OpenRS 2025 Technology End-of-Year Report

Richard Staunton-Mann , Casey Henderson , Ian Ibbotson.
OpenRS / Knowledge Integration Ltd

Licensing and Use
This work is © 2026 OpenRS, K-Int and licensed under the [Creative Commons Attribution 4.0 International License](https://creativecommons.org/licenses/by/4.0/). You are free to share and adapt the material with appropriate attribution. Attribution should credit both the authors and K-Int and OpenRS. Excerpts must not be presented in a misleading or out-of-context manner, and any modifications must be clearly indicated.

Summary

This document reflects on OpenRS's technical experience through 2025 and sets out the thinking that is shaping its next phase. It is both retrospective and forward-looking, grounded in sustained real-world operation — particularly through the MOBIUS consortium, and through current work in progress — rather than theory alone.

The past two years have demonstrated both the value and the limits of existing approaches to resource sharing. OpenRS's implementation of Direct Consortial Borrowing (DCB) enabled tight integration with circulation systems and made large-scale print sharing operationally viable for public-library consortia. At the same time, that experience revealed deeper structural questions about interoperability, abstraction, identity, trust, and where resource sharing should sit in relation to library management systems.

Several key learnings emerge:

1. **ILL versus DCB is an incomplete framing.**

   These are not opposing philosophies, but different automation strategies suited to different operational contexts. The real challenge lies in supporting multiple forms of resource sharing coherently, without forcing actors and systems into a single model.

2. **Resource sharing cannot be treated solely as an extension of the LMS.**

   While print circulation remains central for many consortia, even a small number of requests that fall outside standard circulation — such as e-book lending, controlled digital lending, or repository-based supply — expose the limits of an LMS-centric approach. The LMS must be treated as one fulfilment service among several.

3. **Contextual automation is the next critical abstraction.**

   Rather than defining generic "workflows", OpenRS is increasingly focused on understanding what is being automated, in which scenarios, and why. Different relationships, materials, and rights contexts require different automations, applied deliberately.

© 2026 Knowledge Integration Ltd, OpenRS. Licensed under CC BY 4.0.

4. **Abstraction matters, even in tightly integrated systems.**

   The OpenRS DCB architecture was a necessary step to escape constraints imposed by earlier deployment models, but experience has shown the continuing importance of strong protocol-level abstraction to maintain clean boundaries and long-term flexibility.

5. **Centralisation creates power as well as efficiency.**

   Shared identifiers and indexes deliver real operational benefits, but they also carry the risk of unintended authority and dependency. OpenRS is explicitly seeking to avoid recreating central hubs that become too big to fail, focusing instead on federation, transparency, and substitutability.

Looking ahead to 2026, OpenRS's priorities are shaped by both architectural learning and financial reality. Institutions are not seeking entirely new budget lines, but ways to redirect existing expenditure toward systems that preserve autonomy and trust. OpenRS v2 aims to address structural issues in the resource-sharing ecosystem — not by replacing one central system with another, but by enabling cooperation without concentration of control.

This document is shared as a **first full draft**. Its purpose is to surface learning, invite challenge, and establish a common basis for discussion as OpenRS plans its next iteration.

---

1. Introduction — A Year of Stabilisation, Trust, and Learning

2025 marked the second full year of production operation for OpenRS, and for the MOBIUS consortium in particular. It was not a simple year, nor a smooth one. At several points, the operational, technical, and organisational challenges facing the consortium were significant.

The most impactful learning from 2025 centred on heterogeneous systems interoperability and, more importantly, on the role that *trust* plays when a shared system serves institutions with different technologies, constraints, and expectations.

In practice, OpenRS cannot control the technology choices, roadmaps, or behaviours of downstream third-party systems and vendors. Yet those factors materially shape the day-to-day experience of consortium members. When that experience is uneven, the effect is not only operational but relational: perceptions of priority, responsiveness, and care directly influence trust in the shared system.

This tension was felt particularly strongly this year by libraries running Polaris. OpenRS invested substantial effort throughout 2025 in improving Polaris interoperability, but persistent challenges — many of them outside OpenRS's direct control — meant that

© 2026 Knowledge Integration Ltd, OpenRS. Licensed under CC BY 4.0.

outcomes did not always reflect that effort. As a result, a perception emerged that Polaris institutions were not receiving the same level of attention as others.

That perception matters, regardless of intent. Maintaining trust across a heterogeneous consortium requires not only technical progress, but also clear ownership of outcomes and proactive engagement when experience falls short. OpenRS takes responsibility for that, and this learning is directly informing priorities and communication going into 2026.

Alongside these operational lessons, 2025 also clarified a deeper architectural issue. The decision to build OpenRS DCB by integrating directly with native ILS APIs was entirely necessary to achieve the level of tight circulation integration required by MOBIUS. Treating resource sharing as wide-area circulation — rather than ILL mediated by shipping labels and book bands — depends on that level of system integration.

What we discounted too heavily, however, was the continuing value of the abstraction layer that a protocol such as ISO 18626 provides.

At the time OpenRS was formed, it was clear that the existing ReShare codebase was not amenable to the level of integration MOBIUS required. That assessment remains correct, and it is difficult to see how a different decision could have been made at that moment. In deviating from that architecture, however, OpenRS also deviated from two things at once: ISO 18626 *and* a very particular platform ecosystem.

Abstraction matters, even in tightly integrated systems and with the benefit of hindsight, it is now clear that the core limitation lay not with ISO 18626, but with tightly constrained platforms — particularly in the way Library Services Platforms built around home grown toolking are currently deployed, including the reliance on edge modules and mediated APIs. This creates an artificial barrier to the kinds of deep system integration required when resource sharing is treated as wide-area circulation.

OpenRS recognises the ongoing importance of protocols like ISO 18626 in setting clear boundaries between tightly integrated systems while offering a common language for them to easily talk with each other. These learnings are now shaping our future developments.

This learning aligns directly with a broader reframing explored later in this report: that the LMS should be understood as a component *within* a resource-sharing ecosystem, rather than the system to which resource sharing must always defer.

---

2. Origins of the OpenRS Fork — When Philosophy Met Practice

OpenRS emerged as a fork within the resource-sharing community that had formed around ReShare. While this fork was often framed externally as a disagreement over standards or implementation detail, the underlying causes were more fundamental.

© 2026 Knowledge Integration Ltd, OpenRS. Licensed under CC BY 4.0.

At the time of the split, some members of the ReShare community held the position that inter-library lending, expressed through ISO 18626 messaging, was the primary model for resource sharing. In contrast, the MOBIUS consortium adopted an approach described as *Direct Consortial Borrowing (DCB)*, which is more closely aligned with wide-area circulation.

This difference proved impossible to reconcile within a single project.

With the benefit of operational experience, it is now clearer that the disagreement **should not be framed as ILL versus DCB**. It was about how tightly resource-sharing systems must integrate with downstream platforms, and what architectural constraints are acceptable when doing so.

In practice, the limitation was not ISO 18626 itself. Rather, it lay in the way Library Services Platforms built around the Okapi ecosystem mediate access to downstream systems. Okapi introduces an additional architectural boundary that, while appropriate for many use cases, those architectural boundaries place limits on how effectively resource sharing can operate when it is treated as subordinate to the LMS Platform. In particular, reliance on edge modules and mediated APIs constrained the ability to interact directly and predictably with proprietary downstream systems.

Stated in an alternate way - the challenge was not with ISO18626 and its reasonably trivial dialectic implementation variants, but with the pluggable binding of ISO18626 onto native systems. We had leaned heavily into the “Adapter” semantics - which work well for calling into other ILS implementations, particularly where NCIP2 is available. But DCB style apps rely heavily on interactors, eventing and reactive messaging which are not well supported by the existing implementations of standards. Long polling of downstream systems in particular was the achilles heel of the legacy architecture - which is well adapted to incoming reactive HTTP load.

For MOBIUS, achieving the required level of circulation integration meant working directly with native ILS APIs. That necessity drove the architectural break that led to OpenRS. At the time the decision was made, there was no credible path to delivering the required DCB outcomes within ReShare. The move to OpenRS DCB was not a rejection of abstraction, nor a preference for tighter coupling in principle; it was a necessary step to create a LMS-native experience wide-area circulation model that could not otherwise be achieved.

However, in pulling away from the ReShare architecture, OpenRS also moved away from protocol mediation and into native APIs. This abstraction introduced in OpenRS DCB — expressed through the concept of a “HostLMS” — proved insufficiently strong to enforce clean system boundaries. While it enabled rapid integration, it did not provide the structural discipline needed to keep interfaces stable, minimal, and substitution-friendly. And in a sense HostLMS is an interstitial manifestation of the “Adapter” pattern of previous iterations.

Had OpenRS retained an explicit protocol abstraction at this layer — such as ISO 18626 — the system would have been forced to maintain cleaner separation between orchestration logic and fulfilment mechanisms. That discipline was lost in the pragmatism and urgency of the initial DCB architecture, even though the need for the protocol layer itself did not disappear.

© 2026 Knowledge Integration Ltd, OpenRS. Licensed under CC BY 4.0.

This distinction is critical. OpenRS DCB works not because it set aside ISO 18626, but because it escaped architectural constraints imposed by naive api-gateway deployment models. In hindsight, those two changes should not have been coupled — but decoupling them was only possible *after* the DCB transition had taken place.

This learning now informs the OpenRS v2 architecture: retaining the ability to integrate deeply with downstream systems where necessary, while reintroducing stronger abstraction boundaries that recognise resource sharing as a layer above — not subordinate to — any individual LMS.

---

3. ILL and DCB Revisited — Expedient Framing, Support, and Deeper Learning

In the early phases of OpenRS, the distinction between **ILL and DCB was framed as a primary architectural and philosophical divide**. With hindsight, it is clear that this framing was too coarse.

That framing was, however, *expedient*. OpenRS has always aspired to address problems at their root rather than incrementally patching symptoms, but the reality of delivery timelines, consortium commitments, and operational urgency meant that the project proceeded in stages. The DCB versus ILL framing provided a workable way to make progress under pressure, even if it simplified the underlying problem more than was ideal.

Two related assumptions were made:

1. The depth of technical debt embedded in existing architectures was greater than initially hoped. Addressing that debt in a meaningful way would have required not only incremental change, but substantial architectural refactoring[1].
2. While many of the communities OpenRS interacts with — particularly around FOLIO — explicitly espouse agile principles, those communities are also, understandably, highly risk-averse. In practice, this meant that the level of refactoring required to meaningfully address accumulated technical debt was unlikely ever to be embraced incrementally. The expectation that the architecture could be progressively refined through small course corrections did not fully account for the social and institutional realities of large, shared platforms.

At the same time, it is important to recognise that some progress *was* possible only because of targeted external support. In particular, EBSCO contributed funding that enabled OpenRS to address specific areas of technical debt and architectural limitations. While this support did not — and could not — extend to the level of wholesale refactoring that might be desirable in an ideal world, it was nonetheless critical. Without it, the project would likely have stalled entirely under the weight of accumulated constraints.

---

[1]Managing Technical Debt remains a problem at the nexus of ethical and sustainable use of open source in our domain.

© 2026 Knowledge Integration Ltd, OpenRS. Licensed under CC BY 4.0.

This experience reinforced a deeper and more uncomfortable learning: **software agility is not primarily a technical property, but a community one**. Processes and methodologies alone are insufficient if there is not a shared, practical understanding of what agility requires — including the willingness to undertake disruptive change when it is genuinely necessary.

Agility is not the aim, nor a stylistic preference in delivery. It is a practical necessity in domains where the full shape of the problem cannot be known in advance, and where new constraints and failure modes emerge only through real-world interoperability with vendor systems, institutional processes, and funding structures. In such environments, change is not evidence of poor planning; it is evidence of learning.

Agility also depends on establishing stable funding - in 2025/6 software cannot be treated like a one-off capital expenditure - the cost of maintaining modern software is not a one-time outlay. Worse, the challenge of maintenance where there is a large installed base means that keeping systems up to date and addressing technical issues is likely more expensive than initial construction.

It is fair to say that OpenRS got some of the framing wrong. At the same time, it is difficult to see how that framing could have been bypassed entirely. The DCB transition was a necessary step — not because it represented a final answer, but because it created the conditions under which the next layer of the problem could be clearly seen.

The real error was not in choosing DCB, but in overestimating the degree to which architectural course correction would be possible once delivery pressures, funding constraints, and community risk tolerance were taken into account. Recognising those limits is now central to how OpenRS approaches its v2 architecture and its engagement with partner communities.

It is also important at this moment to acknowledge the fundamental courage and vision shown by the Mobius leadership in the first DCB rollout. To recognise the power of the DCB approach when applied across a heterogeneous set of vendor systems, and be willing to trust the vision and the team has led to what is today becoming something truly innovative and enabling.

---

4. Resource Sharing Above the LMS — Primacy, Exceptions, and Automation

As the operational and architectural lessons of 2025 accumulated, OpenRS was increasingly pushed toward a more fundamental question — not about protocols or workflows, but about *primacy*.

For consortia such as MOBIUS, this question must be approached carefully and honestly. Day-to-day resource sharing remains overwhelmingly focused on the circulation of physical print materials. For the vast majority of requests, treating resource sharing as an extension of standard LMS circulation continues to work well, and OpenRS DCB was explicitly

© 2026 Knowledge Integration Ltd, OpenRS. Licensed under CC BY 4.0.

designed to support that reality. Indeed, when it comes to issues of staff training and cost-per-request the DCB model is persistently hard if not impossible to beat.

The challenge arises not because the DCVB model is wrong in general, but because it struggles at the margins — and those margins are growing.

As soon as even a small number of requests fall outside the standard print circulation flow, **the limitations of an LMS-centric model become apparent**. E-book lending, controlled digital lending (CDL), access to institutional repositories, article delivery, and other non-circulating resources all press firmly on the same pressure point: they cannot be modelled as circulation events without distortion or loss of meaning.

In these cases, the problem is not volume, but *exception handling*. A model that works for 99.99% of requests but breaks on the remaining fraction still fails operationally, because it lacks a coherent way to reason about alternatives.

This is where OpenRS's understanding of primacy has evolved. The LMS remains a critical fulfilment system — particularly for print monographs — but it must be treated as **one of several potential supplying services**, rather than the system to which all sharing activity must ultimately defer. The role of ERM in digital article lending is a keen area of interest now.

From this perspective, the central question shifts. Instead of asking *"how does this request move through the LMS?"*, OpenRS increasingly asks:

What automations can be applied to which services, in which contexts, to satisfy this request appropriately?

This represents a significant conceptual shift. Historically — including within ReShare — systems spoke in broad terms about "workflows" without always being clear about what was actually being automated, or why that automation added value beyond operational convenience or marketing language. Workflows were named, but their intent and scope were often left implicit.

The emerging OpenRS v2 model is more explicit. **Automation is treated as a first-class concept:** a deliberate, contextual decision about how much human intervention is required, which systems should be engaged, and which policies apply. A tightly integrated circulation automation may be appropriate for close consortial print lending. A looser, protocol-mediated automation may be more appropriate for arms-length lending, digital fulfilment, or repository-based supply.

Seen in this light, earlier debates — including ILL versus DCB — are reframed not as ideological divisions, but as disagreements about *which automations should apply in which scenarios*? The real evolution lies in recognising that **no single automation model can serve all forms of resource sharing**, and that forcing all activity through LMS-centred workflows ultimately constrains, rather than enables, libraries.

This understanding now underpins the OpenRS v2 direction. Resource sharing is treated as a coordination layer that selects and applies appropriate automations across multiple services — of which the LMS remains a central, but not exclusive, participant.

© 2026 Knowledge Integration Ltd, OpenRS. Licensed under CC BY 4.0.

5. Centralisation, Identifiers, and Unintended Power

As OpenRS's understanding of primacy and contextual automation has matured, it has brought renewed focus to a long-standing tension in resource sharing: the role of centralised indexes and shared identifiers.

Shared identifiers for works, manifestations, holdings, and instances are powerful tools. They enable de-duplication, improve matching accuracy, reduce operational friction, and support large-scale efficiency. In practical terms, they underpin many of the gains that make modern resource sharing viable at all.

However, identifiers are never neutral.

When identifiers are coupled with centralised indexes, **they quietly shift from being *assistive infrastructure* to *authoritative reference points***. Over time, what began as an operational convenience can harden into a dependency — not because institutions explicitly chose to cede authority, but because the cost of operating without that centre gradually became prohibitive.

This dynamic is subtle, but its consequences are profound.

Once a single index becomes the place where identity is resolved, policy is encoded, or availability is inferred, it begins to shape behaviour upstream and downstream. Decisions that were once local — about lending, access, prioritisation, or fulfilment — become implicitly constrained by the assumptions and data models embedded in that central service.

In this way, efficiency can quietly transform into governance.

The risk is not that centralisation exists at all — some degree of shared infrastructure is both necessary and desirable — but that its scope and authority expand without explicit consent or ongoing scrutiny. What was invented to reduce duplication can, over time, become "too big to fail", not because it is mandated, but because alternatives have been structurally crowded out.

From the perspective developed earlier in this report, this risk becomes clearer. If resource sharing is treated as an orchestration layer applying contextual automations across multiple services, then identifiers and indexes should serve that orchestration — not define it. They must remain tools, not gatekeepers.

This has direct implications for the future of OpenRS.

OpenRS is explicitly seeking to avoid recreating a model in which a single central index or clearing house holds disproportionate authority over discovery, routing, or fulfilment. The goal is not to replace one hub with another under a different name, but to design shared infrastructure that enables cooperation without concentrating control.

© 2026 Knowledge Integration Ltd, OpenRS. Licensed under CC BY 4.0.

That requires difficult design choices. It means privileging federation over aggregation, transparency over opacity, and substitution over lock-in — even when those choices complicate implementation or slow initial rollout.

It also requires recognising that **identifiers derive their legitimacy from the institutions that use them, not from the systems that store them**. In an ecosystem built around trust, identifiers must be shared *by agreement*, not imposed by necessity.

This principle now sits alongside contextual automation as a cornerstone of the OpenRS v2 architecture. Together, they provide a framework for achieving efficiency without dependency, coordination without centralisation, and scale without surrendering autonomy.

---

6. What We Learned — And Why It Matters

Taken together, the experiences of 2025 have reshaped how OpenRS understands the problem it is trying to solve.

Early debates framed the challenge as a choice between competing models — ILL versus DCB, loose versus tight coupling, protocol versus direct integration. Those distinctions were useful in making progress under delivery pressure, but they obscured a deeper truth: **Resource sharing is not a single workflow, and it cannot be reduced to a single automation model**.

The most important learning from the past two years is that resource sharing is best understood as a collection of *context-dependent automations*, applied selectively across a heterogeneous landscape of systems, materials, and institutional relationships.

From this perspective, many earlier tensions fall into place.

DCB proved effective because it applied a tightly integrated automation where that automation made sense: high-volume, consortially close, print monograph lending. Traditional ILL models continue to make sense where looser coupling, greater human mediation, or protocol-level abstraction are required. Digital fulfilment paths — whether through repositories, CDL, or licensed electronic collections — introduce entirely different constraints and opportunities again.

The mistake was not choosing one approach over another. It was the assumption that a single workflow could adequately describe all of resource sharing.

Equally important is the recognition that *how* automation is applied matters as much as *where*. Automation that merely reproduces legacy workflows at scale offers limited value. Automation that is explicit about intent — about **what decision is being delegated to the system, and under what conditions** — enables clarity, trust, and adaptability.

This has implications beyond technology.

© 2026 Knowledge Integration Ltd, OpenRS. Licensed under CC BY 4.0.

It affects how consortia set policy, how vendors integrate, how communities negotiate change, and how risk is shared. It also shapes how success is measured: not by uniformity of process, but by appropriateness of outcome.

By the end of 2025, OpenRS no longer sees its task as choosing the “right” resource-sharing model. Instead, it is focused on building an architecture capable of supporting *multiple* models coherently — without collapsing them into a lowest common denominator or forcing them through a single system of record.

This shift in understanding matters because it reframes both past disagreements and future decisions. It allows OpenRS to move beyond inherited binaries, to address structural issues directly, and to design systems that reflect how libraries actually operate — not how procurement language or historical standards once described them.

The remainder of this report looks forward, setting out how these lessons shape OpenRS’s priorities for 2026 and the next iteration of its architecture.

---

7. Looking to 2026 — Architecture, Sustainability, and Trust

As OpenRS looks ahead to 2026, the lessons of the past two years point to a clear conclusion: the next phase of work must address not only technical architecture, but the structural conditions in which resource sharing operates.

Across the library ecosystem, institutions are under sustained financial pressure. There is limited appetite — and often limited capacity — to fund entirely new services. What libraries are actively seeking instead are ways to redirect existing expenditure toward systems that offer greater transparency, flexibility, and institutional control.

This reality shapes OpenRS’s priorities.

There is a widely recognised structural problem in the resource-sharing landscape: critical infrastructure has become increasingly centralised, opaque, and difficult to substitute. While such systems often began as operational efficiencies, over time they have accumulated authority and dependency in ways that were not always explicitly intended by their participants.

OpenRS’s challenge is not to criticise that outcome, nor to recreate it under a different banner. The task for 2026 is to address the *structure* of the problem itself — to enable cooperation, coordination, and efficiency **without** concentrating control or creating another system that becomes “too big to fail”.

There has been a deliberate shift in emphasis for v2: away from treating flexibility as something that can be preserved in abstraction, and toward making uncertainty, trade-offs, and decision points explicit from the outset. The objective is not agility for its own sake, but

© 2026 Knowledge Integration Ltd, OpenRS. Licensed under CC BY 4.0.

the capacity to respond coherently as new problems surface — in ways that a real community, operating under real constraints, can sustain over time.

This has direct architectural consequences.

OpenRS v2 is being shaped around a small number of foundational principles established through experience:

- Contextual automation, rather than uniform workflows
- Strong abstraction boundaries, even where deep integration is required
- Federation over aggregation, wherever feasible
- Shared infrastructure without central authority
- Substitutability as a design goal, not an afterthought

These principles are not abstract ideals. They are responses to specific operational, financial, and governance pressures observed over multiple years of real-world use.

Sustainability remains a critical concern. OpenRS must evolve into a service that institutions are willing to fund on an ongoing basis — not because it is novel, but because it delivers tangible value and reduces dependency risk. In practice, this means aligning OpenRS with existing budget lines, supporting incremental adoption, and avoiding architectures that force all participants into a single mode of operation.

Trust is central to this effort.

Trust is built when institutions believe that:

- Their needs are treated as first-class, even when they differ from the majority
- Architectural decisions are made transparently and revisited when necessary
- No single organisation, vendor, or index accumulates disproportionate influence
- Participation does not entail surrendering long-term autonomy

OpenRS's experience with MOBIUS has reinforced that trust is not established through intent alone, but through sustained attention to outcomes — particularly when those outcomes are uneven or difficult.

In 2026, OpenRS will focus on translating the conceptual advances outlined in this report into concrete architectural change. This includes revisiting protocol boundaries, strengthening abstraction layers, expanding the range of fulfilment services that can be orchestrated coherently, and ensuring that no single automation path is treated as normative.

The goal is not to resolve every tension in resource sharing, but to create an environment in which those tensions can be navigated deliberately, without defaulting to centralisation or rigid models.

OpenRS enters 2026 with greater clarity about the problem it exists to solve, and with a renewed commitment to addressing it at the level that matters: structure, not rhetoric; architecture, not allegiance; trust, not dependency.

© 2026 Knowledge Integration Ltd, OpenRS. Licensed under CC BY 4.0.