

WHITEPAPER · ARCHITEKTUR & GOVERNANCE

Vibe-Coding-Realität: Zugriff statt Kopie

Was sich verändert, wenn Makler Daten aus einer Enterprise-API dauerhaft selbst speichern.

**Ein modernes Frontend zu bauen ist Entwicklung.
Daten dauerhaft zu speichern ist Betrieb.**

Beides ist möglich. Es sind aber unterschiedliche Verantwortungs- und Risikomodelle.

WORUM ES GEHT

Moderne Werkzeuge machen eigene Makler-Frontends in kurzer Zeit möglich. Die Entscheidung dahinter bleibt dieselbe: Werden Daten zur Laufzeit aus dem führenden System abgerufen, zeitlich begrenzt zwischengespeichert oder dauerhaft in einer eigenen Datenbank gehalten? Dieses Whitepaper vergleicht die drei Modelle, ordnet die Pflichten aus DSGVO und DORA differenziert ein und liefert eine Entscheidungsmatrix sowie ein Arbeitsblatt mit zwölf Fragen für den nächsten Architekturworkshop.

HERAUSGEBER

IT Warehouse AG

Borsteler Chaussee 51 · 22453 Hamburg
info@it-warehouse.de · +49 40 323354 0

ADRESSATEN

Maklerunternehmen und Integrationspartner, die Daten und Dokumente über eine Enterprise-API in eigene Frontends integrieren.

Stand: 21. August 2026

Dieses Whitepaper ist eine technische und organisatorische Orientierung. Es ersetzt keine Rechtsberatung und keine Prüfung des konkreten Einzelfalls.

Inhalt

Kapitel 10 ist als Arbeitsblatt gestaltet und kann einzeln in Architektur- und Projektworkshops verwendet werden.

Executive Summary.....	3
1 Ausgangslage: moderne Oberfläche, neue Betriebsrealität.....	4
2 Die Architektur im Bild.....	5
3 Build und Operate sind zwei verschiedene Disziplinen.....	6
<i>Aus der Praxis: ein Maklerhaus und ein fertiges Frontend.....</i>	<i>7</i>
4 Was sich durch persistente Speicherung konkret ändert.....	8
5 DSGVO: nicht der Cloud-Name entscheidet, sondern die Verarbeitung.....	9
6 DORA: für Makler nicht pauschal anwendbar, aber häufig relevant.....	10
7 AWS, Azure und andere Plattformen: Cloud ersetzt keine Governance.....	11
8 Das Performance-Argument: legitim, aber messbar machen.....	12
9 Entscheidungsmatrix.....	13
10 Architektur-Check: Zwölf Fragen vor der Entscheidung.....	15
11 Empfohlener Entscheidungsprozess.....	16
12 Fazit.....	17
Über den Autor · Quellen und Rechtsgrundlagen.....	17

Executive Summary

Vibe Coding, moderne Low-Code-Werkzeuge und KI-gestützte Entwicklung senken die Hürde, individuelle Makler-Frontends zu bauen. Damit rückt die Entscheidung hinter der Oberfläche in den Vordergrund: Werden Daten zur Laufzeit aus dem führenden System abgerufen – oder entsteht beim Makler eine zweite, dauerhaft gespeicherte Datenbasis?

Die zweite Variante ist nicht falsch und nicht unzulässig. Sie verändert aber das Betriebsmodell. Eine persistente Kopie erzeugt zusätzliche Verarbeitungsvorgänge, Speicherorte, Berechtigungsebenen, Löscho- und Berichtigungsprozesse, Protokollierungsbedarfe, Backup- und Wiederherstellungsfragen sowie Abhängigkeiten von Cloud- und Entwicklungsdienstleistern.

DIE ARCHITEKTURENTSCHEIDUNG IN 30 SEKUNDEN

A

Laufzeitabruf

Die Datenquelle bleibt führend. Das Frontend ruft Personen-, Vertrags-, Dokumenten- und Auftragsdaten bei Bedarf über die Enterprise-API ab.

Geringste zusätzliche Betriebsverantwortung.

B

Temporärer Cache

Daten werden zeitlich begrenzt gespeichert, um definierte technische Anforderungen zu erfüllen.

Zusätzliche Datenhaltung mit definiertem Zweck und definierter Lebensdauer.

C

Persistente Replikation

Daten werden dauerhaft in einer eigenen Datenbank oder Cloud-Infrastruktur gespeichert.

Eigener Datenbestand mit Verantwortung für Betrieb, Schutz, Aktualität und Löschung.

Die drei Modelle sind keine Rangfolge. Welches Modell trägt, hängt vom Use Case, vom Datenumfang und von der Fähigkeit ab, den zusätzlichen Betrieb dauerhaft zu führen.

DREI PUNKTE FÜR DIE ENTSCHEIDUNGSVORLAGE

1 Persistenz ist eine Architekturentscheidung

Performance kann ein legitimes Ziel sein. Sie sollte gegen Datenminimierung, Aktualität, Sicherheitsaufwand, Betriebsfähigkeit und regulatorische Nachweisbarkeit abgewogen werden.

2 Die DSGVO-Rolle ändert sich nicht automatisch

Ob ein Makler Verantwortlicher, gemeinsam Verantwortlicher oder Auftragsverarbeiter ist, folgt aus Zwecken und Mitteln der Verarbeitung – nicht allein aus dem Speicherort. Persistenz erweitert aber regelmäßig den eigenen Verarbeitungskontext und die dazugehörigen Pflichten.

3 DORA ist differenziert zu betrachten

Viele Versicherungsvermittler sind als Kleinst-, kleine oder mittlere Unternehmen vom unmittelbaren DORA-Anwendungsbereich ausgenommen. Trotzdem können DORA-nahe Anforderungen mittelbar relevant werden, etwa durch Versicherer, Verträge, Auslagerungs- oder Sicherheitsanforderungen.

LEITGEDANKE

Die Frage lautet nicht: „Dürfen wir die Daten speichern?“ Die bessere Frage lautet: „Welchen zusätzlichen Betrieb übernehmen wir mit dieser Speicherung – und wollen wir ihn dauerhaft beherrschen?“

1. Ausgangslage: moderne Oberfläche, neue Betriebsrealität

Ein typisches Integrationsprojekt beginnt heute pragmatisch: Das Maklerunternehmen möchte seinen Mitarbeitenden oder Kunden ein eigenes, modernes Frontend bieten.

Personenstammdaten, Vertragsdaten, Dokumente und Aufträge werden über eine Enterprise-API bereitgestellt. Die Oberfläche selbst lässt sich mit modernen Entwicklungswerkzeugen in kurzer Zeit erstellen.

Spätestens bei der Frage nach der Datenhaltung teilt sich der Architekturfad.

Modell	Beschreibung	Betriebliche Folge
A. Laufzeitabruf	Das Frontend ruft benötigte Daten bei Bedarf über abgesicherte Services ab. Lokale Zustände bleiben flüchtig oder auf das technisch Notwendige begrenzt.	Die führende Datenhaltung bleibt zentral. Der Integrator betreibt vor allem Oberfläche, Authentisierung, Autorisierung und sichere Kommunikation.
B. Begrenzter Cache	Bestimmte Daten werden zeitlich begrenzt zwischengespeichert, um Antwortzeiten oder Ausfallszenarien zu optimieren.	Zusätzliche Regeln für TTL, Invalidierung, Verschlüsselung, Berechtigungen und Löschung werden nötig.
C. Persistente Replikation	Daten und Dokumente werden dauerhaft in einer eigenen Datenbank, einem Data Store oder Cloud-Speicher abgelegt.	Es entsteht eine zweite Datenhaltung mit eigenem Lebenszyklus, eigener Sicherheits- und Betriebsverantwortung und Synchronisationsbedarf.

Browser-Cache, TLS-Verarbeitung, Session-Zustände und technische Logs bedeuten bereits Verarbeitung. „Die Daten bleiben nur im Quellsystem“ ist deshalb technisch selten wörtlich zu verstehen. Entscheidend ist, ob eine zusätzliche, dauerhaft nutzbare Datenbasis beim Integrator entsteht.

2. Die Architektur im Bild

Der Unterschied zwischen Laufzeitabruf und persistenter Replikation lässt sich am Datenfluss ablesen. In Variante C entsteht zwischen Frontend und Enterprise-API eine zusätzliche Zone, die betrieben, geschützt, protokolliert, aktuell gehalten und am Ende wieder geräumt werden muss.

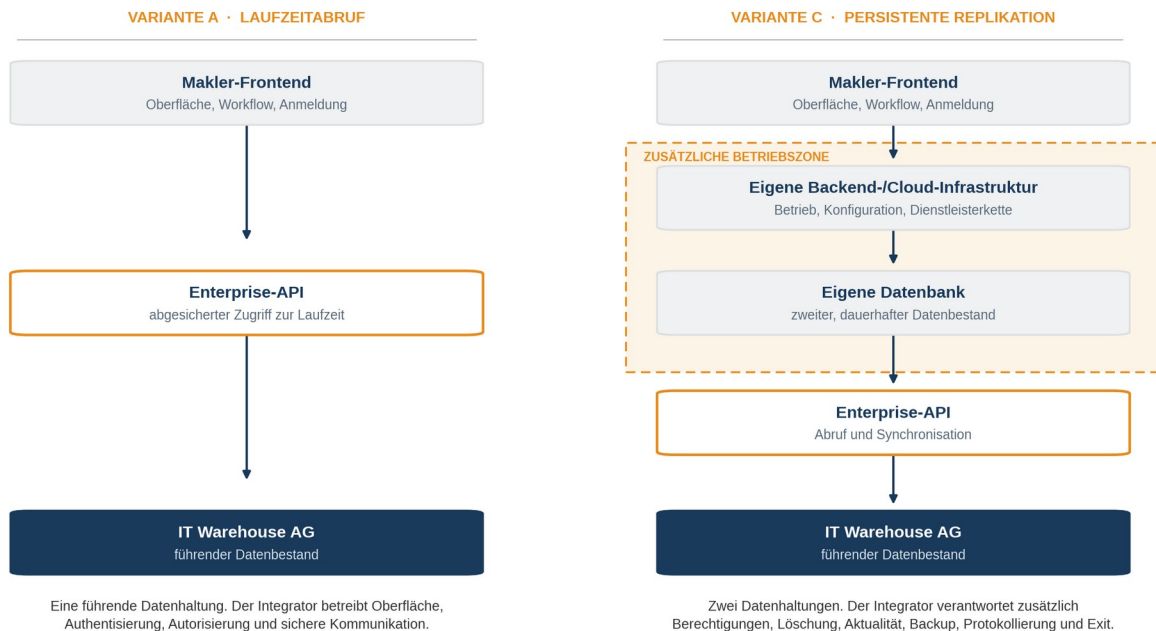


Abbildung 1: Datenfluss bei Laufzeitabruf (A) und bei persistenter Replikation (C). Variante B liegt dazwischen: ein Cache mit definierter Lebensdauer.

Die Darstellung bewertet nicht. Sie zeigt, welche Bausteine im eigenen Verantwortungsbereich liegen. Wer Variante C wählt, sollte diese Bausteine benennen, besetzen und finanzieren können.

3. Build und Operate sind zwei verschiedene Disziplinen

Ein Frontend zu bauen und einen Datenbestand zu betreiben sind unterschiedliche Aufgaben mit unterschiedlichen Fähigkeiten, Rollen und Kostenverläufen. Die Entwicklung endet mit dem Release. Der Betrieb beginnt dort und hört nicht mehr auf.

Build

Einmalig, sichtbar, schnell iterierbar

- modernes Frontend
- User Experience
- Workflow und Fachlogik
- API-Integration
- Vibe Coding und Low-Code
- schnelle Iteration

Operate

Dauerhaft, unsichtbar, nachweispflichtig

- Berechtigungen und IAM
- Löschung und Aufbewahrung
- Security und Verschlüsselung
- Backup und Restore
- Monitoring
- Incident Management
- Auditierbarkeit
- Dienstleistersteuerung
- Datenlebenszyklus

Die sichtbare Qualität eines Frontends sagt wenig über die Qualität seines Betriebsmodells aus.

Beide Disziplinen sind notwendig. Sie werden nur selten von denselben Personen geleistet und selten im selben Budget geplant.

AUS DER PRAXIS

Ein Maklerhaus, ein fertiges Frontend und eine Frage, die niemand gestellt hatte

Ein Maklerunternehmen wollte Personen-, Vertrags-, Dokumenten- und Auftragsdaten über die Enterprise-API in ein eigenes Frontend übernehmen. Ein Entwicklungshaus hatte die Oberfläche bereits gebaut: modern, schnell, fachlich passend. Der Antrieb war nicht Kostenersparnis, sondern Unabhängigkeit – eigene Ideen umsetzen, ohne sie für den übernächsten Sprint eines Dienstleisters anzumelden.

Geplant war, die abgerufenen Daten aus Performancegründen dauerhaft im eigenen Backend zu speichern. Die Performance war dabei nicht gemessen, sondern angenommen.

Im Gespräch wurde die Trennung zwischen Entwicklung und Betrieb zum ersten Mal ausgesprochen: Ein Frontend zu bauen ist das eine, personenbezogene Vorsorgedaten dauerhaft zu verarbeiten das andere. Berechtigungen in der Kopie, Löschung über Backups und Suchindizes hinweg, Protokollierung, Synchronisation, Incident-Prozess und Exit waren im Projekt nicht vorgesehen – sie waren schlicht nicht Thema gewesen.

Das Verfahren wurde daraufhin geändert. Die Daten bleiben im führenden System und werden zur Laufzeit über den Service abgerufen. Das Frontend blieb unverändert.

Der Fall ist anonymisiert wiedergegeben. Er steht nicht für ein Versäumnis des Entwicklungshauses: Im Frontend ist der Betriebsaufwand einer zusätzlichen Datenhaltung nicht sichtbar.

4. Was sich durch persistente Speicherung konkret ändert

Die folgenden Punkte beschreiben keinen Ausnahmefall, sondern den Normalbetrieb eines eigenen Datenbestands.

- **Datenbestand:** Neben dem führenden System entsteht ein weiterer Bestand, der aktuell, vollständig, korrekt und löscher gehalten werden muss.
- **Berechtigungen:** Zugriffsrechte müssen nicht mehr nur im Quellsystem, sondern auch in der neuen Plattform wirksam und nachvollziehbar sein.
- **Löschung und Aufbewahrung:** Löschkonzepte, gesetzliche Aufbewahrungsfristen, Kundenbeendigung und Zweckfortfall müssen auch für Kopien, Backups und abgeleitete Daten gelten.
- **Berichtigung und Aktualität:** Änderungen im Quellsystem müssen in der Kopie zeitnah und zuverlässig nachgeführt werden. Sonst steigt das Risiko widersprüchlicher Datenstände.
- **Protokollierung und Nachweis:** Das Unternehmen muss in seiner eigenen Systemlandschaft nachvollziehen können, welche Daten verarbeitet werden, wer Zugriff hatte und welche Sicherheitsmaßnahmen gelten.
- **Incident Response:** Ein Sicherheitsvorfall in der zusätzlichen Datenhaltung wird zum eigenen operativen Ereignis mit Analyse-, Kommunikations- und gegebenenfalls Meldeprozessen.
- **Cloud- und Dienstleisterkette:** Bei AWS, Azure oder anderen Plattformen kommen Vertrags-, Unterauftrags-, Standort-, Transfer-, Konfigurations- und Exit-Fragen hinzu.
- **Business Continuity:** Backups, Restore-Tests, RTO und RPO, Schlüsselmanagement, Monitoring und Wiederanlauf werden Teil des eigenen Betriebsmodells.
- **Exit und Anbieterwechsel:** Die zusätzliche Datenhaltung muss bei Vertragsende, Plattform- oder Providerwechsel geordnet exportiert, migriert und gelöscht werden können.

5. DSGVO: nicht der Cloud-Name entscheidet, sondern die Verarbeitung

Die DSGVO verbietet weder Cloud-Nutzung noch persistente Speicherung. Sie verlangt, dass personenbezogene Daten zweckgebunden, auf das notwendige Maß beschränkt, richtig, nicht länger als erforderlich gespeichert und angemessen geschützt werden. Art. 5 Abs. 1 DSGVO nennt unter anderem Zweckbindung, Datenminimierung, Richtigkeit, Speicherbegrenzung sowie Integrität und Vertraulichkeit.

Art. 25 DSGVO verankert Datenschutz durch Technikgestaltung und datenschutzfreundliche Voreinstellungen. Die Architekturentscheidung – Laufzeitabruf, Cache oder Replikation – ist damit unmittelbar eine Frage, bei der Datenschutzprinzipien bereits im Design berücksichtigt werden sollten.

Die Rollenfrage

Die bloße Tatsache, dass Daten auf einer eigenen Datenbank oder in einer Cloud gespeichert werden, macht ein Unternehmen nicht automatisch zum Verantwortlichen oder Auftragsverarbeiter. Nach den Leitlinien des Europäischen Datenschutzausschusses kommt es darauf an, wer Zwecke und wesentliche Mittel der Verarbeitung bestimmt.

Für ein Maklerunternehmen kann die datenschutzrechtliche Einordnung je nach Prozess bereits unabhängig von der technischen Architektur feststehen. Eine persistente Kopie ist aber ein zusätzlicher Verarbeitungsschritt, der in die bestehende Rollen-, Vertrags- und Zweckprüfung einbezogen werden muss. Werden Cloud- oder Entwicklungsdienstleister eingesetzt, ist zudem zu klären, ob und wie diese als Auftragsverarbeiter beziehungsweise Unterauftragsverarbeiter eingebunden sind.

Typische Prüffelder bei eigener Datenhaltung

Die folgenden Felder sind bei einer zusätzlichen Datenhaltung regelmäßig zu prüfen. Sie ersetzen keine Einzelfallbewertung.

Zweck & Rechtsgrundlage Warum wird die Kopie benötigt? Ist der Speicherzweck dokumentiert?	Datenminimierung Müssen wirklich alle Daten repliziert werden?
Speicherbegrenzung Welche Fristen gelten für Kopien, Backups und Logs?	Privacy by Design Werden standardmäßig nur notwendige Daten verarbeitet?
Sicherheit nach Art. 32 Verschlüsselung, Zugriffsschutz, Wiederherstellbarkeit, Tests?	Verzeichnis der Verarbeitung Ist der zusätzliche Vorgang dokumentiert?
Auftragsverarbeitung Sind Dienstleister vertraglich eingebunden und prüfbar?	Drittlandtransfers Zugriffe außerhalb des EWR? Welche Transfergrundlage?
Betroffenenrechte Sind Auskunft, Berichtigung und Löschung auch in der Kopie umsetzbar?	Datenschutz-Folgenabschätzung Ist eine DSFA erforderlich? Einzelfallabhängig.

Abbildung 2: Prüffelder und Leitfragen bei eigener Datenhaltung.

6. DORA: für Makler nicht pauschal anwendbar, aber häufig relevant

DORA erfasst Versicherungsvermittler grundsätzlich als Finanzunternehmen. Art. 2 Abs. 3 lit. e DORA nimmt jedoch Versicherungs-, Rückversicherungs- und Versicherungsnebenvermittler aus, wenn sie Kleinstunternehmen oder kleine oder mittlere Unternehmen sind. Die Aussage „Makler müssen DORA erfüllen“ ist deshalb ohne Prüfung von Unternehmensgröße und konkretem Status zu pauschal.

Die Umkehrung ist allerdings ebenso wenig eine Faustformel. Ob ein Vermittler die Größenkriterien erfüllt, hängt von den Definitionen in Art. 3 DORA ab, und die deutsche Umsetzung in § 34d Abs. 13 GewO fasst den Anwendungsbereich enger als der Verordnungstext. Ob und in welchem Umfang DORA-Pflichten unmittelbar greifen, gehört deshalb in die Einzelfallprüfung.

Unabhängig davon kann DORA mittelbar ins Projekt hineinwirken – zum Beispiel, wenn ein Versicherer oder anderer DORA-pflichtiger Vertragspartner Daten, Prozesse oder IKT-Leistungen in die Architektur einbindet und eigene Kontroll-, Audit-, Sicherheits- oder Exit-Anforderungen vertraglich weitergibt.

PRAXISREGEL

Nicht mit „DORA gilt“ oder „DORA gilt nicht“ beginnen. Zuerst den Anwendungsbereich prüfen. Danach klären, welche Anforderungen unmittelbar gesetzlich, vertraglich oder aus der Lieferkette entstehen.

Warum die Architektur für DORA-pflichtige Beteiligte relevant ist

Art. 28 DORA verlangt von Finanzunternehmen, IKT-Drittparteirisiken als Bestandteil ihres IKT-Risikomanagements zu steuern. Die Verantwortung für die Einhaltung bleibt beim Finanzunternehmen. Wird aus einem reinen API-Zugriff eine zusätzliche Datenplattform, können sich damit auch Kritikalität, Abhängigkeit und Prüftiefe des eingesetzten Dienstleister- und Cloud-Setups verändern.

7. AWS, Azure und andere Plattformen: Cloud ersetzt keine Governance

Eine persistente Speicherung in AWS, Microsoft Azure oder einer anderen professionellen Cloud kann technisch sehr sicher umgesetzt werden. Diese Plattformen bieten Sicherheitsfunktionen, die kaum ein Maklerhaus selbst betreiben könnte. Der Einsatz eines großen Hyperscalers löst die Governancefragen jedoch nicht automatisch. Entscheidend sind die konkrete Architektur und Konfiguration sowie die vertragliche und organisatorische Einbindung.

Steuerungsfeld	Was zu klären ist
Region und Datenpfad	Wo liegen Primärdaten, Replikate, Backups und Telemetrie tatsächlich? Welche administrativen Zugriffe sind möglich?
Identity & Access Management	Wer kann auf Daten, Schlüssel, Backups und Administrationsfunktionen zugreifen? Wie werden privilegierte Konten überwacht?
Verschlüsselung und Schlüssel	Wer kontrolliert die Schlüssel? Wie werden Rotation, Recovery und Rollentrennung umgesetzt?
Logging	Welche Logs enthalten personenbezogene Daten? Wie lange werden sie gespeichert und wer wertet sie aus?
Unterauftragsverarbeiter	Welche weiteren Dienstleister sind eingebunden und wie werden Änderungen der Kette kontrolliert?
Backup und Restore	Wie schnell und wie verlässlich kann wiederhergestellt werden? Werden Restore-Prozesse getestet?
Exit	Wie werden Daten vollständig exportiert und anschließend einschließlich Kopien und Backups gelöscht?
Drittlandbezug	Gibt es Übermittlungen oder zugriffsbedingte Drittlandbezüge? Falls ja: Sind die Anforderungen der Art. 44 ff. DSGVO erfüllt?

8. Das Performance-Argument: legitim, aber messbar machen

Persistenz wird häufig mit Performance begründet. Das kann technisch sinnvoll sein, etwa bei großen Datenmengen, komplexen Abfragen, Offline-Fähigkeit oder harten Latenzzielen. „Performance“ ist zunächst aber eine technische Anforderung und noch keine Begründung für einen zweiten dauerhaften Datenbestand. Erst messen, dann speichern.

Frage	Was zu klären ist
Latenz	Welche Antwortzeit benötigt der Nutzer tatsächlich? Wie hoch sind p50, p95 und p99 der relevanten API-Calls?
Volumen	Welche Datenmengen werden je Nutzer und Vorgang wirklich übertragen?
Nutzungsmuster	Werden dieselben Daten häufig wiederholt gelesen oder nur situativ?
Cache-Fähigkeit	Kann ein kurzer, kontrollierter Cache die gleiche Wirkung erzielen wie eine dauerhafte Replikation?
Teildaten	Reicht es, Suchindizes, IDs oder nicht sensitive Metadaten lokal vorzuhalten?
Resilienz	Was soll bei einer API-Störung passieren? Read-only-Cache, Wartungsmodus oder vollständiger Offline-Betrieb?
Konsistenz	Wie werden Änderungen und Löschungen aus dem führenden System zuverlässig in der Kopie nachvollzogen?

Technische Alternativen, die vor der Persistenz geprüft werden können

- Optimierung der API-Nutzung: gezielte Endpunkte, weniger Roundtrips, schlankere Payloads
- gezieltes Caching mit definierter TTL und definierter Invalidierung
- Pagination und serverseitige Filterung statt vollständiger Datenübernahme
- asynchrones Laden und Pre-Fetching für absehbare Nutzerpfade
- CDN für geeignete statische Inhalte
- ereignisbasierte Aktualisierung statt periodischer Vollabgleiche
- lokale Speicherung nur der Metadaten, die für Suche und Navigation wirklich gebraucht werden

Keine dieser Alternativen ersetzt die Messung. Je nach Use Case kann persistente Speicherung technisch die richtige Lösung sein. Dann sollte sie begründet, dokumentiert und betrieblich unterlegt sein.

9. Entscheidungsmatrix

Die Matrix bewertet nicht, welches Modell besser ist. Sie zeigt, was in welchem Modell zu leisten ist.

Kriterium	A - Laufzeitabruf	B - temporärer Cache	C - persistente Replikation
Datenhaltung	keine zusätzliche persistente Fachdatenhaltung	zeitlich begrenzter Zwischenspeicher	dauerhafter eigener Datenbestand
Datenaktualität	Stand des Quellsystems	abhängig von TTL und Invalidierung	abhängig von Synchronisation
Performance-Abhängigkeit	abhängig von API und Netz	teilweise entkoppelt	weitgehend entkoppelt
Betriebsaufwand	Oberfläche und Zugriff	zusätzlich Cache-Regeln	vollständiger Datenbankbetrieb
Berechtigungskonzept	im Quellsystem geführt	Cache-Zugriffe zusätzlich absichern	eigenes Rollen- und Rechtemodell nötig
Löschkonzept	folgt dem Quellsystem	Ablauf über TTL definierbar	eigene Löschr- und Fristenlogik über alle Kopien
Backup und Restore	beim Betreiber des Quellsystems	meist nicht erforderlich	eigene Sicherung, Restore-Tests, RTO und RPO
Incident-Verantwortung	im Wesentlichen Frontend und Zugänge	zusätzlich Cache-Inhalte	eigener Datenbestand ist eigenes Schadensszenario
Synchronisationslogik	entfällt	Invalidierung und Refresh	Abgleich, Konfliktlösung, Fehlerbehandlung
Cloud- und Dienstleister	auf Entwicklung begrenzt	je nach Cache-Technologie	Verträge, Unterauftrag, Konfiguration, Transfers
Exit-Komplexität	gering	gering bis mittel	Export, Migration und nachweisbare Löschung
Auditaufwand	überschaubar	zusätzliche Nachweise zu TTL und Schutz	Nachweise über Betrieb, Schutz und Lebenszyklus

Welches Modell passt zu welchem Ziel?

Ziel oder Anforderung	Tendenz	Begründung
Aktualität der Daten	Laufzeitabruf	Das führende System liefert den aktuellen Stand, Konsistenzprobleme sind gering.
Sehr niedrige Latenz ohne Netzabhängigkeit	Cache oder Persistenz	Kann technisch überlegen sein. Der zusätzliche Betrieb ist bewusst einzuplanen.
Offline-Nutzung	Persistenz	Ohne lokale Datenhaltung meist nicht realisierbar.
Minimierung eigener Datenbestände	Laufzeitabruf	Reduziert dauerhaft gespeicherte Kopien beim Integrator.
Komplexe lokale Analysen	Persistenz oder abgeleitete Daten	Möglich und oft sinnvoll. Zweck, Umfang und Löschung der Analysebestände festlegen.
Schneller Projektstart	Laufzeitabruf	Weniger Betriebsbausteine und keine Synchronisationslogik.
Hohe Eigenständigkeit vom API-Anbieter	Persistenz	Erhöht die Autonomie und zugleich die Verantwortung für Datenlebenszyklus und Aktualität.
Einfache Lösch- und Berichtigungsprozesse	Laufzeitabruf	Weniger Orte, an denen Änderungen technisch nachvollzogen werden müssen.

ARCHITEKTURREGEL

So wenig Persistenz wie möglich.

So viel Persistenz wie fachlich erforderlich.

Und jeder zusätzliche Datenbestand braucht ein Betriebsmodell.

ARCHITEKTUR-CHECK

10. Zwölf Fragen vor der Entscheidung für persistente Datenhaltung

Diese Seite ist als Arbeitsblatt gedacht. Sie kann in Architektur- und Projektworkshops einzeln verwendet werden. Die Fragen ersetzen keine Einzelfallprüfung, schaffen aber ein gemeinsames Verständnis zwischen Fachbereich, Makler, Entwicklung, Datenschutz und IT-Betrieb.

- | | | |
|-----------|---|--------------------------|
| 01 | Welchen konkreten Zweck erfüllt die persistente Kopie? | ☐ |
| | Fachlich und technisch begründet, nicht nur „schneller“. | |
| 02 | Welche Daten werden tatsächlich benötigt? | ☐ |
| | Personenstammdaten, Verträge, Dokumente, Aufträge, Kommunikationsdaten. | |
| 03 | Wie lange müssen sie gespeichert werden? | ☐ |
| | Aufbewahrungsfristen, Zweckfortfall, Vertragsende. | |
| 04 | Wer darf darauf zugreifen? | ☐ |
| | Rollen, Mandantentrennung, privilegierte Konten. | |
| 05 | Wie wird der Zugriff protokolliert? | ☐ |
| | Umfang, Aufbewahrung und Auswertung der Logs. | |
| 06 | Wie werden Datenänderungen synchronisiert? | ☐ |
| | Richtung, Frequenz, ereignisbasiert oder periodisch. | |
| 07 | Was passiert bei widersprüchlichen Datenständen? | ☐ |
| | Konfliktlösung, führendes System, Fehlerbehandlung. | |
| 08 | Wie funktioniert Löschung? | ☐ |
| | Primärdaten, Kopien, Suchindizes, abgeleitete Daten. | |
| 09 | Was befindet sich noch in Backups? | ☐ |
| | Aufbewahrung, Verschlüsselung, Löschnachweis. | |
| 10 | Welche Dienstleister verarbeiten die Daten? | ☐ |
| | Cloud, Entwicklung, Unterauftragsverarbeiter, Drittlandbezug. | |
| 11 | Wie sieht der Incident-Prozess aus? | ☐ |
| | Erkennung, Bewertung, Meldewege, Zuständigkeit. | |
| 12 | Wie werden die Daten bei Vertragsende vollständig entfernt? | ☐ |
| | Export, Migration, nachweisbare Löschung, Exit-Fristen. | |

ERGÄNZENDE PRÜFPUNKTE

- Wer bestimmt Zweck und wesentliche Mittel der zusätzlichen Verarbeitung?
- Welche Vertrags- und Datenschutzrollen haben Makler, API-Anbieter, Entwicklungshaus und Cloud?
- Wo liegen Primärdaten, Replikate, Backups, Logs und Suchindizes?
- Welche Wiederanlauf- und Restore-Ziele gelten und wann wurden sie zuletzt getestet?
- Ist DORA unmittelbar anwendbar? Falls nicht: Gibt es vertragliche DORA-nahe Anforderungen?
- Wurde das Performanceproblem gemessen und wurden Alternativen getestet?

11. Empfohlener Entscheidungsprozess

Acht Schritte, die sich in Projekten bewährt haben. Sie lassen sich in wenigen Tagen durchlaufen und ersparen später Diskussionen über Zuständigkeiten.

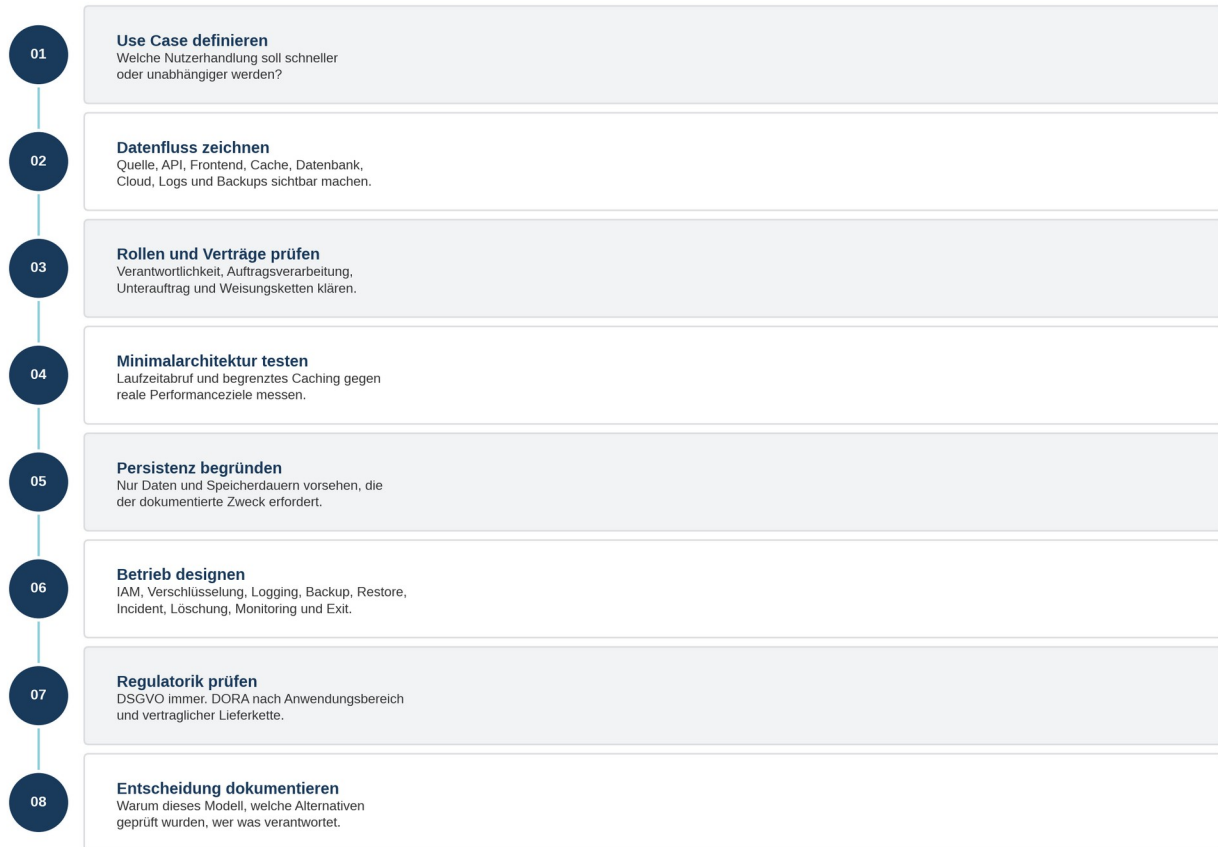


Abbildung 3: Empfohlene Schrittfolge vor der Architekturentscheidung.

12. Fazit

Individuelle Frontends werden einfacher, schneller und günstiger zu entwickeln. Das ist eine Chance: Makler können Prozesse und Nutzererlebnisse sehr viel stärker an die eigene Arbeitsweise anpassen.

Die niedrige Entwicklungshürde ist jedoch keine niedrige Betriebshürde. Wer Daten nur zur Laufzeit nutzt, integriert einen Service. Wer Daten dauerhaft übernimmt, betreibt zusätzlich einen eigenen Datenbestand – mit den dazugehörigen organisatorischen, technischen, datenschutzrechtlichen und gegebenenfalls aufsichtsrechtlichen Konsequenzen.

Die Entscheidung darf für Persistenz ausfallen. Sie sollte nur bewusst ausfallen. Wer die zusätzliche Datenhaltung fachlich braucht und sie professionell betreiben kann, sieht sie als Teil einer sauberen Architektur vor. Wer sie nicht braucht, sollte sich den zusätzlichen Betrieb nicht allein aus einem vermuteten Performancevorteil aufladen.

Die Frage ist nicht, ob Daten kopiert werden dürfen. Die Frage ist, ob der zusätzliche Datenbestand einen Zweck erfüllt, der seinen Betrieb rechtfertigt.

Die IT Warehouse AG unterstützt Integrationspartner gern bei der Auswahl eines geeigneten Architekturmodells für die Enterprise-API.

Über den Autor



Mario Pustan

**Vorstand IT Warehouse AG und smart!Cloud Services AG,
Geschäftsführer Caldera Service GmbH und Insurance Warehouse GmbH**

Krawatte und Kapuze in Personalunion: Vorstand mit bAV-Akte unterm Arm, Entwickler in Jeans und Hoodie. Beides gleichzeitig.

Mario übersetzt seit fast 30 Jahren zwischen Compliance und Code, zwischen Vorstandstisch und Server-Console. Aufsichtsrecht aus dem Maschinenraum erklärt – das ist auch der rote Faden dieses Whitepapers.

[linkedin.com/in/pustan](https://www.linkedin.com/in/pustan)

Quellen und Rechtsgrundlagen

- DSGVO – Verordnung (EU) 2016/679, insbesondere Art. 5, 25, 28, 30, 32, 35 und 44 ff.
- DORA – Verordnung (EU) 2022/2554, insbesondere Art. 2, 3 und 28
- Europäischer Datenschutzausschuss, Leitlinien 07/2020 zu den Begriffen „Verantwortlicher“ und „Auftragsverarbeiter“
- Europäischer Datenschutzausschuss, Hinweise für KMU zur Abgrenzung von Verantwortlichem und Auftragsverarbeiter
- Europäischer Datenschutzausschuss, Materialien zur Rechenschaftspflicht