain et al. 2022; Staatskanzlei Schleswig-Holstein 2024; Kern et al. 2022].

The French study then makes the analytic move that the others miss. Knowing that an application is installed tells you nothing about what it is used for. Their worked example takes a spreadsheet and postulates three populations: a large majority doing simple arithmetic; a smaller group using moderately complex functions; and a small group “créant et maintenant des microsystèmes d’information non régulés fonctionnant à base de macros riches et de fichiers interconnectés” [Lecrivain et al. 2022].

Reasoning at the application level concludes: move the first two groups, keep the incumbent for the third. Reasoning at the function level concludes something better: the first group needs a calculator, the second is served by any spreadsheet, and the third’s need “mérite un autre outil qu’un tableur”. The unregulated macro estate is a governance problem that the migration has surfaced, and treating it as a reason to keep the incumbent preserves both the dependency and the problem.

20.3 The triptych

The French method runs every scope through three phases: **pilot, remediation, deployment**. Remediation begins during the pilot, as soon as problems appear, and continues after it ends. Scopes are defined by function and sequenced by assessed difficulty, simplest first, which also trains the IT department on easy cases before hard ones [Lecrivain et al. 2022].

20.4 Who is in the pilot

The French study’s most counter-intuitive and best-evidenced recommendation: recruit **both enthusiasts and the most reluctant users**. The reluctant are not given the new system. They are interviewed for the reasons behind their reluctance, which either surface genuine functional gaps early enough to remediate or supply the communication material for everyone else. Both groups must be influential in their teams, and the reluctant must also be technically expert so that their objections track the most demanding uses [Lecrivain et al. 2022].

20.5 Pair the change with a gain

Both the French study and the Schleswig-Holstein strategy independently land on this. Schleswig-Holstein sells its groupware migration on larger mailboxes and easy smartphone access. The French field cases pair migration with new hardware and with new services that are easier on the new platform, and the study is explicit that the size of the benefit matters less than its existence: “Ces avantages peuvent donc être aussi importants qu’anecdotiques du moment qu’ils inscrivent dans l’esprit des utilisateurs un gain” [Lecrivain et al. 2022; Staatskanzlei Schleswig-Holstein 2024].

For the back end specifically, the available gains are mailbox and attachment size limits, external sharing that does not require a workaround, meeting links that work for outside participants without a client install, and search that works. Pick one, deliver it in the same release as the disruption, and say so.

20.6 On deadlines

The French study argues against progress indicators and completion targets on principle, recommending “d’éviter les indicateurs d’avancement et la logique de délai” on the grounds that a deadline on a necessarily gradual migration raises training and remediation costs and hardens resistance. Their field cases ran four to seven years, and they reframe the goal away from any metric: the objective is that the new platform “devienne le système par défaut dans l’esprit des personnes” [Lecrivain et al. 2022].

This guide adopts the reasoning and rejects the conclusion for most organisations. The reasoning is sound: a percentage-migrated target creates pressure to migrate users who are not ready, which produces exactly the visible failures that end programmes. An organisation accountable to a board cannot run a multi-year investment with no indicators, and the French recommendation is available to their field cases partly because those cases had no such accountability.

The resolution is to measure readiness instead of adoption:

Indicator	Why it leads
Applications behind the federated identity broker	Every one makes the directory more replaceable
Applications reachable through a browser or standard protocol	Reduces coupling to any client platform
New documents in open formats	Bounds the eventual conversion problem
Services with a tested restore	The continuity precondition of §16
Remediation items closed against opened	The only measure of whether the backlog converges
Scopes through pilot	Progress that does not require migrating anyone

Seats migrated is then an output, never a target.

20.7 Programme shape and indicative durations

§20.6 argues against a completion target for user migration. That is a different thing from refusing to plan, and the distinction matters because the French recommendation is easily misread as licence to leave the programme unbounded, the starting point of coexistence overruns (§22).

What should not have a deadline: the proportion of users migrated. What must be planned: the sequence, the dependencies, the capability build, and the decommissioning of each source system.

Phase 1, Foundation. Six to twelve months. Inventory by function (§20.2); identity federation behind a broker (§5.5); the open-format switch for new documents (§15.5); the hosting decision (§12); observability and backup (§16, §18.1); and the sovereign reserve (§16.5). Every item is reversible, none requires a user to change anything, and the phase

delivers standalone value. An organisation that stops here has taken the middle position in the executive summary.

Phase 2, Substitution. Twelve to thirty months. File services first, as the lowest-disruption change (§6.3); then sync, sharing, and collaborative editing; then mail and groupware, which is the largest single domain and whose shape depends entirely on the Outlook decision (§7.1). Runs in waves by organisational unit (§15.2). This phase carries essentially all of the user-visible disruption and all of the coexistence cost.

Phase 3, Consolidation. Twelve to twenty-four months. Chat and video; then telephony and secure print, both integration projects with contractual lead times outside your control (§9.5, §11); then decommissioning, which is a tracked deliverable and not an afterthought.

Total: three to five years for a mid-sized estate. The two documented field cases ran four to seven years, and both were public administrations migrating desktops as well as services [Lecrivain et al. 2022].

What compresses it: an integrated suite instead of component assembly (§4); managed hosting in place of a self-built platform (§12.5); a single site; a homogeneous application portfolio; and an incumbent agreement that expires conveniently. What extends it: a large departmental application estate whose owners you do not control (§22); regulated workflows with signature or retention obligations; telephony with contractual lead times; and any attempt to run Phase 2 before Phase 1 is finished.

The dependency that most often breaks a plan is that **Phase 2 cannot start before identity federation works**, and identity federation is routinely underestimated because it appears to be a small technical task. It is the phase-gate.

20.8 Stop criteria and off-ramps

§22 rates strategic abandonment as the only catastrophic risk, which makes this guide read as a one-way door. It should not. There is a difference between a programme stopped because the evidence says so and a programme stopped because the sponsor changed, and the way to keep them distinguishable is to write the criteria down before starting.

Pre-commit the criteria. After the fact, any stop looks like failure and any continuation looks like commitment, so the decision becomes political by default. Criteria agreed at approval, reviewed on a schedule, convert that into an evidence question. This is also the strongest available defence against §22.1′s failure mode: when the criteria are public and unmet, “the new leadership prefers the incumbent” is harder to present as “the programme was failing”.

Criteria to pre-commit, each with a threshold you set:

- The remediation backlog does not converge across three consecutive review cycles (§20.6′s leading indicator, used as a gate and not merely reported).
- Service-desk volume per hundred migrated seats does not trend back toward baseline within a defined period after each wave.
- A blocking exception turns out to affect a materially larger population than scoped, which usually means the §20.2 inventory was wrong.
- The capability test in §18.4 fails repeatedly on hiring or retention, with managed hosting (§12.5) rejected or unavailable.
- A critical supplier fails and §19.2′s named second sources decline the work.

Off-ramps, by phase. The programme is designed so that stopping is cheap. This property should be explicit, because approval is reasonable only given it:

Stop after	What you keep	What you lose
Foundation	Federated identity, open formats, the inventory, the reserve. The middle position in the executive summary, and a legitimate destination.	Nothing that was going to pay back later

Stop after	What you keep	What you lose
A domain in Substitution	Every domain already migrated, permanently	The coexistence cost continues for the domains not yet done
Before Consolidation	The bulk of the licence reduction	Telephony and print remain, which was the plan anyway

Every phase delivers standalone value and none requires the next. That is the design property that makes the criteria above safe to write down. Verify it still holds whenever the plan is revised.

20.9 Co-determination and staff consultation

In Germany, Austria, the Netherlands, France and elsewhere, several elements of this programme require agreement with a works council or equivalent body before deployment, and the requirement is on the critical path for the pilot, and no afterthought.

What typically triggers it: changes to the technical equipment staff work with; any capability to access a workstation remotely; the introduction or alteration of systems capable of monitoring performance or behaviour; and the processing of employee-generated content.

Which parts of this programme are affected: remote support (the companion guide's §14), because it permits observation of a workstation; inventory and management agents, because they report what staff have and use; central logging, because authentication and access records are behaviour data; and AI assistance (§10), because it processes employee content and its outputs can be construed as assessment.

Lead time is the issue. Negotiating a works agreement takes months and cannot be compressed by starting the technical work first. Begin the conversation when the pilot is designed. By the time it is ready, the lead time has gone.

One argument that helps. Self-hosting is frequently an easier conversation than a hosted incumbent, because content, telemetry and logs remain inside the organisation and the scope of processing is something the organisation can actually commit to and demonstrate. §10.3 makes the same point about meeting audio and local inference. Where an existing works agreement constrains what may be sent to a third-party service, the migration may relieve rather than create a constraint. Check whether that is so before assuming the reverse.

21. Costs

21.1 What the sources support

Not much.

The German feasibility study declines to quantify at all, restricting itself to a directional indication that excludes development and migration costs entirely because they "konnten zum Zeitpunkt der Machbarkeitsstudie 2020 nicht beziffert werden", and concluding that Linux workplace operation is "langfristig ähnlich wirtschaftlich" as the Windows infrastructure with "Einsparungen … erst nach mehreren Jahren zu erwarten" [Kern et al. 2022]. The French study contains no cost figures whatsoever. Schleswig-Holstein contains two numbers in twenty-one pages and states that eliminating licence costs is not the objective [Staatskanzlei Schleswig-Holstein 2024].

Only the Swiss backend study models costs, arriving at roughly CHF 4.9m, 6m, and 17m over three years for 100, 1,000, and 10,000 users [Standtke and Tiede 2024a]. Those totals are not reproducible from the published document, and the shape of the curve is itself informative: a tenfold increase in seats raises the total by 22 percent, which means the model is dominated by fixed project cost. The seat-count sensitivity of a migration is low. The fixed cost of doing it at all is high.

21.2 The model

`verify/cost_model.py` implements a parametric model on four rules: published list prices only, carried with source and date; **no extrapolation** of per-seat prices beyond their published tiers, with the resulting gaps reported instead of filled; the incumbent cost supplied by the organisation and never assumed; and effort and productivity loss as explicit parameters with placeholder defaults.

The no-extrapolation rule is the substantive part. Vendors publish per-seat prices up to a tier and then quote "project pricing", and the reflex is to fit a curve through the published points. The model reports how many stack components had no published price at each seat count and by how much the row is therefore understated. At 25,000 seats, one component of the illustrative stack has no published price at all, which means the headline figure for the largest estate is the least reliable number in the table. That is the normal situation and it is usually hidden.

21.3 What it produces

Running the default scenarios gives, for an illustrative stack of Nextcloud, Collabora, OpenXchange, Element, and Univention, on placeholder effort assumptions:

Seats	Subscriptions/yr	One-off days	3-yr total	Per seat/yr	Unpriced components
250	17,695	144	300,885	401	3
2,500	336,345	364	2,195,835	293	0
25,000	2,281,895	953	15,489,285	207	1

All figures in EUR, at February 2024 published prices, with a placeholder day rate and a placeholder productivity loss of one user-day each. The 250-seat row omits three components because all three publish their lowest tier at 300 seats, which is itself a finding: below about 300 seats much of this market does not publish prices, and small organisations are negotiating without a reference point.

These numbers demonstrate a method. The only way to obtain a defensible figure is to run the model with your own rates, your own effort estimates, your own quotations, and your own incumbent cost.

21.4 The costs the model does not carry

Enumerated because they are the ones that turn a business case around, and because a model that hides them is worse than no model:

- **Infrastructure**: compute, storage, network, and the redundancy of §13.6. Frequently larger than the subscriptions.
- **Backup and continuity** (§16), including the off-site copy and its egress.
- **Security capability** (§17): tooling, monitoring, and the people to respond.
- **Filtering** for mail, which is often bought.
- **Parallel running**: both stacks, for the duration. The German study notes this explicitly as a longer-term parallel operation to be reckoned with [Kern et al. 2022].
- **Data migration tooling and effort** (§14), the usual point at which schedules slip.
- **Training**, for administrators and users, and the German study's warning that suitable training may not be available in the market at all.
- **Recruitment and retention** for the capability of §18.4.
- **Upstream funding** (§19.3), which §2.5 establishes as an obligation and no discretionary extra.
- **The incumbent's exit terms**: what remains payable, for how long, and what a partial reduction actually saves given the agreement's structure.

That last item comes before anything else is estimated. Enterprise agreements are frequently structured so that reducing the seat count mid-term saves nothing, which means the

migration's savings begin at the renewal date whatever the technical timeline says. Establish that date first; it is often the real constraint on the whole programme.

21.5 The comparison that matters

A three-year comparison against the incumbent's current run rate will usually favour the incumbent, because §2.3′s investment phase falls inside the window and the plateau does not. This is the correct result and it should be presented.

The comparison that carries the argument is against the incumbent's *projected* cost under its observed rate of increase, over the period in which the organisation cannot switch. Schleswig-Holstein's fourfold-in-a-decade figure is the kind of input that belongs on that side of the ledger, read as the expenditure series it is. So does an estimate of what the next bundled capability will cost when it arrives already included and there is no reason to look for an alternative to it.

Performing that comparison needs a number this guide has refused to invent. So the model inverts it, and derives the quantity that requires no assumption about your agreement: the incumbent cost per seat per year at which the migration breaks even over a given horizon. You then compare against a figure you already have.

Seats	Horizon	Flat incumbent price	Price rising 8.7%/yr
250	3 yr	401	368
250	5 yr	269	226
250	10 yr	170	113
2,500	3 yr	293	269
2,500	5 yr	229	193
2,500	10 yr	182	122
25,000	3 yr	207	190
25,000	5 yr	160	135
25,000	10 yr	126	84

EUR per seat per year, undiscounted, on the same assumptions as §21.3 and inheriting all of its limitations, including the unpriced components. Above the figure, the migration is cheaper over that horizon. Below it, the incumbent is.

Which escalation rate, and why it matters. Two series are available and they measure different things. The 8.7 percent used above is a *price* series: the measured annual growth of prices on Europe's locked cloud-and-software base [Asterès 2026]. Schleswig-Holstein's "more than fourfold over roughly ten years" implies 14.9 percent, and that is an *expenditure* series: it compounds unit price with seat growth and with scope expansion, and the source concedes as much when it says the rise "cannot be explained solely by the addition of new services and workplaces" [Staatskanzlei Schleswig-Holstein 2024]. Carrying an expenditure series as a price escalation overstates the migration case, in the one place where this guide's author has an interest in overstating it. The tables therefore use the price series and report the expenditure series as an upper bound. **Substitute your own rate from your own invoice history**, which is a better input than either and one you already possess: three years of renewal quotations is enough to compute it.

Discounting. The comparison above is undiscounted, and that flatters the migration. A migration pays its effort and disruption at the start and the incumbent pays over the horizon, so a euro of migration cost is worth more than a euro of incumbent cost avoided in year eight. Applying a discount rate raises the break-even the incumbent's price must clear:

Seats	Incumbent price path	r = 0%	r = 4%	r = 8%
250	flat	170	188	208
250	price +8.7%	113	129	146

Seats	Incumbent price path	r = 0%	r = 4%	r = 8%
250	expenditure +14.9%	84	98	112
2,500	flat	182	191	200
2,500	price +8.7%	122	131	141
2,500	expenditure +14.9%	90	99	108
25,000	flat	126	132	139
25,000	price +8.7%	84	91	98
25,000	expenditure +14.9%	62	69	75

Ten-year break-even in EUR per seat per year. The three-year figures move by only a few percent under the same rates, because there is little to discount; the ten-year column is where the choice of rate shows, and the ten-year column is the one a migration case leans on. Use your organisation's own hurdle rate. The direction is the point: discounting and escalation push the answer opposite ways, and a case that reports one without the other has chosen its result.

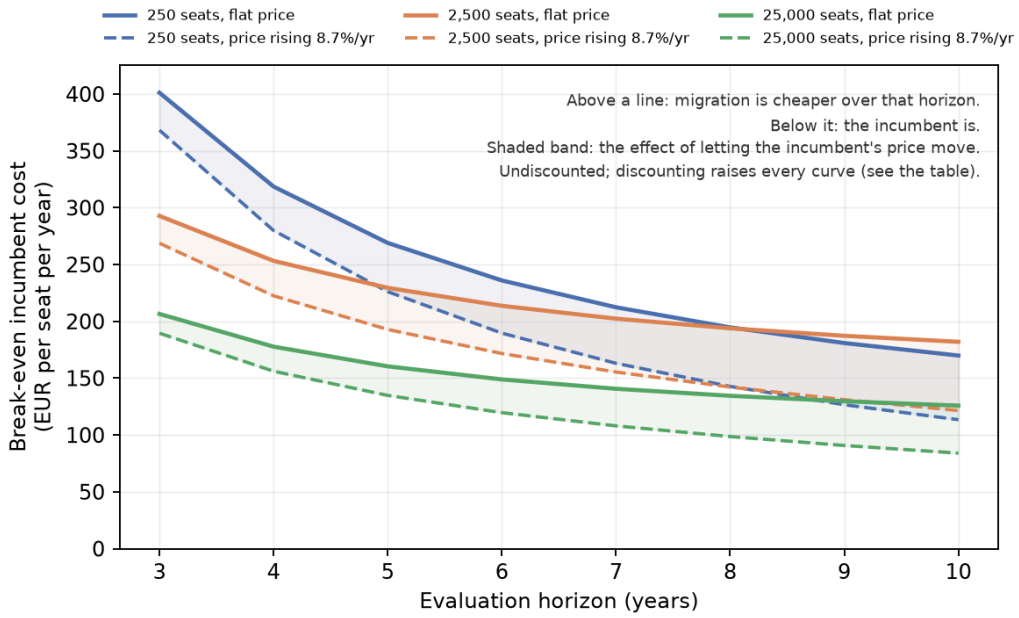

Figure 1. The break-even incumbent cost as a function of the horizon you evaluate over, for the three estate profiles of §1.2. Solid lines hold the incumbent's price flat; dashed lines let it rise at the measured 8.7 percent for a locked base. Both families are undiscounted, so both sit below their discounted equivalents in the table above. Generated by `verify/plot_breakeven.py`*, which asserts agreement with the table.*

Three readings from the table:

Escalation does much of the work. At ten years the break-even figure falls to about two thirds of its flat value once the incumbent's price is allowed to rise at the measured rate for a locked base, and to about half under the expenditure bound. A migration that looks marginal against today's bill can be comfortable against the bill you will actually receive, and the difference is not a modelling trick: it is the exposure described in §2.2. Discounting takes part of that gain back, which is why §21.5 reports both.

Scale helps, and less than expected. Going from 250 to 25,000 seats, a hundredfold increase, cuts the break-even figure roughly in half. The fixed cost of migrating at all dominates, which is §21.1′s finding restated, and it means small organisations face the hardest arithmetic and should look hardest at §4′s suites and §12.5′s managed hosting.

The horizon is the argument. At three years the migration must clear a high bar; at ten it clears a low one. That is the shape of an investment (§2.3), and it should reach a board as both columns side by side with the reason for the difference stated. A single number chosen because it flatters the conclusion is worse than either.

How much of this holds under other parameters. All three readings are asserted on point estimates of parameters this guide does not know, so `verify/cost_model.py` sweeps them: 20,000 draws over declared ranges for the day rate (800 to 1,800 EUR), the effort multiplier (0.6 to 1.8 on the workstream estimates), productivity loss (0.25 to 3 user-days), the loaded user-day cost (200 to 450 EUR), the discount rate (0 to 8 percent) and price escalation (3 to 12 percent). All three hold on 100 percent of the sampled space, because each is structural rather than numerical: escalation lowers the bar whenever the price rises at all, scale lowers it whenever a fixed cost is spread wider, and the bar falls with the horizon whenever a one-off cost is amortised. What the sweep adds is the magnitude, which is not structural: at the median draw the ten-year bar falls to 0.72 of its flat value under escalation, and the 25,000-seat figure is 0.58 of the 250-seat figure. The readings are safe; the adjectives around them are the part to state carefully.

22. Risks

The German study rates its risks on an ALARP matrix, and the ordering is more interesting than the individual entries [Kern et al. 2022].

Complexity and **line-of-business application dependencies** are both rated probable and critical. These are the operational risks, and they are correctly placed at the top. The specific mechanism named for complexity is a shortage of mixed-platform expertise, causing dependencies to be missed and effort to be misjudged.

Cost shifting is rated occasional and minor: licence savings are expected to reappear as support costs, with the admission that the organisation "lacks experience values" to size the shift.

User acceptance is rated occasional and minor. Here the German study and the French field evidence diverge sharply, and the French evidence should be preferred because it is observational rather than anticipatory. The French study found that "La principale difficulté … est la très forte présence d'une résistance au changement dans les équipes de la DSI ; une résistance plus forte même que celle des utilisateurs finaux", caused by the loss of hard-won expertise in the incumbent ecosystem [Lecrivain et al. 2022]. The resistance to plan for is your own IT department's, and the remedy is to let them learn on easy scopes first.

Abandonment of the strategy is the only risk the German study rates catastrophic. Their mitigation is structural and clever: design the new infrastructure to be platform-independent and to serve more than one client platform, so that a change of political direction does not strand the investment. "Die Entwicklung einer nur auf Linux-Arbeitsplätze ausgerichteten Infrastruktur im Rahmen dieses Projekts wäre somit zu vermeiden" [Kern et al. 2022].

Three risks specific to the back end deserve adding to their list:

Capability failure. The organisation cannot staff §18.4′s requirement, discovers this after go-live, and operates a critical estate on one person's knowledge. Mitigation: test the staffing assumption before committing, and treat buying the stack operated as a legitimate answer.

Key-person loss. The migration concentrates new and marketable skills in a small group at the moment the organisation most depends on them, and the market for those skills is tighter than for the incumbent's. Mitigation: §18.5′s coverage rule, audited and not assumed, and planned rotation of early-wave work through more of the team than strictly needs it.

Continuity gap. Backup and restore are designed after the migration and not before, and the first real restore is the first test. Mitigation: §16.3′s testing schedule, with the first full-environment rebuild rehearsed before production.

Coexistence overrun. The transition period extends indefinitely, doubling the run cost and normalising the degradation of §15.4. Mitigation: an end date for each service's coexistence, with the decommissioning of the source system as a tracked deliverable, never an afterthought.

That last one is the quiet killer of otherwise successful programmes: the old system is never switched off, so the savings never arrive and the estate is permanently more complex than either option alone.

22.1 Munich, and what it does and does not show

Munich's LiMux programme migrated roughly 15,000 desktops by 2013 and produced around 18,000 LibreOffice document templates. In November 2017 the city council voted to move back to Microsoft clients [Edge 2017].

The reversal is routinely cited as evidence that Linux migrations fail. The record does not support that reading. The city commissioned a review from Accenture, a Microsoft partner, which "identified several problems, one of which was about an old version of Windows that was still in use, but the biggest problems were organizational rather than technical" [Edge 2017]. Problems unrelated to the desktop were attributed to it anyway: "iPhones did not work with the city's infrastructure, which was blamed on Limux though it had nothing to do with the desktop client." Microsoft moved its German headquarters to Munich in 2013, and the mayor elected in 2014 "ran partly on the idea of switching away from Limux" and "was quoted in some articles as being a Microsoft fan." The article's summary of the decision process is that "a lot of parties had already made up their mind" before the technical evaluation happened.

The competing cost claims are themselves a lesson: the city's IT committee estimated €20m of savings from Linux, while a Microsoft-funded study by HP claimed a €43m increase and was never published [Edge 2017].

The corroborating evidence sits in the German feasibility study, which had no reason to be kind to LiMux. Its only reference to Munich is a technical exchange with former LiMux engineers, from which it took a user-profile-roaming design that "wurde … erarbeitet und erfolgreich erprobt" [Kern et al. 2022]. A programme whose engineering the next serious feasibility study borrows from was not failing technically.

The correct lesson is the one the German study encodes in its risk matrix. A migration of this kind faces its existential risk politically and commercially rather than technically, the incumbent is an active participant in that risk, and the mitigations are governance mitigations: durable executive sponsorship, a platform-independent infrastructure design that withstands a change of direction, and scepticism about vendor-funded evaluations. The French study reaches the same conclusion from the field, listing "les pressions des éditeurs" among its four brakes and recommending executive sponsorship specifically "pour permettre de faire face aux pressions à venir des fournisseurs perdant leur hégémonie sur un client" [Lecrivain et al. 2022].

23. What to do first

In order, and each step stands on its own even if the next one never happens.

1. **Establish the incumbent agreement's renewal date and exit terms.** It usually constrains the programme more than any technical factor. §21.4.
2. **Compute your own break-even figure and your own escalation rate.** Both come from invoices you already hold, and together they decide whether to start the rest of this list. §21.5.

3. **Take the hosting decision explicitly**, and classify your data before taking it. Deferring this lands a sovereignty programme somewhere that does not deliver sovereignty. §12.
4. **Measure actual AI-assistant usage** before accepting it as a requirement. It is frequently bought broadly and used narrowly. §10.4.
5. **Inventory by function.** Not by application, and not by licence count alone. §20.2.
6. **Put applications behind a standard identity broker.** The highest-leverage single move available, and it makes everything after it reversible. §5.5.
7. **Switch the document format.** New documents in an open format, existing documents left alone. §15.5.
8. **Replace the file server with Samba.** Lowest disruption per unit of dependency removed of anything in this guide. §6.3.
9. **Build the sovereign reserve.** Small, exercised, defensible on continuity grounds alone, and it produces the skills the rest of the programme needs. §16.5.
10. **Stand up observability and backup before the first production service.** §16.3, §18.1.
11. **Pilot an integrated suite** against a real organisational unit, and use the pilot to discover your own coexistence problems instead of to evaluate features. §4.
12. **Decide the Outlook question**, because it determines the groupware choice and the sequencing of the largest single domain. §7.1.
13. **Set a decommissioning date for each source system** at the moment its replacement is approved. §22.
14. **Sequence telephony and secure print last**, and budget them as integration projects. §9.5, §11.

Glossary

ACL. Access control list. The per-object permission structure; §6.4 covers the Windows and POSIX model mismatch.

ActiveSync. Microsoft's mobile synchronisation protocol for mail, calendar, and contacts. §7.1 covers its licensing significance.

CAL. Client access licence. A licence charged for using a server that is separately licensed for its operation. §2.1.

CalDAV / CardDAV. Open standards for calendar and contact synchronisation over HTTP.

CRDT. Conflict-free replicated data type. The data structure underlying newer collaborative editors, allowing concurrent editing without a central lock.

CUPS. The standard Unix printing system. §11.

Federation. Two independently operated systems interoperating as peers, as in mail or Matrix. Distinct from a single system with multiple tenants.

Free/busy. Availability information exposed from a calendar without the event detail. §7.4, §15.3.

IdP. Identity provider. The service that authenticates users and asserts identity to applications.

IMAP. The standard protocol for mail retrieval, and the basis of most mail migration tooling. §14.2.

MAPI/HTTP. The protocol native Outlook uses to talk to Exchange. §7.1.

OIDC. OpenID Connect. The modern standard for delegated authentication, layered on OAuth 2.0.

ODF. OpenDocument Format. The ISO-standardised document format; the target of the format switch in §15.5.

OOXML. Office Open XML. The Microsoft Office document format family.

SAML. An older federation standard, still widely required by enterprise applications.

SCIM. System for Cross-domain Identity Management. The standard for provisioning accounts into applications.

SMB. The file-sharing protocol used by Windows file services, implemented on Unix by Samba. §6.

SFU. Selective forwarding unit. The video architecture that forwards streams without mixing them. §9.4.

SPF / DKIM / DMARC / MTA-STS. The mail authentication and transport-security record family. §17.3.

WSJF. Weighted Shortest Job First. The prioritisation method the Swiss frontend study uses; developed in the companion guide's §2.

References

Asterès (2026) *From technological dependency to economic capture: the cost of cloud and software services inflation to Europe*. Study commissioned by Cigref, May. (Source of the 8.7 percent annual price growth on Europe's locked cloud-and-software base.)

Competition and Markets Authority (2025) *Cloud services market investigation: final decision report*, 31 July. London: CMA.

Dixit, A. K. and Pindyck, R. S. (1994) *Investment under Uncertainty*. Princeton: Princeton University Press.

Farrell, J. and Klemperer, P. (2007) 'Coordination and lock-in: competition with switching costs and network effects', in *Handbook of Industrial Organization*, vol. 3. Amsterdam: Elsevier, pp. 1967-2072.

Edge, J. (2017) 'Goodbye Limux', *LWN.net*, 8 November. Available at: https://lwn.net/Articles/737818/

Free Software Foundation (2026) 'AGPL is not a tool for taking freedom away', 15 April. Available at: https://www.fsf.org/blogs/licensing/agpl-is-not-a-tool-for-taking-freedom-away

Klemperer, P. (1995) 'Competition when consumers have switching costs', *Review of Economic Studies*, 62(4), pp. 515-539.

Kern, S., Voss, N., Kempe, C., Blank-Babazadeh, M., Hinz, A. and Gabriel, M. (2022) *Linux-Arbeitsplatz für die öffentliche Verwaltung*. Kiel: Der Ministerpräsident des Landes Schleswig-Holstein, with Dataport AöR. CC BY.

Lecrivain, D., Hemmel, P. and Facia, G.-C. (2022) *Poste de travail Linux: état de l'art et conduite du changement*, v1.2. Paris: Direction générale des Finances publiques, marché de support logiciel libre (SLL-2022, Veille Poste Linux). CC BY-SA.

'The logo trap: ONLYOFFICE's AGPLv3 gambit and what it means for free and open source compliance' (2026), *IT en Recht*. Available at: https://itenrecht.nl/artikelen/the-logo-trap-onlyoffice-s-agplv3-gambit-and-what-it-means-for-free-and-open-source-compliance (author to be confirmed)

Shapiro, C. and Varian, H. R. (1998) *Information Rules: A Strategic Guide to the Network Economy*. Boston: Harvard Business School Press.

Software Freedom Conservancy (2026) 'AGPLv3§7¶4 Empowers Users to Thwart Badgeware', 16 April. Available at: https://sfconservancy.org/blog/2026/apr/16/badgeware-onlyoffice-nextcloud-affero-gpl/

Staatskanzlei Schleswig-Holstein (2024) *Open Innovation and Open-Source Strategy of Land Schleswig-Holstein*, v1.00, 20 November. Kiel: Der Ministerpräsident des Landes Schleswig-Holstein.

Standtke, R. and Tiede, M. (2024a) *Studie zu Open-Source-Alternativen von Microsoft Services und Produkten in der Schweizerischen Bundesverwaltung: Backend-Services*, v1.8,

February. Bern: Berner Fachhochschule, Institut Public Sector Transformation, Digital Sustainability Lab. Commissioned by the Schweizerische Bundeskanzlei. CC BY 4.0.

Standtke, R. and Tiede, M. (2024b) *Studie zu Open-Source-Alternativen von Microsoft Services und Produkten in der Schweizerischen Bundesverwaltung: Frontend-Services (Client-Anwendungen)*, v1.0, February. Bern: Berner Fachhochschule, Institut Public Sector Transformation, Digital Sustainability Lab. Commissioned by the Schweizerische Bundeskanzlei. CC BY 4.0.

Annex A. Working notes (temporary annex)

Internal notes relevant to this guide, removed at finalisation:

- `../notes/common/sources.md`: provenance, method, and limits of each source document, including what the corpus does and does not support.
- `../papers/notes/backend-services-wp.md`: this paper's internal companion, holding version history, verification debt, parked material, and the open questions.

Annex B. Reproduction: scripts, data, and figures

- `../verify/plot_breakeven.py`: Figure 1. Deterministic. Run `uv run --with matplotlib verify/plot_breakeven.py`. Its self-check asserts that the plotted series agree with the §21.5 table, so the figure cannot silently drift from the text.
- `../verify/check_paper_tables.py`: parses every cost table back out of both guides and recomputes each cell against the model, so a pasted table cannot drift from the script that produced it. Run `uv run verify/check_paper_tables.py`; it checks 129 published cells. This is the enforcement of the rule the bullet below states.
- `../verify/score_example.py`: the §3.3 worked scoring example. Run `uv run verify/score_example.py`. Asserts every total in that table, the gate result, and the weight-sensitivity claim. The input scores are judgement and are labelled as such in the paper; only the arithmetic is verified.
- `../verify/cost_model.py`: the parametric cost model of §21. Deterministic, no random seed required. Run `uv run verify/cost_model.py` for the scenario tables of §21.3, or `uv run verify/cost_model.py --self-check` for the assertions alone. The self-check runs on every invocation.
- Published prices are embedded in the script's `CATALOGUE` with their source and date (February 2024, from Standtke and Tiede 2024a), and are stale by construction. Effort, day rate, and productivity-loss figures are placeholders and are labelled as such in the script's output.
- The §21.3 and §21.5 tables are generated by the script and pasted; regenerating them after any parameter change is required, and `check_paper_tables.py` fails if it is not done. The §21.5 break-even figures are derived: they invert the model to solve for the incumbent per-seat cost at which the present value of the two paths matches, at a flat price, at the measured price escalation of Asterès (2026), and at three discount rates. The expenditure series of Staatskanzlei Schleswig-Holstein (2024) is carried as an upper bound and is not used as a price rate.
- The sizing heuristics of §13.5 are planning aids and are not computed by any script. They are not reproducible quantities and are labelled as heuristics in the text.
- Figures: `papers/figures/breakeven-backend.png` (Figure 1), regenerated by `plot_breakeven.py`.