The Open Source Workstation: A Migration Guide for Applications, the Desktop, and the Managed Linux Estate

Stéfane Fermigier

Draft working paper v0.9, 2026-08-30. Guide 2 of 2. The companion volume, `backend-services-wp.md`, covers directory, storage, mail, and real-time communication. Product statuses are stated as of the date given in §1.5; entries marked † were moving at that date and need re-checking before a procurement decision.

Abstract

This guide covers the end-user workstation: the applications on it, the operating system under them, and the machinery that manages both across an estate. It adapts a February 2024 Swiss federal study of client applications and two 2022 studies of the Linux desktop, one German and one French, from their public-administration setting to organisations generally.

It differs from the Swiss source in one structural respect. That study excluded the client operating system by design, stating that "Ein Umstieg auf Linux wird in dieser Studie explizit nicht angeschaut" [Standtke and Tiede 2024b]. This guide treats the Linux desktop as the destination and open-source applications on Windows or macOS as a transitional stage worth completing on its own terms. The two stages are separable, the first is cheap and reversible, and an organisation that stops after it has still removed real dependencies.

Three findings shape the advice. The Swiss study's own prioritisation method produces an ordering that inverts the usual migration plan: stateless single-purpose tools first, and the office suite and mail client **last**, because those carry the format entanglement. The German and French sources agree that the operating system should be the last thing to change, after every layer above it has already moved. And the French field evidence, which is the only observational evidence in the corpus, finds that the strongest resistance comes from the IT department rather than from users.

The management layer decides these projects. Applications reached parity years ago; §§7 to 15 cover provisioning, policy, packaging, inventory, client identity, encryption, printing, remote support, and hardware, which carry the actual work and which the sources treat thinly or not at all.

Executive summary

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

The two decisions are separable, and only one of them is hard

Stage 1, open-source applications on the operating system you already run. No operating-system change, no fleet-management change, no licence cost, reversible at any point, and applications install alongside the incumbents so nothing has to be switched off (§2.2). An organisation that does Stage 1 and stops has moved its documents to an open format, made its applications portable, and removed a set of licence dependencies. That is a legitimate end state and it is the right answer for many organisations.

Stage 2, the Linux desktop. This stage is substantially harder, with the difficulty concentrated in the management layer (§§7 to 15). Applications reached parity years ago; provisioning, policy, packaging, key escrow, and remote support carry the work.

The counter-intuitive finding worth acting on

The Swiss study's own prioritisation method, applied to its own recommendations, puts **the office suite and the mail client last** and stateless single-purpose tools first (§2.1). Media player, password manager, archiver, PDF viewer, image viewer, desktop search: these carry

no shared formats, install in parallel, cost nothing, and build the packaging and support machinery you will need later.

Most migration plans start with the office suite. That maximises disruption at the moment the programme has the least credibility. Starting at the other end means that by the time the office suite moves, the organisation has done this a dozen times and the service desk knows how.

What it costs

The model in §21 derives the incumbent per-seat cost at which a workstation migration breaks even, so that no assumption about your agreement is needed:

Estate	3 years	5 years	10 years
250 seats	364	218	109
2,500 seats	169	101	51
25,000 seats	118	71	35

EUR per seat per year, on the assumptions in §21. Above the figure, migration is cheaper over that horizon.

Two features of the workstation case differ from the back end. **There is no subscription line at all**, because none of the recommended applications charges one, so the cost is effort and disruption. And **productivity loss dominates at scale**: at 25,000 seats, one lost user-day per person is several times the technical effort. Each change-management recommendation in §20 is, read economically, an attack on that single term.

How long

Indicative, derived in §20.9. Stage 1 runs 12 to 24 months and can begin immediately. Stage 2 adds 24 to 48 months and should not begin until Stage 1 is well advanced and the management layer of §7.2 is proven. The two documented field cases ran four to seven years end to end, at the pace of hardware renewal (§15.3).

The five things that actually decide it

1. **Your line-of-business application portfolio**, which you probably do not fully know, because departments bought much of it without telling you (§16.1). This is the workstream that overruns.
2. **Whether you have design, pre-press, or engineering populations.** The field evidence rates professional graphics maturity at 1 out of 5 and says migrating expert users is not currently possible (§4.5). Plan an exception, do not argue with it.
3. **Whether your regulated workflows need signatures in proprietary formats**, which remains unsolved (§4.4).
4. **Whether your IT department is willing.** The only field evidence in the corpus finds internal resistance stronger than user resistance (§20.6). This is the binding constraint and it is rarely the one in the plan.
5. **Whether your hardware and peripheral estate is compatible**, including smartcards, which is a common late blocker (§12.3, §13).

What no source supports

A claim of savings in the first three years. The most rigorous source concludes long-run cost parity with savings only after several years, and the French study contains no cost figures at all and records that cost reduction was an observation after the fact rather than the initial motive (§21.3).

The one thing to do this quarter
Install the incumbent's fonts wherever the office suite runs, and deploy the top of the priority list in §2.1. The first is the cheapest quality improvement available anywhere in this guide (§4.1). The second costs nothing, risks nothing, and builds the capability the rest depends on.

1. Scope and use

1.1 The two stages
Stage 1: open-source applications on the operating system you have. Cross-platform applications install alongside the incumbents, on Windows or macOS. The stage requires no operating-system change and no fleet-management change, in most cases carries no licence cost, and is reversible at any point.

Stage 2: the Linux desktop. This stage replaces the operating system, the desktop environment, and the fleet-management stack underneath. It is substantially harder, and only sensible once Stage 1 is well advanced.

The sequencing is not a hedge. Both German-language sources argue that Stage 1 is the thing that makes Stage 2 affordable. Schleswig-Holstein installs LibreOffice "parallel to the existing Microsoft Office package" because it "can be run on terminals based on either Windows or Linux and is therefore the ideal office communication environment" [Staatskanzlei Schleswig-Holstein 2024]. The German feasibility study generalises it into a doctrine, listing the preparatory measures to be executed while still on Windows so that the operating-system change becomes "nur noch ein 'kleiner' Schritt" [Kern et al. 2022].

An organisation that completes Stage 1 and never starts Stage 2 has still moved its documents to an open format, made its applications portable, and removed a set of licence dependencies. That is a defensible end state.

One correction to the framing, developed in §7: parts of the Stage 2 machinery should be built during Stage 1. Packaging, inventory, and identity work serve both stages, and deferring them until the operating system changes concentrates risk at the worst moment.

1.2 Who it is for
Organisations with a managed estate of a few hundred seats upward. The sources are public-sector, and two of their assumptions do not transfer. Procurement law prevents the German study from naming a distribution, which is a constraint most private organisations do not have. And the French field cases ran on horizons of four to seven years with no completion target, which most organisations cannot replicate. Both are flagged where they matter.

1.3 What is out of scope
Mobile devices. Phones and tablets are a separate estate with their own management platform and procurement, and the realistic advice is to integrate with that platform and not to replace it as part of this programme. Mobile *access* to the migrated services is treated in the companion guide's §7.6, and it belongs in the pilot acceptance criteria of §23.1.

Servers and the services themselves, which are the companion guide's subject.

1.4 Verdict scale
As in the companion guide: **Ready**, **Viable with conditions**, **Emerging**, **Not yet a replacement**. These describe fitness for a role in a managed estate and not software quality.

1.5 State of information
Assessments are current to **January 2026** and were re-checked against the sources. A verification pass against upstream sources was outstanding when this draft was written; claims that were moving carry †. The Swiss study's own star ratings are quoted with attribution where useful and are not this guide's ratings.

1.6 Control

Each table records the steward and where control sits, decomposed as in the companion guide into legal domicile, development locus, and funding dependency. Jurisdiction is one criterion among five and does not get lexicographic priority; the economic channels (switching cost, supplier concentration, rent extraction) are the ones that reliably determine whether an organisation can change its mind.

The client application layer has a **much deeper European supplier base** than the back end. VLC, Recoll, FreshRSS, Penpot, OpenProject, Passbolt, Krita, darktable, Scribus, PeaZIP, Blender, KDE, and The Document Foundation are all European-stewarded, and several are the best option in their category on the merits. There is one exception, and §4.3 treats it at length because its history is instructive about how control moves.

2. Sequencing Stage 1

2.1 The finding worth stealing

The Swiss frontend study ranks its recommendations by Weighted Shortest Job First: cost of delay divided by job size, where cost of delay sums business value, time criticality, and opportunity enablement on a Fibonacci scale. Time criticality is defined by **format entanglement**: low when the tool is stateless, medium for a single format, high "wenn mehrere, mitunter auch miteinander verknüpfte Formate involviert sind". Job size is small for provisioning and configuration alone, large where multiple applications with migration and parallel operation are involved [Standtke and Tiede 2024b].

The resulting order is the useful output, and it inverts what most migration plans do first:

Short term (highest priority). Media player (27), password manager (16), file compression (11.3), PDF reader (9), image viewer (7.4), desktop search (6.8), image editing (6), file transfer (5.3), PDF editing via LibreOffice (5.3), secure messaging client (5.2).

Medium term. Project planning, diagramming, accessible-document creation, OCR, screenshots (all 4.6), accessibility checker (4.2), desktop database (3.6), feed reader (3.6), mind mapping (3.3), note-taking (3.2), **chat and meetings (3.2)**.

Long term (lowest priority). E-book reader, PDF creation (3), calendar (2.8), **the office suite (2.6)**, desktop publishing and vector graphics (2.6), **the mail client (1.8 to 2.0)**.

The office suite ranks 2.6 and the mail client ranks below 2. They are the applications every migration plan starts with, and this method puts them last, because they carry the deepest format and workflow entanglement and the largest job size. Starting with them maximises disruption at the moment the programme has the least credibility.

2.2 What follows

Start with tools that hold no state and share no formats. They are cheap, they build the packaging and distribution machinery you will need later, they let the service desk learn, and each one removes a dependency at almost no risk. By the time the office suite moves, the organisation has done it a dozen times.

The Swiss study's other structural observation supports this: because none of the recommended applications charges a licence fee, they can be installed in parallel with the incumbents, and "es kann mit einer Mehrprodukte-Strategie gearbeitet werden" [Standtke and Tiede 2024b]. Parallel installation removes the need for a cutover on most of this list.

2.3 Adapting the ranking to your estate

The Swiss ranking encodes their business values, which were set by their client. Recomputing it for your organisation takes an afternoon and is worth doing, because the ordering is the method's actual output, and it is sensitive to two inputs:

- **User business value**, which is your own priority for the incumbent application.
- **Job size**, which depends on how many people use the application and on whether a parallel-running path exists.

The two structural terms transfer unchanged. Time criticality really is driven by format entanglement, and job size really is driven by whether migration and parallel operation are needed. An organisation that recomputes and finds the office suite near the top should check how it scored job size.

3. Evaluation framework

The five criteria from the companion guide apply unchanged: functional coverage against functions actually used; maturity at scale; manageability in an estate; supplier depth; and exit path.

Two apply differently on the client. **Manageability** dominates: a client application that cannot be packaged, pre-configured, updated silently, and locked down centrally is not deployable at estate scale whatever its features, and this eliminates otherwise excellent tools. **Exit path** on the client is mostly about file formats, which makes it unusually tractable: an application that reads and writes a documented format has a short exit path almost regardless of its other properties.

A client-specific manageability checklist to apply before any application enters the catalogue:

Question	Why it decides
Does it install unattended, without user interaction?	Anything else does not scale
Can settings be pre-seeded and locked centrally?	Users must not be able to disable security settings
Does it update through a mechanism you control?	Self-updating applications defeat patch management
Does it authenticate against the identity provider?	Otherwise it becomes a separate credential estate
Does it work without administrative rights?	Users should not need them
Is there a policy mechanism (files, registry, dconf)?	The lockdown substrate of §9
Is it packaged for your platforms, or must you package it?	Packaging is a permanent maintenance cost

3.1 Re-validating these tables

The four checks that re-running this assessment requires (liveness, standing, control, and coverage) are set out in the companion guide's §3.4 and apply here unchanged.

One warning specific to this guide: **the client application catalogue ages faster than anything else in either volume.** Server-side components are chosen on multi-year procurement cycles and change slowly; desktop applications in categories such as note-taking, diagramming, and PDF handling turn over quickly, have low switching costs, and are frequently displaced by entrants that did not exist when the previous list was drawn. The standing and coverage checks matter most exactly where the products are cheapest to adopt, which is the opposite of where attention naturally goes.

4. Stage 1: applications

4.1 Office suite

Option	Steward and control	Licence	Supplier depth	Verdict
LibreOffice	The Document Foundation (DE, Berlin); contributors led by Collab-	MPL-2.0	Broad	Ready

Option	Steward and control	Licence	Supplier depth	Verdict
	ora (UK), Allotropia (DE), Red Hat			
Collabora Office	Collabora Productivity (UK)	MPL-2.0	Moderate	Ready; the supported build
ONLYOFFICE Desktop	Ascensio System SIA (Riga, LV), held via a UK entity by a Singapore holding company	AGPLv3 plus disputed §7 additional terms	Moderate	Viable with conditions; see the companion guide's §8.3

Default: LibreOffice, as a supported build with a stated maintenance horizon, deployed on the incumbent operating system first (§1.1).

ONLYOFFICE carries additional terms under section 7 of the AGPLv3, requiring retention of the product logo while granting no trademark rights in it. The Free Software Foundation and the Software Freedom Conservancy both analysed the arrangement critically in April 2026, and it led to a publisher dispute with Nextcloud over a rebranded fork. The companion guide's §8.3 sets out the sequence and its scope. For a desktop deployment the practical exposure is low, since an organisation installing the suite unmodified is not redistributing it; the exposure lands on anyone repackaging it into their own image, rebranding it, or shipping it onward. Check §10.4 before adding it to an internal repository.

LibreOffice is the choice, and both German-language sources select it. The German study calls it "das am weitesten entwickelte Produkt" and assesses the feature set as presenting "außer in Details, wie in Microsoft Office" [Kern et al. 2022]. Schleswig-Holstein names it a milestone in its own right and reports having initiated an accessible-PDF assistant that went upstream, which is a good example of the funding obligation from the companion guide's §2.5 discharged in practice [Staatskanzlei Schleswig-Holstein 2024].

For a managed estate, deploy a **supported build** with a defined support horizon, and do not track the fastest release line. The Document Foundation maintains parallel branches for this purpose, and the commercial builds from Collabora and Allotropia exist to give an organisation a supplier and a maintenance commitment.

Practical points the sources supply and most migration plans miss:

Install the incumbent's fonts. The French field evidence is emphatic and slightly deflating: "Beaucoup de problèmes sont, en fait, résolus en intégrant les polices propriétaires de Microsoft" [Lecrivain et al. 2022]. A large share of the layout complaints attributed to the office suite are font substitution: line breaks move, pagination shifts, and a document that was one page becomes two. Licensing for the font set needs checking, and metric-compatible substitutes exist where it cannot be resolved. Do this before any pilot begins.

Switch the format, leave the archive. The German study's rule bounds an otherwise unbounded task: existing documents stay in their current format, new documents are written in the open one [Kern et al. 2022]. Bulk conversion of a document archive generates work proportional to the archive and value proportional to almost nothing.

Templates are a real project. Munich produced around 18,000 templates over its programme [Edge 2017]. Corporate templates, letterheads, and report formats have to be rebuilt and are frequently owned by a communications function outside IT. Start this early; it is a long-lead item that goes unassigned.

Macros are a governance problem the migration surfaces. The Swiss field scores put the office suite at 4.5 on maturity, with the residual gaps on macros and on presentations. §20.2 sets out the function-level analysis that is the right response to a macro estate.

Interoperability testing needs a real corpus. Take a stratified sample of real documents from across the organisation, including the worst offenders, and render them in both

suites side by side. Complaints without a reproducible document are unactionable, and a corpus turns an argument into a defect list.

4.2 Mail and PIM client

Option	Steward and control	Licence	Supplier depth	Verdict
Thunderbird	MZLA Technologies, subsidiary of the Mozilla Foundation (US)	MPL-2.0	Moderate	Ready
Betterbird	Independent maintainer (CH)	MPL-2.0	Thin	Viable with conditions
Evolution	GNOME project; Red Hat-led contributions	LGPL	Moderate	Ready, notably for Exchange estates
KMail / Kontact	KDE e.V. (DE)	GPL	Moderate	Viable with conditions

Default: Thunderbird, and move it last (§2.1). Evolution instead where clients must talk to an unmigrated Exchange during coexistence.

Thunderbird is the Swiss study's recommendation at four stars, covering Outlook plus the two secure-messaging add-ins in use [Standtke and Tiede 2024b]. Its steward is American, which is a fact to record and not a disqualification: the licence permits forking, the code is community-developed, and **Betterbird** demonstrates that the fork path is live.

Evolution deserves more attention than the Swiss study gives it, because its Exchange Web Services support lets it talk to an unmigrated Exchange while the user runs Linux. That makes it a coexistence tool and not a destination, and a valuable one during a long transition.

Enterprise deployment specifics that decide whether this works:

- **Central configuration.** Account auto-configuration, server settings, and policy locking need to be pre-seeded, or every user becomes a service-desk call.
- **Signatures**, applied centrally and consistently, which is invariably someone's project and is usually discovered late.
- **Address book.** The organisational directory must resolve, over LDAP or CardDAV, with the same completion behaviour users expect.
- **Delegation and shared mailboxes**, on which the client and server capabilities have to be checked together against the companion guide's §7.5.
- **Archives**, which frequently sit in local files on user machines and are invisible to a server-side migration. §17.2.

The sequencing point from §2.1 governs this whole category: the mail client ranks last on the Swiss priority list, because calendar, contacts, free/busy, delegation, shared mailboxes, and mobile sync all meet in mail. Move it after the back end has moved.

4.3 Browser, and where control actually went

Option	Steward and control	Engine	Licence	Verdict
Firefox	Mozilla (US)	Gecko	MPL-2.0	Ready
Chromium	Google (US)	Blink	BSD-3	Ready
LibreWolf	Community	Gecko	MPL-2.0	Viable
Vivaldi	Vivaldi Technologies (NO)	Blink	Partly proprietary UI	Viable with conditions†

Default: whichever of Firefox or a Chromium build your internal web applications test clean against, chosen and standardised while still on the incumbent operating system. The engine is not among the things you are choosing.

No browser engine is under European control today. Every option above renders with Gecko or Blink, and building a third engine from nothing is beyond any single organisation.

The lineage matters, because it is the clearest available case of the mechanism at work. Blink descends from WebKit, which Apple forked in 2002 from **KHTML and KJS**, the rendering and JavaScript engines written for KDE's Konqueror browser by a largely European team working under a German non-profit association. The engine that renders the majority of the web today grew from European free software. Two decades of sustained investment, and with it the stewardship, accumulated elsewhere.

This is not a licensing failure, and the licences worked exactly as written: the code was free, so it could be taken, extended, and built upon, exactly as free software intends. It is a **value-capture** outcome. Whoever funds sustained engineering on a commons ends up setting its direction, and the entity setting the direction is the one whose jurisdiction and commercial interests then bind everyone downstream. This is why §1.6 decomposes control into domicile, development locus, and funding dependency, and why the third of those is the one that moves the other two over time.

Two practical consequences follow, and they point in different directions. For an estate, nothing changes: choose your browser and your supplier, and accept that the engine's stewardship is not among the things you are choosing. For anyone thinking about European technology strategy, the lesson is that **origin is not control**, and funding sustained maintenance converts the first into the second. A European-originated component with no European funding behind it becomes a European-originated component that Europe does not steward.

The German study reports the operational position as unproblematic: both major browsers can be centrally pre-configured through policy files, self-managed CA roots registered, and settings locked so the system-wide default holds [Kern et al. 2022]. Their direction of travel favoured a Chromium-based browser for compatibility with open-source videoconferencing.

Two rules. Choose the standard browser **while still on the incumbent OS**, because the compatibility testing of internal web applications has to happen there anyway. And test cross-platform: the German study specifies test schemes run "aus jedem der eingesetzten Arbeitsplatz-Betriebssysteme" [Kern et al. 2022].

Legacy web applications are the residual problem, and they are usually a small, identifiable set that depends on a retired browser technology. The German study's worked example is instructive: their most widely deployed departmental application had a web client "zu 75 % über Webtechnologien realisiert" with the remaining quarter supplied by a proprietary browser component supported only in one browser on one operating system [Kern et al. 2022]. Applications like this are §16 problems and not browser problems.

4.4 PDF

Option	Steward and control	Licence	Verdict
Okular	KDE e.V. (DE)	GPL	Ready for reading and annotation
LibreOffice Draw	TDF (DE)	MPL-2.0	Ready for editing
pdfarranger	Community	GPL	Ready for page operations
Xournal++	Community	GPL	Viable for annotation and handwriting
veraPDF	PDF Association (DE) and Open Preservation Foundation	GPL/MPL	Ready for PDF/A and accessibility validation
Stirling-PDF	Community, single-maintainer origin	Licence changed in 2024–25†	Viable with conditions†

Option	Steward and control	Licence	Verdict
clawPDF	Community	AGPL	Viable as a virtual PDF printer

The Swiss study recommends **Firefox's built-in viewer** for plain reading, the cheapest possible answer and correct for most users, and **LibreOffice** for editing in place of Acrobat Pro [Standtke and Tiede 2024b]. **veraPDF** is the accessibility and archival validator and matters more than its obscurity suggests, for the reasons in §19.

The gap both Swiss studies record is **qualified electronic signatures in proprietary formats**. The French study puts it directly: handling or producing signatures in proprietary formats "reste cependant encore impossible", and suggests the stakes may justify building something [Lecrivain et al. 2022]. Organisations with a regulated signature workflow should scope this first, because it is the most likely single blocker in the whole application layer. Check three things specifically: the signature format required by your regulator; whether validation as well as creation is needed; and whether the signing credential lives on a smartcard whose middleware is covered by §12.3.

4.5 Notes, projects, diagrams, graphics

Category	Options (steward)	Verdict
Notes	Joplin (community); Nextcloud Notes (DE); Trilium (community); Logseq (community); Standard Notes (Proton, CH)	Joplin Ready; others Viable
Project management	**OpenProject** (DE); **Taiga** (ES); **Vikunja** (DE); Redmine (community)	OpenProject Ready
Diagramming	**Penpot** (Kaleidos, ES); draw.io (JGraph, UK); Inkscape (community); Excalidraw (US); PlantUML / Mermaid (community)	Penpot and draw.io Ready
Raster graphics	GIMP 3.x (GNOME Foundation); **Krita** (Stichting Krita, NL)	Both Ready
Vector and DTP	Inkscape; **Scribus** (community, European-heavy)	Ready
Photography	**darktable** (community, German-led); RawTherapee	Ready
3D	**Blender** (Blender Foundation, NL)	Ready
Mind mapping	Freeplane (community)	Ready

The European supplier base is strongest in this category and where the Swiss ratings hold up best. **Penpot** is their recommendation over LibreOffice Draw and draw.io for Visio replacement at four stars, and it is Spanish [Standtke and Tiede 2024b]. **OpenProject** is German and rated four stars against Microsoft Project. **Krita** is recommended alongside GIMP, which is the right relationship: Krita is a painting application and GIMP is an image editor, and organisations that deploy one expecting the other are disappointed for reasons that have nothing to do with quality.

Two weak spots to plan around.

Desktop databases. The Swiss study rates LibreOffice Base two stars against Microsoft Access, usable "nur mit Mühen". That rating is deserved, and it matters. Departmental Access databases are, like macro-laden spreadsheets, unmanaged information systems that a migration exposes. They are rarely replaced by another desktop database; the realistic destinations are a small web application, a shared spreadsheet where the requirement was never really a database, or a proper application where it was. §20.2 is the analysis that decides which.

Professional graphics. The French expert assessment scores graphics at 2 on criticality and 2 on maturity, and the field assessment drops maturity to **1**, with the flat statement

that “Il est aujourd’hui impossible de migrer des populations expertes de ce domaine sur Linux” [Lecrivain et al. 2022]. This is the strongest negative result in the corpus and arguing with it wastes credibility. For design, pre-press, and marketing populations, plan to keep the incumbent platform, and treat that as a documented exception under §16.4 and close it. The French cases mitigated it by observing that graphics was not their core business; an organisation where it is has a materially different problem.

4.6 AI assistance on the desktop

The economics and the replacement options for generative AI assistance are treated in the companion guide’s §10, because the question is decided at the service layer and not on the endpoint. Two points belong here.

The assistant is a reason people give for not moving, and it is usually untested. §10.4 recommends measuring actual usage before accepting it as a requirement, and the same telemetry that supports that measurement is available for the desktop applications. Organisations frequently discover that assistant use concentrates in a small population doing a small number of tasks, which converts a platform-wide objection into a scoped exception under §16.4.

Local inference changes the calculus for sensitive work. Where the model runs on the endpoint or on infrastructure the organisation controls, document and mail drafting assistance can be offered without content leaving the estate, which is a materially easier conversation with a works council or a regulator than the alternative. This is one of the few places where the open-source position is stronger than the incumbent’s.†

4.7 Security and utilities

Category	Options (steward)	Verdict
Password management	**KeePassXC** (community); **Passbolt** (LU); Vaultwarden (community, AGPL); Proton Pass (CH)	KeePassXC Ready; Passbolt Ready for team credentials
Compression	7-Zip (community); **PeaZIP** (IT); Ark (KDE, DE)	Ready
Desktop search	**Recoll** (FR)	Ready
Feed reading	**FreshRSS** (FR); Tiny Tiny RSS	Ready
Media	**VLC** (VideoLAN, FR); mpv	Ready
Image viewing	nomacs (community); gwenview (KDE)	Ready
Screenshots	Flameshot; Spectacle (KDE); ShareX (Windows only)	Ready
Screen recording	OBS Studio (community)	Ready

KeePassXC is the Swiss study’s only five-star rating, with “keine bekannten” weaknesses recorded [Standtke and Tiede 2024b]. **Passbolt** belongs on their list: Luxembourg-based, purpose-built for shared team credentials, and covering a use case KeePassXC does not.

Note that the Swiss screenshot recommendation, ShareX, is Windows-only, a good illustration of why Stage 1 choices should be filtered for cross-platform availability even when Stage 2 is uncertain. The German study makes exactly this a selection rule: replace “best deployable under Windows” with cross-platform availability as the criterion [Kern et al. 2022]. On Linux the equivalents are Flameshot and Spectacle.

4.8 The consolidated view

Reading the categories against the incumbent estate, four patterns emerge and they determine how much each replacement costs:

Pattern	Examples	Migration cost
Stateless tool	Media player, image viewer, screenshot, archiver	Near zero; install in parallel and stop paying
Own-format tool with export	Mind maps, diagrams, notes	Low; one-time conversion, then done
Shared-format tool	Office suite, PDF	High; fidelity matters because documents circulate
Workflow hub	Mail client, project management	Highest; integrations and habits, more than files

The Swiss WSJF ordering is this table sorted, which is why it works. When a new application comes up for assessment, placing it in one of these four rows gives most of the answer before any feature comparison begins.

5. Stage 2: choosing a distribution

5.1 Or declining to

The German study refuses to name one: "Wir legen uns nicht auf eine konkrete Distribution fest" [Kern et al. 2022]. The reason is procurement law, since specifying a product is only lawful where the requirements can be met by one product alone, so they tender for **support services** and let the distribution follow. Their award criteria transfer regardless of your procurement regime: the contribution to digital sovereignty, integrability into existing technical and staff processes, enterprise support quality, and "Planbarkeit, Verlässlichkeit und die Möglichkeit, technische Anforderungen einzubringen": predictability, reliability, and the ability to get your own technical requirements into the product.

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

The French study also declines to compare distributions, and structures the choice instead: ease of use, desktop environments offered, documentation and community, packaging system, update frequency, and who maintains it [Lecrivain et al. 2022]. It distinguishes commercial from community distributions on the ground that "les choix en matière de technologie ou de marketing ne sont pas fondés sur les mêmes critères" depending on who makes them, and notes that a free edition does not make a distribution non-commercial.

5.2 The options

Option	Steward and control	Notes	Verdict
openSUSE Leap / SUSE Linux Enterprise Desktop	SUSE (DE, Nuremberg)	The European enterprise distribution; YaST as an administration front end	Ready
Ubuntu LTS	Canonical (UK)	Five-year LTS support with direct LTS-to-LTS upgrades; widest hardware and independent-software coverage	Ready
Debian	Debian Project (community, distributed)	No commercial steward; support bought from independent integrators	Ready with a support contract
RHEL / Fedora Workstation	Red Hat / IBM (US)	Strongest identity-management integration through FreeIPA and SSSD	Ready

Option	Steward and control	Notes	Verdict
Linux Mint	Community	Ubuntu-derived, Cinnamon desktop, strong on driver availability	Viable

For an organisation weighting European supply, **SUSE** most directly satisfies the criterion, and it is the option the sources underplay. For an organisation weighting hardware certification and third-party software support, Ubuntu LTS is the pragmatic answer. For an organisation that wants no commercial steward in the loop, Debian plus an independent integrator is a real choice and is the one that most tests §16.4′s capability question in the companion guide.

5.3 What actually decides it

Ranked by how often each is the deciding factor in practice:

1. **Support relationship**, per §5.1. This is the procurement object.
2. **Lifecycle length and upgrade path.** An estate refreshed every four to five years needs a support horizon at least that long, and a proven in-place major upgrade path.
3. **Hardware certification** for the machines you actually buy, per §15.
4. **Fleet-tooling support**, per §7 to §11. The management tool and the distribution constrain each other, and the German study makes this dependency explicit for configuration management.
5. **Third-party software availability**, for the applications that ship binaries.
6. **Staff familiarity**, which is a real cost and is routinely dismissed.

5.4 Image-based systems

Image-based and immutable variants are the direction of travel for managed fleets, because they make the endpoint a reproducible artefact.† The system image is built centrally, shipped whole, and updated atomically with a bootable rollback; applications arrive as containerised or sandboxed packages layered on top.

For a managed estate this is the right model, and it changes §§8 to 10 substantially: provisioning becomes image deployment, configuration drift largely disappears, and patching becomes image promotion. It also demands a build pipeline and a different operational mindset, and it constrains what can be installed locally, which is the point and is also what generates objections.

Evaluate this before committing to a package-based design that will need replacing. An organisation starting a desktop programme in 2026 that builds a traditional package-and-configure estate is building the previous generation.

6. The desktop environment

The German study never discusses this, which is a gap. Its requirements are behavioural: intuitive operation, accessible design considered early, central branding, and centrally-locked security settings that users cannot change while cosmetic settings stay user-adjustable [Kern et al. 2022].

Option	Steward and control	Lockdown mechanism	Notes
KDE Plasma	KDE e.V. (DE)	Kiosk framework, per-key locking	European-stewarded; the same association whose engine work is discussed in §4.3
GNOME	GNOME Foundation (US-registered; development international)	dconf / GSettings lockdown	Widest enterprise documentation

Option	Steward and control	Lockdown mechanism	Notes
Cinnamon, Xfce, others	Community	Varies	Lower administrative tooling maturity

Either of the first two is defensible, and there is now one concrete operational discriminator beyond taste and familiarity. **Plasma** has the stronger central-lockdown story through its kiosk framework, which maps directly onto §9′s policy problem, and it is European-stewarded through KDE e.V. **GNOME** locks down through dconf and GSettings, which works well and is more widely documented in enterprise deployments, and its **remote-desktop implementation currently supports headless and pre-login access where KDE's does not** (§14).

Weigh that against the lockdown advantage according to how your service desk actually works. An estate that recovers broken machines remotely will feel the §14 difference every week; an estate with strong on-site support and demanding policy requirements may reasonably weigh the kiosk framework higher. Otherwise the choice should follow whichever your fleet-management tooling and your support staff already know, because the support cost of an unfamiliar environment exceeds any feature difference.

The French study treats the plurality of environments as a strength over the incumbent, scoring the desktop at maturity 5 and noting that quality depends on the environment chosen [Lecrivain et al. 2022]. It also makes a small point with disproportionate impact: on the default GNOME layout "il est très simple de le déplacer en bas, au même emplacement que celui de Windows". Matching the incumbent's spatial layout removes a category of complaint cheaply, and the instinct to present the new environment on its own terms costs more goodwill than it is worth in the first year.

Wayland is now the default in both environments. The remaining enterprise gaps are in remote support (§14) and some accessibility paths (§19), and both need testing against your requirements; assumption will not do.

7. The managed estate: reference architecture

Migrations are decided here. The Swiss frontend study contains nothing on it, the French main volume contains nothing because that material sits in an annex not in the corpus, and the German study is the only real source. Its treatment is a decision framework, with most product choices deferred.

7.1 The layers

A managed Linux estate needs eight capabilities. Naming them separately keeps each replaceable, which is the German study's design rule: interfaces "so generisch wie möglich" and no hard dependencies between the workplace and the infrastructure [Kern et al. 2022].

Layer	Function	Section
Provisioning	Bare machine to managed endpoint, unattended	§8
Configuration	Declared state enforced continuously	§9
Software	Packaging, distribution, patching	§10
Inventory	What exists, what version, what licence	§11
Identity	Directory join, authentication, smartcard	§12
Data protection	Disk encryption, key escrow, backup	§12.4
Output	Printing and peripherals	§13
Support	Remote access, service desk, field service	§14

The German study names four components that must remain interchangeable for the long term: the distribution, the desktop environment, the standard browser, and the lifecycle-

management front end [Kern et al. 2022]. That list is the practical test of whether the architecture has been kept generic.

7.2 Build it during Stage 1

This delivers the correction promised in §1.1. Four of these layers can and should be built while the estate is still running the incumbent operating system:

- **Inventory** (§11) is a hard precondition for everything and is platform-independent.
- **Packaging and distribution** (§10) can be exercised on the Stage 1 applications, the machinery-on-cheap-cases point of §2.2.
- **Identity** (§12) work on the directory side serves both platforms.
- **Remote support** (§14) tooling should be selected and proven before the pilot needs it.

Deferring all eight to Stage 2 concentrates every unfamiliar technology into the same window as the operating-system change, causing a pilot to fail for reasons unrelated to the desktop.

7.3 Operating processes

The German study defines the process set for a managed model line. It is the part that determines running cost [Kern et al. 2022]:

- **Order management**: a single intake channel for commissioning, decommissioning, moves, and non-standard software.
- **Service desk**, where first-level support is **independent of the endpoint and operating system**, escalating to specialised second and third level. This is the key design decision for a mixed estate: the first line should not need to know which platform the caller is on.
- **Initial setup**: centralised base install before delivery, completed on site without user interaction.
- **Scheduled replacement**, uniform across platforms.
- **Remote support as the standard path**, with field service limited to setup, exchange, and removal.
- **Service coordination** as the interface between the customer and the platform.

7.4 Who runs this

The staffing model is set out in the companion guide's §18.5 and applies to both guides. Three additions are specific to the workstation.

The service desk is retrained and not replaced, and the German study's design decision is the one to copy: first-level support should be independent of the endpoint operating system, escalating to specialised second and third level [Kern et al. 2022]. A desk that must establish which platform the caller is on before it can help doubles its training cost and cuts first-contact resolution in half, and the mixed period will last years.

Field service shrinks and changes shape. With provisioning per §8.1, the technician's role at setup reduces to ensuring network reachability. That is an efficiency and it also means the field capability that used to absorb configuration problems no longer does, so those problems surface at the service desk instead.

Packaging becomes a standing function. §10 makes patch management and software distribution one system, and someone has to own the internal repository, the promotion process, and the delta discipline of §10.3. This frequently has no owner at the start and is the capability whose absence shows up first.

8. Provisioning

8.1 The requirement

The German study sets the right bar: fully automated installation, executable by a third party with no interaction from the central team, with a standardised base image applied before

delivery and final configuration completed on site without user involvement. The field technician's only job is to ensure network reachability [Kern et al. 2022].

Two integration requirements are easy to miss and expensive to retrofit: an interface to the directory so **machine accounts are created automatically** during provisioning, and **certificate automation** so machine certificates for the VPN are issued during the unattended install. The study rates the latter "machbar, aber ggf. zeitaufwendig".

8.2 Tooling

Approach	Tools	Fit
Network install with answer file	**FAI** (DE), Kickstart, AutoYaST, preseed	Mature, flexible, per-machine variability
Image deployment	Disk imaging, or image-based systems per §5.4	Fast, reproducible, less variable
Cloud-init style	cloud-init, ignition	Good where machines are provisioned like instances

FAI is the mature answer in the Debian family and is German-originated. The German study reports successfully testing **AutoYaST** for SUSE Linux Enterprise Desktop, and notes SaltBoot as an alternative within the SUSE management line [Kern et al. 2022]. Both major lifecycle-management products support the installation automation of common distributions natively.

8.3 The golden image

Whether the estate is package-based or image-based, there is a base build, and it should be produced by a pipeline and never by hand. The German study lists the required toolchain: "Git Code Repository, Source Code Review Workflows, Continuous Integration, Software-/Paket-Release-Management" [Kern et al. 2022].

What belongs in the base build: the distribution at a pinned version; the desktop environment with corporate branding; the standard application set; security configuration and the local firewall; identity and certificate configuration; management agents; and the disk-encryption setup of §12.4.

Exclude anything user-specific, anything that changes more often than the image is rebuilt, and anything a group of users needs but most do not. Those arrive through §9 and §10.

The German study's related rule is to deliver branding, disabled desktop functions, and customer defaults through **purpose-built overlay packages**, "um eine Überfrachtung der KM-Konfiguration zu vermeiden" [Kern et al. 2022]. This keeps presentation separate from state enforcement and is a good separation to hold.

8.4 Network considerations

The German study flags a constraint most plans miss: the rollout may not be possible in the network segment where the incumbent management system operates, so provisioning may need separate network segments at each site [Kern et al. 2022]. Where sites are numerous, local provisioning infrastructure is needed, which is the satellite pattern discussed in the companion guide's §13.4 and which the study warns is "sehr kostenintensiv".

9. Configuration management and the policy problem

9.1 Tooling

Highly automated configuration management including remote execution is, in the German study's words, "machbar und im Praxisbetrieb seit Langem erprobt". The candidates are Ansible, Puppet, and Salt, described as of similar value, with the choice explicitly **coupled to the lifecycle-management front end** [Kern et al. 2022].

Layer	Options (steward)	Notes
Lifecycle management	**Foreman/Katello** (community, Red Hat-sponsored); **Uyuni** (SUSE-sponsored, DE); **opsi** (uib GmbH, DE, Mainz); Landscape (Canonical, UK)	opsi is European, AGPL, and manages Windows and Linux clients together, which suits a long mixed period
Configuration	Ansible (Red Hat/IBM, US); Puppet (Perforce, US); Salt (Broadcom, US)	Stewards are American; the tools are broadly supported by independent integrators

The German study's own position: **Foreman** was already in use at their service provider for other projects, and SUSE's management line was already known in their data centre, which is exactly the right basis for a decision at this layer. Existing operational familiarity beats feature comparison.

One caveat the study records and that is easy to forget in a tool evaluation: "Grundsätzlich lassen sich alle Komponenten des Systemmanagements sehr gut ohne LCM-Frontend administrieren." The front end buys usability and reporting, and the underlying capability exists without it.

9.2 Group Policy does not translate

The important finding is the negative one:

> Windows group policies will not map one-to-one to Linux. … benutzer:innenspezifische Gruppenrichtlinien-ACLs werden sich nicht ohne Weiteres im Konfigurationsmanagement abbilden lassen. [Kern et al. 2022]

Group Policy is a per-user, per-machine, hierarchically-scoped, directory-delivered settings system with fine-grained targeting. Linux configuration management is a declarative state system that converges a machine to a described condition. These are different models, and the existing rule set must be **redesigned**. Translation will not work. Organisations that plan a translation discover this late, usually after promising a delivery date.

Policy on a Linux estate is assembled from four mechanisms:

Mechanism	Covers	Scope
dconf / GSettings lockdown, or KDE kiosk	Desktop settings, per-user	User
Configuration management	System state, files, services, packages	Machine
Application policy files	Browser, office suite, individual applications	Machine, sometimes user
Samba group policy client	A subset of directory-delivered settings	Machine

The practical migration path: extract the existing policy set to a readable inventory; classify each setting as security-relevant, functional, or cosmetic; implement the security-relevant ones first and verify them; implement the functional ones as needed; and discard most of the cosmetic ones, which have usually accumulated without review and will not be missed.

The per-user dimension is the hard part. Where a setting must vary by user or group rather than by machine, the mechanisms above are weaker than Group Policy, and the workable designs either reduce per-user variation or accept a login-time mechanism that applies settings from a directory-derived profile.

9.3 Offline behaviour

Configuration management applies when the endpoint reaches the corporate network. Offline, the last transferred settings apply, and the German study specifies that compliance checking continues locally in the background [Kern et al. 2022]. This needs designing, and

cannot be assumed: decide what happens to a machine that has not checked in for weeks, and what the service desk sees when it happens.

10. Software distribution and patching

10.1 A single system

The conceptual shift the German study identifies is that on Linux, **patch management and software distribution are the same system**, unlike on Windows where they are separate [Kern et al. 2022]. Packages come from the distribution's archives or local mirrors, with a small number from third-party or in-house archives. The number supplied from outside the distribution "soll nach Möglichkeit niedrig gehalten werden".

10.2 The quality-assurance layer

Between upstream and the estate there must be a staging mechanism: package mirrors plus a defined promotion process. **Foreman with Katello** and **Uyuni** both provide interfaces for testing updates before release for rollout, which the German study names as a reason to choose either [Kern et al. 2022].

A workable promotion model: an upstream mirror, snapshotted on a schedule; a test channel consumed by IT machines; a pilot channel consumed by volunteers; and a production channel. Security updates need an express path that skips ahead with a shorter soak, and that path should be exercised routinely; an emergency is the wrong place to use it first.

10.3 Keep the delta small

The study's stated top priority for patch management:

> [Es] muss … die Delta … zwischen Distributionspaketen und selbst bereitgestellten Paketen so gering wie möglich [gehalten werden]. Hierfür muss eine Zusammenarbeit mit dem Distributor möglich sein. [Kern et al. 2022]

Any locally-modified package is a permanent maintenance liability and a step toward the accidental fork the study warns about elsewhere. Where a change is needed, the route that ends the cost is upstreaming it, which is the French study's argument for developing with the community: it "assure le maintien de la fonctionnalité dans les versions futures du logiciel" [Lecrivain et al. 2022].

10.4 Application delivery beyond the distribution

The German study identifies a specific risk: a tendered distribution may not supply a needed third-party application, so the estate needs "Verfahren … Komponenten entkoppelt vom Release-Zyklus der Distribution zu installieren und zu aktualisieren, z. B. Bereitstellung in Container-Paketformaten" [Kern et al. 2022].

Mechanism	Best for	Watch
Distribution packages	Everything the distribution ships	Version lag against upstream
Internal repository	Your own packages and overlays	Keep it small, per §10.3
Flatpak	Desktop applications needing newer versions than the distribution ships	Runtime storage; sandbox permissions need policy
Vendor repositories	Commercial applications	Each adds a trust relationship and a supply-chain dependency

Flatpak is the mechanism that resolves the tension between a long-lived stable base and desktop applications that move quickly, and for a managed estate it should be pointed at an internal remote so that what users can install remains a decision you make.

10.5 Self-service

The German study proposes tenant-separated **software catalogues** allowing users to install approved applications themselves, technically possible even with limited connectivity, sub-

ject to security policy [Kern et al. 2022]. A catalogue removes a large class of service-desk tickets and makes the estate's application inventory reflect actual demand rather than what people managed to get approved.

11. Inventory and reporting

Every source makes inventory a precondition. The French study calls it "préalable incontournable à tout projet de migration" and its annex names **OCS Inventory** and **Ansible** [Lecrivain et al. 2022]. **GLPI** (Teclib, France) with its agent is the other widely-deployed European option, and the two are frequently paired. The German study leaves the product open, noting only that all Linux workstations must be inventoried centrally for hardware and software composition, possibly as a component of the lifecycle-management front end.

Schleswig-Holstein's version is the most disciplined: a licence-management system introduced in **2020, before any migration**, yielding per-department needs "correct to the day" and used explicitly to size the migrations that followed [Staatskanzlei Schleswig-Holstein 2024]. The lesson is that inventory, deployed as a planning instrument, paid for itself before the migration started by exposing licences that were paid for and unused.

What the inventory must answer, which is more than most asset registers do:

- What hardware exists, with age and warranty state, per §15.
- What software is installed, at what version, and where it came from.
- What is **actually used**, and by whom, which is the §20.2 question and the one that needs usage data rather than installation data.
- What each machine depends on: peripherals, network shares, printers, and line-of-business applications.
- Which machines carry the exceptions of §16.4.

12. Client identity, smartcards, and encryption

12.1 Directory join

The German study's medium-term design binds Linux clients to the existing Active Directory, with a long-term move to an open-source identity system left in analysis [Kern et al. 2022]. This is the right sequencing: the client platform and the directory are separate migrations, and doing both at once removes the ability to attribute a failure to either.

The mechanism: **SSSD** provides AD-sourced accounts and groups transparently and offline; `libpam-mklocaluser` creates corporate accounts persistently on the endpoint. Credentials and group memberships are cached, so after a first login on the corporate network a user can log in fully offline. On the Windows side, POSIX accounts are supplied through the Identity Management for UNIX role, requiring "insbesondere durch eine LDAP-Schema-Erweiterung".

The client is made fully Kerberos-capable so single sign-on, including in the browser, works as it does on Windows. Users notice the absence of browser-based single sign-on to internal applications immediately.

The study's process advice is sound and easily skipped: provide a **production-like test environment** for identity during development, document Linux-specific changes to it extensively, and agree a roadmap with the infrastructure teams for when the first Linux clients may join the production directory and how the developed procedures transfer [Kern et al. 2022].

12.2 Home directories and profiles

A design decision with large operational consequences. The German study recommends the **endpoint as the primary location for personal work data**, mainly for performance, with background reconciliation to a central location when connected. The stated alternative,

server-side primary storage with an offline copy, “je nach Bandbreite und Latenz der Netzwerkverbindung” degrades the experience [Kern et al. 2022].

Its treatment of group shares should carry over: shares are reachable over SMB or a distributed namespace, protocol-level locking protects concurrent edits, and displaying or changing ACLs from a Linux client against a Windows file-server backend is called out as problematic. **Offline use of group shares is not provided**, because conflicting offline edits cannot be reconciled, and the answer is organisational rules plus user training, with no technical mechanism behind it.

The profile-roaming work the study took from the LiMux engineers is the most transferable engineering detail in the source: profile synchronisation needs a strategy **per application**, all local file attributes must be representable at the sync destination, and login or logout waits must be visualised and fundamentally avoided [Kern et al. 2022].

12.3 Smartcards and eID

Handled through PKCS#11 and PAM. The Swiss frontend study puts the smartcard client out of scope on hardware-dependency grounds, which is a warning and not a solution [Standtke and Tiede 2024b]: where a specific vendor’s middleware is required by the card and reader in use, replacing it “wäre … nur mit unverhältnismäßigem Aufwand möglich”.

Check three things early, because this is a common late blocker: whether the card’s vendor supplies a Linux PKCS#11 module; whether the reader is supported; and whether every consumer of the credential works with it, including the browser, the mail client, the signature workflow of §4.4, and the login path itself.

12.4 Disk encryption and key escrow

Full-disk encryption is a hard requirement for mobile endpoints and “ein Muss der Systemhärtung” [Kern et al. 2022]. The technical stack is LUKS with TPM binding, through systemd-cryptenroll or network-bound unlocking for machines that stay on-site. The German study reports that automated setup with **passphrase escrow into Active Directory combined with TPM protection** “wurde erfolgreich in einem Testprojekt geprüft und bestätigt”, and adds that the endpoint chipset must support secure boot and automatic decryption, with optional strengthening through a second factor such as a token required at start.

Escrow is the part to design, never to assume. A fleet of encrypted laptops with no tested recovery path is a data-loss incident waiting for its trigger, and this is one of the few places in this guide where the failure mode is unrecoverable. The requirements: a recovery key generated at provisioning; stored centrally with access control and audit; retrievable by the service desk through a defined procedure; rotated when used; and **tested**, including the case where the machine will not boot and the case where the escrow system itself is unavailable.

13. Printing and peripherals

CUPS with IPP, as covered in the companion guide’s §11. Three client-side points.

Harden the local print service. The German study warns it “ist je nach Distribution in seiner Standardkonfiguration zu offen konfiguriert” [Kern et al. 2022].

Multifunction devices lose their panel features. Full-feature interfaces come from the manufacturer and “kommen daher im Linux-Kontext eher selten vor”. Printing works; scan-to-destination, accounting, and device-panel workflows frequently do not. Enumerate which of these are used before assuming they must be preserved.

Peripherals generally need an inventory. Beyond printers: signature pads, card readers, specialist scanners, labelling machines, laboratory and industrial instruments, headsets, docking stations, and anything with a vendor driver. Each is a potential exception under

§16.4. This inventory is cheap to produce from §11 and is frequently the source of the surprises that stall a pilot.

14. Remote support and the service desk

The German study admits a straight gap:

> Eine Enterprise-taugliche Fernwartungsoberfläche für Helpdesk-Mitarbeiter:innen fehlt aktuell noch. Hier ist Entwicklungsarbeit nötig, ggf. in Zusammenarbeit mit bestehenden Open-Source-Projekten. [Kern et al. 2022]

The position improved after 2022. Both major desktop environments now expose Wayland sessions over RDP, which closed the protocol gap that made VNC the only option. **They did not close the same gap.**

GNOME Remote Desktop speaks RDP and VNC and supports both user-present screen sharing and **remote login in a headless, pre-login mode integrated with the display manager**. **KDE's KRdp** exposes a Plasma Wayland session over RDP but is built around sharing the **active, logged-in session**, and is not a clean headless or pre-login solution out of the box. Reported trouble through 2024 and 2025 clusters on black screens at RDP login and on portal misconfiguration.

That asymmetry matters more than it looks, because the requirement list below asks whether the tool works *before* login, and that is where the two diverge. An estate whose service desk must recover machines that will not reach a desktop has a materially different experience on the two environments today. The remaining common weakness is the console around the protocol: session brokering, unattended access, consent prompts, audit logging, and ticket-queue integration.

Option	Steward and control	Notes
ThinLinc	Cendio (SE, Linköping)	Commercial, European, mature session brokering
Apache Guacamole	Apache Software Foundation (US)	Browser-based gateway for RDP, VNC, SSH
GNOME Remote Desktop	GNOME project	RDP and VNC; headless and pre-login supported
KDE KRdp	KDE e.V. (DE)	RDP; active-session sharing only
RustDesk	RustDesk (origin CN)	Capable; note the control question
TigerVNC	Community	The baseline the German study assumed

Requirements to specify before choosing: attended and unattended modes; explicit user consent for attended sessions, which is frequently a works-council or data-protection requirement; audit logging of who connected to what and when; file transfer and its restrictions; performance over the slowest link in the estate; and whether the tool works before login, which decides whether it can help with the failures that matter most.

The German study's related design decision is the one to copy: **first-level support should be independent of the endpoint operating system** [Kern et al. 2022]. A service desk that must ask which platform the caller uses before it can help doubles its training cost and cuts its first-contact resolution by half.

Remote support belongs in Stage 1 per §7.2. It determines whether the service desk can support the pilot at all, and discovering its limits during the pilot means the pilot measures the support tooling, not the desktop.

15. Hardware selection and procurement

15.1 The preinstalled licence

The French study records a cost the others miss. Machines arrive with an incumbent licence preinstalled and paid for, so migration wastes it: “Il eût été préférable d’avoir plus simplement des machines nues de tout système d’exploitation”, and the reduced availability of bare machines appears in their synthesis as one of four brakes, “le surcoût matériel” [Lecrivain et al. 2022].

Raise this at contract renewal, well before deployment. Bare or Linux-preloaded configurations exist in most vendors’ business ranges, frequently without appearing in the standard catalogue, and the request has to be made through the account relationship.

15.2 Certification and testing

The German study reports that current Linux distributions install and run on the notebooks in its standard catalogue, that two representative models were successfully tested, and that future catalogue devices will be checked for both platforms [Kern et al. 2022]. That last commitment is the transferable one: **add Linux compatibility to the hardware qualification process**, so that the estate does not acquire machines that cannot run the target platform during the years before migration.

What to test per model: suspend and resume; graphics including external displays and docking; wireless and wired networking; the fingerprint reader and any biometric device; the camera and microphone, including their hardware mute controls; battery reporting and power management; TPM and secure boot, per §12.4; and firmware update capability, which is easy to overlook and which becomes a security obligation.

15.3 The refresh cycle as the migration vehicle

Both French field cases migrated at the pace of hardware renewal, which pairs the change with a new machine and delivers the visible gain §20.5 recommends. The German economic assessment adds the counterpart: most existing hardware can continue to be used, and a Linux estate typically extends the useful life of endpoints that a new incumbent OS release would have retired [Kern et al. 2022].

These two facts are in tension and the resolution is a policy choice. Migrating on refresh is smoother and slower; migrating in place is faster and loses the pairing benefit. A common compromise is to migrate on refresh by default and in place for volunteers.

16. The Windows-only application problem

This is rated probable and critical in the German risk assessment, and it is the workstream most likely to be underestimated [Kern et al. 2022].

16.1 You probably do not know what you run

The German study’s admission is one most organisations will recognise:

> Es liegt keine vollständige Liste der Fachanwendungen vor, die in den verschiedenen Landesbehörden genutzt werden. [Kern et al. 2022]

Procurement of departmental applications sat with departments and not with central IT, so a cataloguing project had to be commissioned before the migration could be planned. Any organisation with devolved departmental purchasing is in the same position and should assume so until proven otherwise. §11 is the instrument; the discovery usually takes longer than expected and should start first.

16.2 The four provisioning routes

The German study defines four, none requiring the application to change:

1. **Desktop virtualisation**, as VDI (a per-user virtual desktop) or terminal services (a shared multi-user server desktop). Complete Windows, so applications run unre-

stricted. For terminal services, application compatibility with a multi-user environment has to be checked.
2. **Local desktop virtualisation**: a hypervisor on the endpoint itself. Differs from VDI in management and resource consumption, and not in user experience.
3. **Published applications**: only the application, with "kein sichtbarer Unterschied zu einer lokal installierten Fachanwendung" and links between applications preserved.
4. **Wine** as a compatibility layer, with the caveat that it is not a native Windows environment and the effects "lässt sich nur im Einzelfall überprüfen".

A fifth, which the German study treats as the destination and not a route, is to **replace the application with a web-delivered one**, either by the vendor's own roadmap or by procurement pressure.

The study also adds a user-experience requirement that is easy to miss: no **Medienbruch**. Even a virtualised application must appear embedded in the desktop. A window into another computer is the Medienbruch the requirement forbids.

The policy bound on all four: they apply to line-of-business applications only, and strategically they "können … nur kurzfristige Lösungen sein". With one important exception: for highly specific applications, alternative provisioning is "keine Zwischenlösung …, sondern die Standardvorgehensweise" [Kern et al. 2022]. A permanent terminal server for a handful of specialist tools is a legitimate architecture.

Options include Apache Guacamole and Kasm Workspaces (both US-stewarded), ThinLinc (Swedish), and CrossOver from CodeWeavers (US) for the Wine route.

16.3 The remediation decision

The French study supplies the decision framework the German one lacks, comparing four remediation routes [Lecrivain et al. 2022]:

Route	Effort	Schedule control	Ecosystem linkage
Ask the community	Low	**None**	Weak
Ask the vendor	Low	Moderate, often paid	None
Develop internally	High	**Best**	None
Develop with the community	High	Moderate	**Strong**

Their guidance: asking the community is "une stratégie attentiste" suitable only for non-critical functions or where remediation can wait. Asking the vendor works where a support relationship already exists. Internal development gives the best schedule control and carries a hidden cost, since long-run maintenance falls entirely on you, which "peut remettre en cause l'intérêt de la fonction sur Linux"; choose it only where neither community nor vendor wants the feature. Development with the community costs more in steering overhead and carries the decisive benefit: "cela assure le maintien de la fonctionnalité dans les versions futures du logiciel."

The study is explicit that the arbitration should **not** be made at the moment a problem is discovered, that it may follow an attempted remediation, and that a failed attempt can send the decision back to not remediating at all.

It also insists the global strategy be decided and communicated in advance: "Cette décision clé est prise et communiquée bien en amont, elle donnera un état d'esprit au projet." Teams that know whether the organisation prefers to fund upstream work or to build internally can position themselves without asking each time.

16.4 Exceptions, and their governance

The alternative to remediating is to keep the incumbent platform for that function, or to drop the function. The French study is clear that keeping Windows where it is needed is the intended outcome and not a defeat: "L'objectif n'étant pas d'avoir un parc épuré de tout Windows, mais de faire en sorte de ne le maintenir que là où il est nécessaire."

It attaches a governance requirement that experience justifies: the rules granting an exception must be **formalised, shared, and made transparent**, "afin d'éviter toute impression de traitement de faveur et même mieux : les empêcher" [Lecrivain et al. 2022]. Exceptions granted case by case in private become a status good, and the programme loses control of its own scope.

A workable exception register records, per exception: the function (never the application); the population; the justification; the provisioning route from §16.2; the review date; and what would have to change for the exception to end. Reviewing the register on a schedule stops the residual estate from becoming permanent by default.

16.5 The strategic ordering

Web delivery first, then terminal services, then local virtualisation, then compatibility layers, then keeping the incumbent platform.

Both the French and German sources converge on web delivery as the real answer. The French call web technology "un catalyseur clé" and recommend steering development toward web or thin clients and making applications modular on open standards, with the explicit purpose of "réduire les coûts/barrières à la sortie d'une ou de plusieurs solutions" [Lecrivain et al. 2022]. The German study turns it into procurement policy: require cross-platform or platform-independent development in future tenders, require import and export of open document formats, and prefer that new departmental applications be developed as open source under a suitable licence [Kern et al. 2022].

That procurement rule is the most effective item in this section, and it costs nothing to adopt today. Every application procured under it is one that will not need remediating later, and the benefit accrues across every future platform decision, and not only this one.

17. Client data migration

The companion guide's §14 covers server-side data. Data on the endpoint is a separate problem and is routinely forgotten until users notice.

17.1 What is on the machine

- **Documents** in local folders, including the ones outside any synchronised location.
- **Mail archives** in local files, frequently large and frequently the user's only copy.
- **Browser profiles**: bookmarks, saved passwords, certificates, extensions.
- **Application settings and customisation**: toolbars, macros, templates, dictionaries, signatures.
- **Certificates and keys** in local stores, some of which cannot be exported by design.
- **Peripherals' local configuration**, per §13.
- **Whatever the user considers essential**, which is discoverable only by asking.

17.2 The approach

Move personal data to a synchronised location during Stage 1. This is the German study's preparatory measure applied to user data: provide the platform-independent store on the incumbent platform first, so that at the moment of the operating-system change the data is already in a location the new system can reach [Kern et al. 2022]. It converts a per-machine migration into a configuration change.

Mail archives need a campaign of their own. They are large, they are local, and their owners frequently do not know they exist. Discovery through §11, then either import to the server or convert to a documented format and store centrally.

Browser profiles migrate through the browser, provided the standard browser was chosen during Stage 1 per §4.3, which is one more reason for that sequencing. Saved passwords should be migrated into the password manager of §4.7, which turns a data-migration task into a security improvement.

Application customisation mostly does not migrate, so the plan is to inventory what exists, rebuild what matters centrally (templates, dictionaries, signatures), and let the rest go with warning.

17.3 The pre-migration conversation
The highest-value fifteen minutes in a desktop migration is asking each user, before their machine changes, what they have locally that they cannot lose. It surfaces the unsuspected archive, the macro that runs a monthly report, and the certificate that signs a regulatory submission. It also converts the migration from something done to the user into something done with them, which is worth as much as the data it saves.

18. Security baseline

18.1 The baseline itself
Adopt a published hardening baseline and record deviations, as the German study did by deriving requirements from the national baseline-protection catalogue alongside the existing platform's requirements [Kern et al. 2022]. Writing one from scratch produces a weaker result and cannot be audited against anything.

Elements the sources name explicitly: a **centrally configurable local firewall**, "voraussichtlich die Standard-Firewall der verwendeten Linux-Distribution"; standardised security settings at system and user level that **users cannot circumvent**; role models for users and groups mapping application and filesystem rights; secure boot and TPM support in the hardware; full-disk encryption per §12.4; and remote shell access restricted to the internal network and IT support [Kern et al. 2022].

18.2 Users without administrative rights
The estate should run with users lacking administrative rights, with a defined elevation path for the cases that need it. This is the single largest security difference between a managed and an unmanaged Linux estate, and it is easier to establish at the start than to impose later.

18.3 Endpoint protection
The sources are thin here. The German study lists a virus scanner as a category of base software without evaluating any product, and mentions a web proxy with virus protection as a network-side control [Kern et al. 2022].

The position to hold: the endpoint-protection market for Linux desktops is thinner than for Windows, the threat model is different, and the compensating controls (no administrative rights, packages from signed repositories, application sandboxing, and central configuration enforcement) carry more weight than they do on the incumbent platform. Where a regulator or an insurer requires an endpoint agent, verify Linux desktop support specifically, since server support is more common and is not the same product.

18.4 Telemetry
The German study records a requirement that has gained importance and should be stated as a design goal: transmission of user, device, and licence information to manufacturers must be **limited to a necessary minimum** [Kern et al. 2022]. This is straightforwardly easier to achieve on the target platform and is a benefit to include in §20.5′s list of gains.

18.5 The comparison that will be made
The general form of this argument, and the analytic move that resolves it, is in the companion guide's §17.7: the security objection is about **who operates the service** and not about which software runs, so it is answered by the hosting and operating decision.

Three points are specific to the endpoint.

The endpoint threat model differs. The dominant commodity endpoint threats target the incumbent platform, because the users are there. A smaller installed base is not a security

property to rely on, and it is a real reduction in exposure to untargeted attacks. State this accurately in both directions: it helps against commodity threats and not against a targeted adversary.

The compensating controls carry more weight here. Users without administrative rights (§18.2), packages from signed repositories through a controlled pipeline (§10), application sandboxing, and centrally enforced configuration (§9) together cover much of what an endpoint agent is bought to cover. That argument is stronger than the claim that endpoint protection is unnecessary.

The gap is in tooling maturity. §18.3 records that the endpoint-protection market here is thinner. Where a regulator or an insurer requires a specific control, verify the product covers the desktop platform specifically, and treat a gap as a scoping constraint on the migration, and never something to discover at audit.

19. Accessibility and regulatory constraints

The sources are weak here and the weakness should be corrected, and never transposed.

The German study mentions accessibility four times, names no standard, assistive technology, testing method, or acceptance gate [Kern et al. 2022]. Notably it uses better document accessibility as an argument to *sell* the migration to staff, which is fair, while never treating accessibility as a requirement the migration must meet. Schleswig-Holstein does better, committing to accessible-PDF production through an assistant it initiated into LibreOffice [Staatskanzlei Schleswig-Holstein 2024]. The French study devotes an annex to it, which is not in the corpus.

For a 2026 migration this is not optional. The **European Accessibility Act** has been in force since **28 June 2025**, and **EN 301 549**, currently version **3.2.1**, is the harmonised standard against which digital accessibility is assessed. Conformity with it confers a **presumption of conformity** with the Act, and that presumption makes it the practical target: demonstrating conformance to the standard is a tractable engineering task, whereas arguing accessibility from first principles to a regulator is not. Any workstation migration in scope has to demonstrate that the replacement stack meets at least the bar the incumbent met.

Four practical consequences:

Test the assistive-technology path before the pilot. Screen-reader support on Linux runs through Orca and the AT-SPI stack, and coverage varies by application and by desktop environment.† Users of assistive technology belong **in** the pilot and must not be the last population migrated, because discovering a blocker after the estate has moved leaves no options.

Test the whole chain. Accessibility fails where components meet: the login screen, the disk-encryption passphrase prompt, the remote-support session, the browser plus the web application, and the office suite plus the document. Each is a separate test.

Validate document output. veraPDF (§4.4) checks PDF/A and PDF/UA conformance, which is the machine-checkable part of the obligation. Accessible document *production* also needs training, because the tools cannot make an unstructured document accessible.

Make accessibility an award criterion. The German study's tender criteria do not include it, an omission to correct.

Adjacent regimes to scope with someone competent, since assumption is the failure mode: **NIS2** for security obligations on the estate, and the **Cyber Resilience Act**, whose **reporting obligations apply from 11 September 2026** and whose remaining obligations apply from **11 December 2027**. The Act's treatment of open-source stewards (Article 24, with the fine exemption in Article 64(10)) is set out in the companion guide's §17.6. It matters here because §10.3′s rule about keeping your package delta small, and §16.3′s route of developing with the community, both affect whether an organisation is distributing products with digital elements in its own right.

20. Change management

20.1 What the evidence actually is

One source in the corpus has field evidence: the French study, from two cases, both post-migration, both interviewed with the same instrument [Lecrivain et al. 2022]. The authors temper their own result, noting that two actors observed after the fact "n'expriment donc pas le risque et la charge à laquelle faire face". It is nonetheless the only observational data available and should be preferred to the other sources' anticipations where they conflict.

Their headline finding, stated before the method: both cases report "une très forte satisfaction vis-à-vis de Linux Poste de Travail sur les différents périmètres fonctionnels", and both retained Windows machines where needed, decided "à l'aune du pragmatisme".

20.2 Inventory by function

Knowing an application is installed tells you nothing about what it is used for. Decompose the user population by function: most users of a spreadsheet need arithmetic, some need real spreadsheet features, and a few maintain "microsystèmes d'information non régulés" built from macros and interlinked files [Lecrivain et al. 2022].

Reasoning by application keeps the incumbent for the third group. Reasoning by function observes that the third group's need "mérite un autre outil qu'un tableur" and that the organisation might build it. The macro estate and the departmental database estate are governance problems the migration surfaces; treating them as reasons to stop preserves both the dependency and the problem.

The French study's criticality criteria for this exercise are usable directly: the size of the user population, adherence to business operations, the importance of the service rendered, and regulatory-compliance impact. Their arbitration matrix then plots criticality against maturity with thresholds at 3 on a 5-point scale, giving four zones. The study states the decision rule: if most areas fall in the support zone, migration can be approached with confidence; if most fall in the risk zone, a complete migration would be "très audacieux" [Lecrivain et al. 2022].

20.3 The triptych, and who is in the pilot

Scopes pass through **pilot, remediation, deployment**, with remediation starting during the pilot and continuing after it. Scopes are defined by function and sequenced simplest-first, which trains the IT department on easy cases [Lecrivain et al. 2022]. Areas of differing sensitivity get different remediation lengths, so a simple scope and a hard one starting together will deploy far apart, and the study is explicit that its schematic "n'a pas vocation à donner une durée précise".

Recruit enthusiasts **and the most reluctant users**. The reluctant are interviewed for the reasons behind their reluctance, and their machines are left alone. Real gaps surface early enough to remediate; unfounded fears become the communication material for everyone else. Both groups must be influential, and the reluctant must also be technically expert so their objections track demanding usage. Anchoring the pilot on "deux profils aux extrémités opposées du spectre de confiance" maximises the chance of having met every remediable case before deployment.

20.4 Training by relay

Three tiers: a small group of experts, a larger group of relays, and everyone else. Experts and relays are recruited from pilot participants by influence. One intensive training for experts, one standard training for relays, and the relays carry it to their colleagues. Formal training volume stays low because habituation does most of the work [Lecrivain et al. 2022].

The ratio guidance in the source is **ordinal only**: experts much fewer than relays, relays much fewer than users. No numbers are given.

The group needing the most training is the IT department, for the reason in §20.6, and the French answer is the same triptych applied to them: continuous learning by discovery

during pilots and remediations, so that “c’est donc l’effet du temps plus que l’intensité des formations qui joueront”.

The German study adds a procurement warning that has a long lead time: suitable training offerings may simply not exist in the local language, tailored programmes and certification paths are available from few providers, and requiring a full programme in a particular language eliminates more of them [Kern et al. 2022]. Certification paths also matter for recruitment, since they let candidates demonstrate knowledge.

20.5 Communication, and the counter-intuitive part

The French field cases used “une communication non agressive” with “globalement peu de « bruit »”. Only two things were formally announced: calls for volunteers, and the decision that the new platform had become the default. There was no announcement of the migration as a project, “afin de ne pas effrayer les utilisateurs” [Lecrivain et al. 2022].

Instead, produce benefits and communicate those. Pair the migration with hardware renewal and with new services that are easier on the new platform. The study is explicit that magnitude matters less than existence: “Ces avantages peuvent donc être aussi importants qu’anecdotiques du moment qu’ils inscrivent dans l’esprit des utilisateurs un gain”, and its examples are modest, such as managing a phone from the desktop or a new shared team space. Schleswig-Holstein arrives at the same tactic independently, selling its groupware change on larger mailboxes and easy phone access [Staatskanzlei Schleswig-Holstein 2024].

This runs against normal programme communication practice, and the reasoning is sound: announcing a migration invites everyone to form a position on it before they have any experience of it.

20.6 The resistance is internal

The finding that most contradicts the German study’s risk model:

> La principale difficulté mise en avant dans les deux cas 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. [Lecrivain et al. 2022]

The cause is a loss of the ability to exploit expertise rooted in the incumbent ecosystem, which is a rational response to a personal cost. The remedies follow from the cause: migrate gradually so teams learn as they go, give them target-like environments to test in, invest in certification paths so the new expertise is portable and visible, and, where it persists, decide from the top. The French study is blunt that this last step is sometimes needed: “il faudra parfois imposer, par la direction, la nouvelle norme”.

The German study reaches the personnel question from the organisational side and adds two things to plan for: role and job descriptions have to be re-examined and rewritten, and the transition from project to line organisation needs preparing early, because the moment of highest personnel demand arrives when the platform ships in volume [Kern et al. 2022].

20.7 Sponsorship

The French study requires “l’adoubement, voire le parrainage des directions”, and gives a specific reason beyond the usual one: executive commitment is needed “pour permettre de faire face aux pressions à venir des fournisseurs perdant leur hégémonie sur un client” [Lecrivain et al. 2022]. Vendor pressure appears in their synthesis as one of four brakes, and §23.1 shows what it looks like when it succeeds.

20.8 Deadlines

The French study argues against progress indicators and completion targets, recommending “d’éviter les indicateurs d’avancement et la logique de délai” and reframing the goal so the new platform “devienne le système par défaut dans l’esprit des personnes”. Their cases ran four to seven years. The end state is described and not measured: “Un parc où Linux est installé partout où il est réaliste qu’il soit” [Lecrivain et al. 2022].

As in the companion guide's §20.6, this guide adopts the reasoning and adjusts the conclusion. A seats-migrated target creates pressure to migrate users who are not ready, which produces the visible failures that end programmes. An organisation accountable to a board still needs indicators. Measure readiness:

Indicator	Why it leads
Applications reachable through a browser or standard protocol	The §16.5 enabler; reduces coupling to any platform
Documents created in open formats	Bounds the conversion problem
Endpoints under automated configuration management	Works on either platform; the §7.2 preparation
Exceptions in the register with a review date	Converts unknown scope into managed scope
Remediation items closed against opened	The only measure of convergence
Scopes through pilot	Progress that requires no seats to migrate
Service-desk tickets per hundred migrated seats	The one number that tells you to slow down

Migration then proceeds at the pace of hardware renewal, per §15.3.

20.9 Programme shape and indicative durations

§20.8 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.

What should not have a deadline: the proportion of endpoints migrated. What must be planned: the sequence, the capability build, and the exception register's review cycle.

Stage 1, twelve to twenty-four months, and it can start now. Inventory by function (§11, §20.2); the incumbent's fonts (§4.1); the stateless tools at the top of the priority list (§2.1); the standard browser, chosen and tested on the incumbent platform (§4.3); the office suite in parallel with the incumbent, with new documents in an open format (§4.1); personal data moved to a synchronised location (§17.2); and the procurement rule requiring platform independence in new applications (§16.5). None of this requires an operating-system decision, and the last item shrinks Stage 2 permanently at no cost.

Stage 1b, running concurrently from month six. The four management layers that serve both platforms: inventory, packaging and distribution, client identity, and remote support (§7.2). Building these during Stage 1 is essential, because deferring them concentrates every unfamiliar technology into the same window as the operating-system change.

Stage 2, twenty-four to forty-eight months. The remaining management layers (provisioning, configuration and policy, encryption with tested escrow); then one function-defined pilot with its written exit criteria (§23.1); then deployment in waves at the pace of hardware renewal (§15.3). The line-of-business remediation of §16 runs throughout and is the workstream most likely to determine the actual end date.

Total: three to six years to a majority-migrated estate, against four to seven observed in the two documented field cases [Lecrivain et al. 2022]. A residual incumbent population is the intended outcome (§16.4).

What compresses it: a homogeneous application portfolio; web-delivered line-of-business applications; hardware already qualified for both platforms (§15.2); and an IT department that wants this. What extends it: professional graphics or engineering populations (§4.5); regulated signature workflows (§4.4); a large unknown departmental application estate (§16.1); smartcard middleware without Linux support (§12.3); and works-council negotiation on remote support and telemetry, which has a long lead time and is easily forgotten.

The dependency that most often breaks a plan is that **the pilot cannot run before remote support works** (§14). A pilot supported by people who cannot see the screen measures the support tooling.

20.10 Co-determination and staff consultation

This binds harder on the workstation than on the back end, because staff work at the workstation and several of the required capabilities are monitoring-adjacent. The general treatment is in the companion guide's §20.9; what follows is specific.

Remote support is the item that most often stalls a pilot. A tool permitting observation of a workstation will generally require agreement, and §14 lists the properties that agreement usually turns on: explicit consent for attended sessions, a visible indicator while connected, audit logging of who connected to what and when, and a documented restriction on file transfer. Specify these as requirements when selecting the tool, because retrofitting consent behaviour to a tool that lacks it is not possible.

Inventory and management agents report what staff have and use (§11). The scope of collection, its retention, and whether it can be used for individual performance assessment are the negotiable points.

Telemetry cuts the other way. §18.4 records the requirement that transmission of user and device information to manufacturers be limited to a necessary minimum, and that is materially easier to achieve and to demonstrate on the target platform. Where an existing works agreement constrains what may leave the organisation, this migration may relieve a constraint and not create one.

Lead time. Begin when the pilot is designed. It is measured in months and sits on the critical path.

21. Costs

The workstation case has a different shape from the back-end case, and the difference is the whole finding.

The Swiss frontend study's entire economic chapter reduces to this: every recommended application is FLOSS, so there are no licence costs; support from a local provider costs money if wanted; and because nothing must be bought, applications can be installed in parallel and a multi-product strategy is available [Standtke and Tiede 2024b]. No figures are given because there are none to give.

`verify/cost_model.py` accordingly produces a workstation table with a zero subscription line:

Seats	Subscriptions/yr	One-off days	3-yr total	Per seat/yr
250	0	165	273,000	364
2,500	0	432	1,268,400	169
25,000	0	1,131	8,857,200	118

In EUR, on placeholder effort, day-rate, and productivity-loss parameters. The numbers demonstrate a method; substitute your own.

One observation holds at any parameter values, and one holds only above a threshold.

Effort and disruption account for everything. This is structural: the subscription line is zero whatever the parameters, because none of the recommended applications charges one. So the estimate is only as good as the effort model, and the effort model is the organisation's own.

Productivity loss dominates at scale, above a threshold this guide can state exactly. Because the model is linear, the crossing point is available in closed form rather than by simulation: disruption exceeds technical effort when lost user-days per person exceed the technical effort bill divided by the seat count times the loaded cost of a user-day.

Seats	Technical effort (days)	Threshold (lost user-days per person)
250	165	2.64

Seats	Technical effort (days)	Threshold (lost user-days per person)
2,500	432	0.69
25,000	1,131	0.18

At the placeholder loss of one user-day per person, disruption dominates at 2,500 and 25,000 seats and effort dominates at 250. The earlier editions of this guide said "productivity loss dominates at scale" without the threshold, which was true at the assumed values and unstated as a condition; the condition is the table above, and "at scale" is doing real work in that sentence. Over the parameter ranges declared in `verify/cost_model.py`, disruption exceeds effort on 98.8 percent of sampled draws at 25,000 seats and on 15.1 percent at 250.

Where the threshold is cleared, disruption is the term to attack, and it is attacked by parallel installation, by relay training, by pairing with hardware renewal, and by moving the office suite late. The recommendations in §20 are, read economically, an attack on this term, and that is the organising idea of the workstation business case at 2,500 seats and above. Below it, the case is about the effort estimate, and §20′s change management is a quality argument instead of an economic one.

21.1 The break-even comparison

Where this frame comes from. The economics here are imported rather than derived, and these are the sources. Switching costs are Klemperer (1995) and the survey in Farrell and Klemperer (2007): an incumbent can price above the competitive level up to the buyer's cost of leaving, which is why this guide treats exit cost as the object and the licence line as a symptom. Shapiro and Varian (1998) is the practitioner statement. The break-even inversion below, and its discounting, sit in the irreversible-investment tradition of Dixit and Pindyck (1994). The companion guide's §2 develops the frame at length; this section states only what §21 uses.

Because there is no subscription line, the comparison is unusually clean: the whole cost is one-off, so the question is simply what recurring incumbent cost it displaces. The model derives the incumbent per-seat figure at which the migration breaks even, which needs no assumption about your agreement.

Seats	Horizon	Flat incumbent price	Price rising 8.7%/yr
250	3 yr	364	334
250	5 yr	218	184
250	10 yr	109	73
2,500	3 yr	169	155
2,500	5 yr	101	85
2,500	10 yr	51	34
25,000	3 yr	118	108
25,000	5 yr	71	60
25,000	10 yr	35	24

EUR per seat per year, undiscounted, on placeholder effort parameters, inheriting every limitation in §21.

Which escalation rate. The 8.7 percent above is a *price* series: the measured annual growth of prices on Europe's locked cloud-and-software base [Asterès 2026]. Schleswig-Holstein's fourfold-in-ten-years implies 14.9 percent, and that is an *expenditure* series, compounding unit price with seat growth and scope expansion. Earlier editions of this guide carried the expenditure figure as a price escalation, which overstated the migration case in the direction its author would prefer. The tables use the price series and report the expenditure series as an upper bound; substitute your own rate from your invoice history [Staatskanzlei Schleswig-Holstein 2024].

Discounting. The table above is undiscounted, which flatters the migration: it pays at the start and the incumbent pays across the horizon. At ten years the choice of rate is material.

Seats	Incumbent price path	r = 0%	r = 4%	r = 8%
250	flat	109	129	151
250	price +8.7%	73	89	106
250	expenditure +14.9%	54	67	81
2,500	flat	51	60	70
2,500	price +8.7%	34	41	49
2,500	expenditure +14.9%	25	31	38
25,000	flat	35	42	49
25,000	price +8.7%	24	29	34
25,000	expenditure +14.9%	18	22	26

Ten-year break-even in EUR per seat per year. At eight percent the ten-year figure is roughly forty percent above its undiscounted value, which is larger than the effect of the escalation choice and larger than most readers expect. Use your organisation's own hurdle rate.

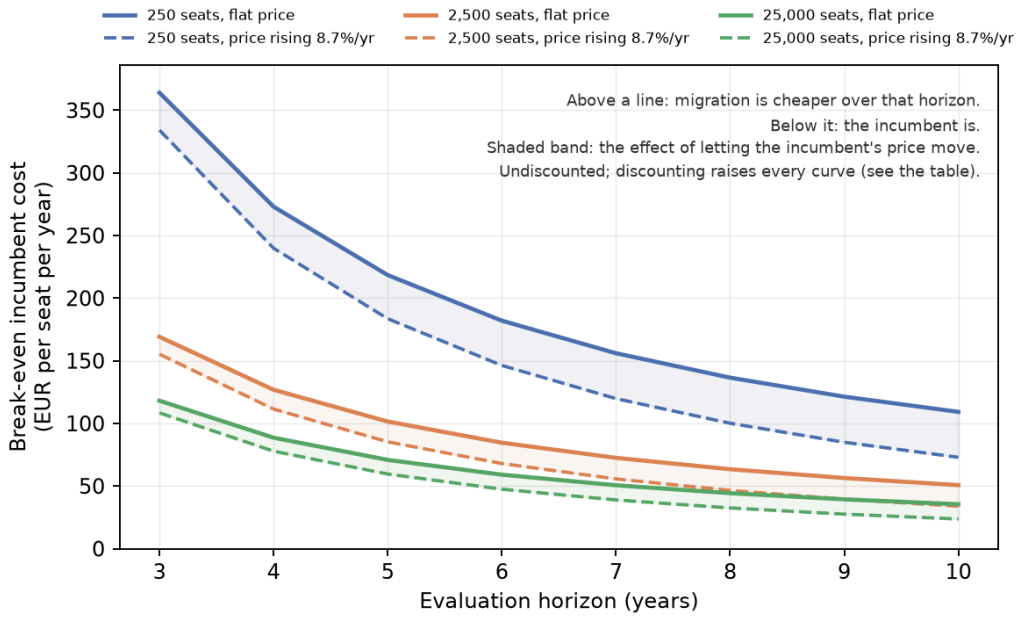

Figure 1. Break-even incumbent cost against the horizon evaluated, for the three estate profiles. Solid lines hold the incumbent's price flat; dashed lines let it rise at the measured 8.7 percent for a locked base. Both families are undiscounted and therefore sit below their discounted equivalents in the table above. Generated by `verify/plot_breakeven.py`.

The figures are lower than the back-end equivalents in the companion guide's §21.5, because there is nothing recurring to pay. They do not capture the part that decides real cases: the residual incumbent estate of §16.4 continues to carry licences, and the alternative-provisioning platform that serves it is a recurring cost of its own. An organisation that migrates 80 percent of endpoints does not remove 80 percent of its client licence cost, and the business case should model the residual explicitly.

21.2 What the model does not carry

- **Fleet-management tooling and its infrastructure** (§§7 to 11), including the site infrastructure the German study warns is "sehr kostenintensiv" where the estate is dispersed.

- **The alternative-provisioning platform** for §16: VDI or terminal-server capacity, which for a large residual estate is a substantial recurring cost and may carry its own incumbent licences.
- **Remediation** (§16.3), whose scale is unknown until §11 and §20.2 are done.
- **Retained incumbent licences** for the exception population of §16.4.
- **Hardware**, offset partly by the extended endpoint life the German study notes and burdened partly by the wasted preinstalled licence the French study records.
- **Training and certification** (§20.4), including the market-availability problem.
- **Accessibility remediation** (§19), which the German study lists as a possible additional cost line without quantifying it.

21.3 What the sources support

The German study's conclusion stands: operation is "langfristig ähnlich wirtschaftlich" as the incumbent, with "Einsparungen … erst nach mehreren Jahren zu erwarten", licence savings reallocated to support, and no experience values available to size the shift [Kern et al. 2022]. The French study contains no cost figures at all, and records that cost reduction "n'est pas nécessairement l'enjeu initial, mais peut résulter d'un constat empirique" after the fact.

Any workstation business case promising savings in the first three years is not supported by anything in this corpus.

22. Risks

The German ALARP ratings, with this guide's adjustments [Kern et al. 2022]:

Complexity: probable, critical. Driven by a shortage of mixed-platform expertise, causing dependencies to be missed and effort misjudged. The named example is small and telling: replacing macros in one office suite with macros in another.

Line-of-business applications: probable, critical. §16. The sub-risk to plan for is that misjudging one application's port delays one department, while misjudging the dependency between ports and the platform's introduction delays everything.

Distribution release cycles: occasional, minor. A tendered distribution may give little flexibility over product cycles, and required third-party software may not be obtainable through it. Mitigation in §10.4.

Community fragmentation: probable, minor. Cooperation with upstream projects is a capability the organisation may not have; the German study notes strong fragmentation among Linux communities and that many projects have no commercial partner behind them. Their answer is to create a named role for it, an open-source community management function attached to the programme.

Own-adaptation operating cost: occasional, minor. Maintaining local modifications may cost more than expected, with "die Gefahr des 'ungewollten' Erstellens eines 'Forks'". Mitigations: fund development across organisations, cooperate closely upstream, and avoid many individual local solutions. This is §10.3 stated as a risk.

User acceptance: occasional, minor. Downgraded relative to §20.6′s finding that the binding constraint is internal IT resistance, which the German study does not rate at all. Their own analysis of acceptance is nonetheless useful: open source carries an image problem, since "kostenlose" products are assumed unable to compete, and smaller marketing budgets make projects look less mature.

Strategic abandonment: remotely conceivable, catastrophic. The only catastrophic entry. Their mitigation is structural: build the infrastructure to be platform-independent and to serve more than one client platform, so a change of direction does not strand the investment. "Die Entwicklung einer nur auf Linux-Arbeitsplätze ausgerichteten Infrastruktur im Rahmen dieses Projekts wäre somit zu vermeiden."

22.1 Munich

Munich's LiMux programme migrated roughly 15,000 desktops by 2013 and produced around 18,000 LibreOffice templates. In November 2017 the city council voted to move back to Microsoft clients [Edge 2017]. The case is treated at length in the companion guide's §22.1; the summary matters here because it is the most-cited desktop migration in existence and it is usually cited wrongly.

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". 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; 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 instructive: the city's IT committee estimated €20m of savings, while a Microsoft-funded HP study claimed a €43m increase and was never published [Edge 2017].

The corroboration sits in the German feasibility study, which had no reason to be kind. Its only reference to Munich is a technical exchange with former LiMux engineers, from which it took the profile-roaming design of §12.2, described as "erarbeitet und erfolgreich erprobt" [Kern et al. 2022]. A programme whose engineering the next serious feasibility study borrows from was not failing technically.

The lesson is the one the German risk matrix encodes: the existential risk is political and commercial, the incumbent is an active participant in it, and the mitigations are governance mitigations. Durable sponsorship per §20.7, a platform-independent architecture per §7.1, and scepticism about vendor-funded evaluations.

23. Pilot design and what to do first

23.1 The pilot

The French triptych (§20.3) leaves the pilot's design implicit. Made explicit:

Population. One function-defined scope, per §20.2. It mixes enthusiasts and refractories, both influential, the refractories technically expert. Include at least one user of assistive technology (§19) and at least one holder of a §16.4 exception, so both are surfaced early.

Duration. Long enough to cross a monthly and a quarterly cycle, because the reports that run monthly and the processes that run quarterly are the macro estate's habitat.

What is measured. Service-desk tickets per seat and their categories; remediation items opened and closed; time to first productive hour; and the specific complaints, with a reproducible artefact attached to each. Complaints without an artefact are noise.

What is not measured. Satisfaction in the abstract, and anything that invites a verdict on the platform.

Exit criteria, written before it starts. The remediation backlog is converging, ticket volume per seat has returned to a defined multiple of baseline, and no unresolved item is rated blocking. Without written criteria the pilot ends when someone loses patience.

The IT department goes first, because §20.6 identifies them as the binding constraint and §20.4 needs the experts recruited from somewhere.

23.2 The order

1. **Inventory by function**, and separately catalogue what departments bought without telling you. §11, §16.1, §20.2.
2. **Install the incumbent's fonts** wherever the office suite will run. The cheapest quality improvement available. §4.1.

3. **Deploy the top of the WSJF list**: media player, password manager, archiver, PDF viewer, image viewer, desktop search. Stateless, parallel-installable, and they build your packaging pipeline. §2.1.
4. **Adopt the procurement rule** requiring platform independence and open formats in new applications. Costs nothing today and shrinks §16 permanently. §16.5.
5. **Choose and deploy the standard browser** on the incumbent OS, and test internal web applications against it there. §4.3.
6. **Deploy the office suite on the incumbent OS**, switch new documents to an open format, leave the archive alone, and start the template project. §4.1.
7. **Move personal data to a synchronised location** while still on the incumbent platform. §17.2.
8. **Build the fleet-management stack during Stage 1** and prove it on the incumbent estate where it applies: inventory, packaging, identity, remote support. §7.2.
9. **Qualify hardware for both platforms** at the next procurement round, and ask for bare machines. §15.
10. **Prove encryption and key escrow**, including the recovery path, before any encrypted machine leaves the building. §12.4.
11. **Scope the accessibility path** and put assistive-technology users in the pilot, never at the end. §19.
12. **Run one function-defined pilot** with enthusiasts and refractories, remediate, then deploy. §23.1.
13. **Move the mail client last.** §4.2.

23.3 When to stop

The general treatment, including why the criteria must be pre-committed, is in the companion guide's §20.8. The workstation-specific criteria:

- **The exception register grows instead of shrinking** across review cycles (§16.4). This is the clearest signal that the §20.2 inventory understated the line-of-business estate, and it is the most common reason a desktop programme stalls.
- **Service-desk volume per hundred migrated seats does not trend toward baseline** after successive waves. §20.8 makes this an indicator; here it is a gate.
- **A blocking peripheral or smartcard dependency** turns out to affect a much larger population than scoped (§12.3, §13).
- **Hardware qualification fails** for the machines the organisation can actually buy (§15.2).
- **The pilot cannot be supported** because remote support does not meet the consent and audit requirements of §20.10, and no alternative is procurable.

The off-ramp is unusually clean on the workstation. State this at approval: because Stage 1 is entirely parallel-installable and platform-independent (§2.2), stopping before Stage 2 leaves the organisation with open formats, portable applications, and a working management layer, and costs nothing that was going to pay back later. Stopping *during* Stage 2 leaves a mixed estate, which is the intended end state anyway (§16.4).

Glossary

AT-SPI. The accessibility toolkit interface on Linux, through which assistive technologies read application state. §19.

Dconf / GSettings. The GNOME configuration system, and the mechanism for centrally locking desktop settings. §9.2.

FAI. Fully Automatic Installation, a Debian-family unattended provisioning system. §8.2.

Flatpak. A distribution-independent packaging and sandboxing format for desktop applications. §10.4.

Golden image. The centrally-built base system installed on every endpoint. §8.3.

Group Policy. The Windows directory-delivered settings system, which has no direct Linux equivalent. §9.2.

Immutable / image-based system. A design in which the operating system is shipped and updated as a whole, atomically, with rollback. §5.4.

Kiosk framework. KDE's mechanism for locking desktop settings centrally. §6.

LCM. Life-cycle management, here the front end coordinating provisioning, configuration, patching, and reporting across an estate. §9.1.

LUKS. The Linux disk-encryption format. §12.4.

PKCS#11. The standard interface between applications and cryptographic tokens such as smartcards. §12.3.

Satellite. A local server providing management services at a remote site under central control. §8.4.

SSSD. System Security Services Daemon, which connects a Linux client to a directory and caches credentials for offline use. §12.1.

TPM. Trusted Platform Module, the hardware element used to bind disk encryption to a machine. §12.4.

Wayland. The display protocol that replaced X11 as the default, with consequences for remote support and accessibility. §6, §14.

Wine. A compatibility layer running Windows applications on Linux without a Windows licence. §16.2.

WSJF. Weighted Shortest Job First, the prioritisation method of §2.1.

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.)

Direction générale de l'armement (2021) *Étude Internet poste libre*, Direction Opérations, Unité de Management du Socle Numérique. Paris: Ministère des Armées.

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) 'You cannot use the GNU (A)GPL to take software 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.

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)

- `../notes/common/sources.md`: provenance, method, and limits of each source document.
- `../papers/notes/workstation-wp.md`: this paper's internal companion, holding version history, verification debt, parked material, open questions.

Annex B. Reproduction: scripts, data, and figures

- `../verify/plot_breakeven.py`: Figure 1, shared with the companion guide. Run `uv run --with matplotlib verify/plot_breakeven.py`; its self-check verifies agreement with the §21.1 table.
- `../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/cost_model.py`: the parametric cost model of §21, shared with the companion guide. Deterministic, no random seed required. `uv run verify/cost_model.py` produces the scenario tables; `--self-check` runs the assertions alone. The self-check runs on every invocation.
- The §21 and §21.1 tables are generated by that script's workstation scenario and pasted; regenerate after any parameter change, and `check_paper_tables.py` fails if it is not done.
- The dominance threshold of §21 is closed form, not simulated: it is the technical effort bill divided by the seat count times the loaded user-day cost, asserted at the crossing point in the model's self-check.
- Effort, day rate, and productivity loss are placeholders, labelled as such in the script output. Published prices carry their source and date (February 2024) and are stale by construction.
- The WSJF scores in §2.1 are transcribed from Standtke and Tiede (2024b) and are not recomputed here. §2.3 describes how to recompute them for a different organisation.
- Figures: `papers/figures/breakeven-workstation.png` (Figure 1), regenerated by `plot_breakeven.py`.