

WHITE PAPER · DATA GOVERNANCE

# Data Products and Data Contracts in Mesh and Fabric

Ownership, product descriptors, open contract standards and safe evolution

---

White Paper 3 · Data Governance series · Version 1.0 · October 2026

Bassirah Technology — Innovation & R&D

# Data Products and Data Contracts in Mesh and Fabric

White Paper 3 · Data Governance series · Bassirah Technology — Innovation & R&D · 28 October 2026

## Document control

| Field   | Value                                                                                                                                                                  |
|---------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Series  | Data Governance, paper 3 of 6                                                                                                                                          |
| Author  | Bassirah Technology — Innovation & R&D                                                                                                                                 |
| Version | 1.0 · 28 October 2026                                                                                                                                                  |
| Scope   | A technology survey. It describes published research, open specifications and publicly documented systems; it does not report Bassirah benchmarks or customer results. |

## Contents

|                                                         |    |
|---------------------------------------------------------|----|
| 1. Abstract.....                                        | 4  |
| 2. Introduction .....                                   | 4  |
| 2.1 Why products and contracts.....                     | 4  |
| 2.2 About the Data Governance series .....              | 4  |
| 2.3 Scope.....                                          | 5  |
| 2.4 Terms used in this paper .....                      | 5  |
| 3. Mesh, fabric and products .....                      | 5  |
| 3.1 Data mesh.....                                      | 5  |
| 3.2 Data fabric .....                                   | 5  |
| 3.3 Data products .....                                 | 6  |
| 3.4 Complementary, not competing .....                  | 6  |
| 4. Evidence from implementations.....                   | 6  |
| 5. Describing data products.....                        | 7  |
| 6. Data contracts.....                                  | 8  |
| 6.1 Origins and evidence .....                          | 8  |
| 6.2 The Open Data Contract Standard .....               | 8  |
| 6.3 Convergence on one format .....                     | 8  |
| 7. Evolution without breakage .....                     | 9  |
| 8. Enforcing contracts .....                            | 9  |
| 9. Knowing whom a change affects.....                   | 10 |
| 10. Federated computational governance in practice..... | 10 |
| 11. Comparative analysis.....                           | 10 |
| 12. Open challenges and future opportunities.....       | 11 |
| 12.1 Evidence of value .....                            | 11 |
| 12.2 Making federation work .....                       | 11 |
| 12.3 Converging product descriptions.....               | 11 |

|                                                  |    |
|--------------------------------------------------|----|
| 12.4 Contracts beyond schema .....               | 11 |
| 12.5 Contracts at the product level.....         | 11 |
| 12.6 Change at scale .....                       | 12 |
| 12.7 Fabric metadata and mesh ownership .....    | 12 |
| 12.8 Data for AI consumers .....                 | 12 |
| 13. What this means for platform decisions ..... | 12 |
| References .....                                 | 13 |

# 1. Abstract

Data mesh and data fabric are two answers to the same problem: central data teams and central platforms struggle to keep up with the number of sources, consumers and changes in a modern organization. Data mesh moves ownership to business domains and treats data as a product; data fabric builds an integration and metadata layer across distributed sources. Both depend on two governance artifacts: the data product, a managed unit of data with an owner and a purpose, and the data contract, an explicit and testable agreement between the producer of data and its consumers. This paper examines both. It sets out what research says about data mesh, data fabric and data products, including the evidence from implementations; describes the open specifications for describing data products, including W3C DCAT, OMG DPROD, Bitol's Open Data Product Standard and the Data Product Descriptor Specification; and describes data contracts, with the Open Data Contract Standard as the open format that the practitioner community has converged on. It then covers schema evolution rules, how contracts are enforced in build tools, streaming registries, quality engines and catalogs, impact analysis before change, and how contracts implement federated computational governance. It compares the maturity of each part, sets out open challenges and closes with notes for platform decisions.

## 2. Introduction

### 2.1 Why products and contracts

Much of what goes wrong with data goes wrong at boundaries. An empirical study of data pipeline quality, combining a literature review, expert interviews and analysis of developer issues, found that incorrect data types are among the main root causes of data-related problems and that compatibility between pipeline stages is a problem area of its own [1]. Even in databases described as schema-free, an implicit schema lives in application code and keeps evolving over a project's history [2]. When one team changes a table or an event and another team's report, model or application depends on it, the change is a governance event, whether anyone treats it as one or not.

Data mesh was proposed partly in response. Its originator criticized central data lakes and warehouses owned by specialist teams as a bottleneck between the domains that produce data and the people who use it, and proposed domain ownership, data as a product and self-serve infrastructure [3], to which a fourth principle, federated computational governance, was added [4]. Data products and data contracts are the artifacts through which those principles are governed: the product makes ownership and purpose explicit, and the contract makes the producer's commitment explicit and testable.

### 2.2 About the Data Governance series

This is the third of six papers. Paper 1 covered the data that AI systems consume and produce. Paper 2 covered continuous, risk-adaptive governance and policy as code. This paper covers the units that those controls attach to. Paper 4 covers data sovereignty and cross-border compliance, Paper 5 privacy-enhancing computation and Paper 6 data sharing ecosystems, where products and contracts cross organizational boundaries.

## 2.3 Scope

This paper covers the concepts of data mesh, data fabric and data products and the research on them; open specifications for describing data products and data contracts; schema evolution and compatibility rules; enforcement of contracts in tools; impact analysis; and the governance model that connects them. It concentrates on peer-reviewed research, open specifications and official documentation, cites foundational practitioner writing where it introduced an idea still in use, and labels practitioner books, articles and project announcements as such. Statements about specifications and products reflect documentation accessed in October 2026; several of these specifications are revised often. The paper does not quote adoption, performance or survey figures from its sources.

## 2.4 Terms used in this paper

A **domain** is a business area that produces or owns data, such as sales or logistics. A **data product** is a managed artifact that satisfies recurring information needs and creates value by transforming and packaging relevant data into consumable form [5]. An **output port** is the interface through which a data product's consumers read it, such as a table, view, topic or API. A **data contract** is a machine-readable agreement for an output port that states its schema, quality expectations, service levels and terms, and that can be checked against the actual data. A **breaking change** is a change that can make existing consumers fail or produce wrong results. **Data mesh** is a sociotechnical, decentralized approach to enterprise data management; **data fabric** is a design concept for reusable integration services, pipelines and semantics built on metadata [6].

# 3. Mesh, fabric and products

## 3.1 Data mesh

Data mesh rests on four principles: domain-oriented decentralized data ownership and architecture; data as a product; self-serve data infrastructure as a platform; and federated computational governance [4]. The data product is described as the architectural quantum of the mesh, bundling the code, data, metadata and infrastructure needed to serve it [4]. Governance is a federation of domain and platform product owners that sets global rules for interoperability, with the platform executing those decisions automatically [4]. The principles were set out at book length in 2022 [7]. These primary sources are practitioner writing, not research, and the peer-reviewed literature that followed treats data mesh as a paradigm shift away from monolithic architectures [8]. A systematic review of the industry literature maps practitioner descriptions of the four principles to service-oriented reference architectures and lists open research questions [9].

## 3.2 Data fabric

Data fabric is a different kind of concept. A peer-reviewed comparison describes it as a design concept for reusable integration services, data pipelines and semantics, built on metadata modeled as a knowledge graph, and notes that the term was propagated mainly by an analyst firm rather than by research [6]. An early academic comparison of data fabric, the logical data warehouse and data mesh made the same observation [10]. Peer-reviewed work on data fabric is still mostly architecture for specific domains; a recent example for

manufacturing combines uniform messaging, semantic integration, automated metadata generation and connector templates around a central knowledge graph [11].

### 3.3 Data products

The term data product is older than data mesh but was long used loosely. Research in information systems defines it, from literature and case studies, as a managed artifact that satisfies recurring information needs and creates value through packaging data into consumable form, and identifies its characteristics and categories [5]. The Data Product Canvas extends this into a design tool that assesses products for customer desirability, technical feasibility and economic viability and supports a managed portfolio of products [12]. Internal enterprise data marketplaces, which let employees find and access company data, have been studied as a consumption layer distinct from a data catalog [13]. The FAIR principles, written for scientific data, offer a useful yardstick by analogy: products should be findable, accessible, interoperable and reusable, with metadata that machines as well as people can act on [14].

### 3.4 Complementary, not competing

Mesh and fabric are often presented as alternatives. The research suggests a different framing. A review that delimits data platforms, data spaces, data meshes and data fabrics characterizes platforms and fabrics as centralized and control-oriented and meshes and data spaces as decentralized and flexibility-oriented [15]. The comparison in Table 1 follows a peer-reviewed analysis that treats data products as the unit of consumption and value, data fabric as the provision and integration layer, and data mesh as the sociotechnical model that bridges them; the three overlap on catalogs, active metadata and the lifecycle of data products, and are best combined [6].

| Aspect           | Data product                                  | Data mesh                                | Data fabric                                    |
|------------------|-----------------------------------------------|------------------------------------------|------------------------------------------------|
| What it is       | A managed unit of data with owner and purpose | A sociotechnical operating model         | A design concept for integration and metadata  |
| Main emphasis    | Consumption and value                         | Ownership and federated governance       | Provision and integration                      |
| Ownership        | Product owner                                 | Business domains                         | Usually central                                |
| Governance style | Lifecycle and quality of each product         | Federated rules executed by the platform | Central policies over a metadata layer         |
| Key artifacts    | Descriptor, output ports, contract            | Domains, products, self-serve platform   | Knowledge graph, catalog, integration services |

The practical question is therefore not mesh or fabric, but which decisions are federated to domains, which are kept central, and how the products and contracts at the boundary are described and enforced.

## 4. Evidence from implementations

Empirical studies of data mesh are few but consistent. An interview study of practitioners found that the hardest challenges were the transition to federated governance, shifting responsibility for data products to domains, and understanding the concept itself; it recommended a cross-domain steering unit, monitoring of how products are used, early quick wins and small dedicated product teams [16]. A case study at a bank, written by practitioners, framed

governance around the data catalog, data quality and data ownership and argued that it needs change management and aligned incentives as well as technology [17]. A public-sector case study in Norway described moving from central data management to a distributed approach because central management conflicted with agile development, and reported both benefits and challenges [18].

On the platform side, practitioners building a mesh platform found inconsistent terminology and little guidance for implementing governance, and proposed driving governance through automation and self-service infrastructure [19]. A catalog of architectural design decisions for self-serve data platforms, validated with experts, positions the platform as what makes products and computational governance practical [20]. Two cautions apply. Several of these studies are written by consultancies that implement data mesh, and none measures outcomes against a control. The evidence supports the claim that governance and ownership are the hard part; it does not yet show by how much data mesh improves results.

## 5. Describing data products

A data product needs a description that both people and machines can read: owner, domain, purpose, output ports, lifecycle status and the contracts that apply. Several open specifications now provide one, as Table 2 shows.

W3C's Data Catalog Vocabulary, version 3, is the established standard for describing catalogs, data sets, distributions and data services, and adds versioning properties and data set series [21]. It does not define a data product. The Data Products Ontology, DPROD, now an Object Management Group specification at version 1.0 beta 2, is a profile of DCAT that does: a data product has an owner, a lifecycle status, a purpose, a domain, and input and output ports that are data services, so that product metadata can be aggregated across marketplaces [22]. Bitol's Open Data Product Standard, version 1.1.0, maintained under the LF AI & Data Foundation, defines products with input and output ports, management information, support and team, and each output port can reference the data contract that governs it [23]. A separate Open Data Product Specification, version 4.1, from the Open Data Product Initiative, uses the same abbreviation but is oriented to products as commercial offerings, with pricing plans, licensing, access methods and contracts [24]. The Open Data Mesh Initiative's Data Product Descriptor Specification, version 1.0.0, describes a product through its input, output, discovery, observability and control ports and internal components, with promises, expectations and obligations on output ports [25].

| Specification                                | Maintainer                     | Version as of October 2026 | Focus                                             |
|----------------------------------------------|--------------------------------|----------------------------|---------------------------------------------------|
| <b>DCAT</b>                                  | W3C                            | 3 (Recommendation, 2024)   | Catalogs, data sets and data services             |
| <b>DPROD</b>                                 | Object Management Group        | 1.0 beta 2                 | Data products as a DCAT profile                   |
| <b>Open Data Product Standard</b>            | Bitol, LF AI & Data Foundation | 1.1.0                      | Products, ports and links to contracts            |
| <b>Open Data Product Specification</b>       | Open Data Product Initiative   | 4.1                        | Products as offerings, with pricing and licensing |
| <b>Data Product Descriptor Specification</b> | Open Data Mesh Initiative      | 1.0.0                      | Products through ports and components             |

The fields these specifications share, owner, domain, ports, lifecycle and a link to a contract or service level, are a good indication of what a product description needs. The landscape is fragmented, and there is no single dominant standard for products as there now is for contracts (Section 6).

Platforms implement products in different ways. Starburst Enterprise documents a data product as a schema of views or materialized views, with a required domain and owner, a draft, published and pending-changes lifecycle, and definitions that can be managed as code through YAML and a command-line tool [26]. DataHub documents an open-source data product entity that groups related assets under one domain, defined in YAML and synchronized with version control [27]. Atlan documents products organized by domain, with output ports, lifecycle states from draft to archived, and contracts linked to output ports [28].

## 6. Data contracts

### 6.1 Origins and evidence

Data contracts emerged from practice. Two practitioner books treat them at length [29], [30], and the main peer-reviewed treatment found for this paper is a systematic review of the practitioner literature on data contracts in data mesh [31]. The concept therefore rests on practitioner experience and on older research on adjacent problems: automated data validation in production machine learning pipelines [32], declarative "unit tests for data" [33], the causes of data pipeline failures [1] and the extensive literature on API evolution and its effect on consumers, which is a useful analog but does not cover data contracts [34].

### 6.2 The Open Data Contract Standard

The Open Data Contract Standard, ODCS, is an open YAML standard for data contracts maintained by the Bitol project under the Linux Foundation's LF AI & Data Foundation [35]. It began as a data contract template developed at PayPal [36]. Bitol moved from sandbox in 2023 to incubation in 2024 and, according to its own announcement, graduated in mid-2026 [37]. The current version, 3.2.0, released in September 2026 together with version 1.1.0 of the companion product standard, is declared backward compatible with version 3.1.0 [38].

An ODCS contract is organized into sections for fundamentals, schema, context, references, data quality, support and communication channels, pricing, team, roles, service-level agreement, infrastructure and servers, custom properties, authoritative definitions, tags and variables [35]. Version 3.2.0 added, among other things, a context section with instructions and constraints for AI agents that consume the data, variables for contracts that span environments, and a deprecation flag on fields [38]. The companion product standard lets each output port name the contract that governs it, so the link between product and contract is explicit [23].

### 6.3 Convergence on one format

The practitioner community has largely settled on ODCS. The Data Contract Specification, an alternative format, now carries a deprecation notice recommending migration to ODCS, with support in its tooling promised until the end of 2026 [39]. The open-source Data Contract CLI works natively with ODCS and documents commands to lint a contract, test it against the live data source it names, compare versions for breaking changes, and import and export other

formats [40]. A single open format matters for governance: contracts can be checked by different tools, stored with code, and read by catalogs and platforms without translation.

## 7. Evolution without breakage

Data products must change, and contracts are mostly about how. Serialization formats provide the formal foundation. Apache Avro defines how a reader's schema resolves against a writer's: fields are matched by name, fields unknown to the reader are ignored, fields missing from the writer take the reader's default value and fail without one, and certain type promotions are allowed [41]. Protocol Buffers rests compatibility on field numbers: adding fields is safe, removing a field is safe only if its number is never reused, and some type changes are wire-compatible but can alter values [42]. Confluent's Schema Registry turns such rules into enforced compatibility modes, backward, forward, full and their transitive variants, each implying an order of upgrade: consumers first for backward compatibility, producers first for forward [43].

Analytical tables need the same discipline. dbt's model versions let a producer publish a breaking change as a new version while consumers move on their own schedule, with a deprecation date after which consumers are warned [44]. Its documentation distinguishes breaking changes, such as removing or renaming a column or changing its type, from non-breaking ones, such as adding a column [44]. The resulting rules are simple to state and worth writing into every contract: add fields with defaults, deprecate before removing, publish a new version for any breaking change, and give consumers a stated window to migrate.

## 8. Enforcing contracts

A contract that is not checked is documentation. Enforcement happens in tools, at different points:

- **At build time.** dbt model contracts require every column to be declared with its name and type, check the model's output against the declaration before building and fail the build on mismatch; whether constraints such as uniqueness are enforced depends on the data platform [45]. dbt also detects breaking changes against the previous state of a project [45].
- **At write time.** Confluent extends registered schemas into data contracts with metadata, tags, domain rules that validate or transform data on write or read, and migration rules between major versions; failures can stop a record or send it to a dead-letter topic. These rules require Confluent's enterprise or advanced cloud governance editions [46].
- **On the data.** Soda Core version 4 verifies YAML contracts with schema, missing-value, validity, duplicate and row-count checks from the command line or an API [47]. The Data Contract CLI tests an ODCS contract against the data source it describes [40]. These build on the research on automated validation [32], [33].
- **In the catalog.** DataHub defines a contract as a bundle of verifiable assertions on schema, freshness, volume and columns; contract status follows assertion results, which can come from DataHub Cloud or from external tools such as dbt tests [48]. The open-source metadata model records contracts for data sets but does not enforce them automatically; enforcement depends on the assertions being evaluated [49].

No single tool covers schema, quality, service levels and semantics end to end. A workable design keeps the contract in one open format, runs schema checks at build or write time, runs quality checks on the data on a schedule or on change, and publishes results to the catalog, where owners and consumers can see whether each contract is being met.

## 9. Knowing whom a change affects

Before changing a contract, a producer needs to know who reads the affected fields. OpenLineage, an open standard for lineage collected while jobs run, records each data set's schema as a facet and supports column-level lineage [50]. DataHub's impact analysis shows downstream dependencies at a chosen depth, with owners and domains, and is framed explicitly for assessing a breaking schema change; it is available in the open-source edition and depends on lineage having been collected [51]. Combined with contract versioning, impact analysis turns a breaking change from a surprise into a planned migration: the producer sees the affected consumers, notifies their owners and sets a deprecation date.

## 10. Federated computational governance in practice

Federated computational governance asks a federation of domain and platform owners to set global rules and the platform to execute them automatically [4]. Contracts and product descriptors are how that works in practice. The federation decides the rules that every product must meet: required fields in each descriptor, mandatory contract sections, evolution policy, classification tags and quality thresholds for critical products. The platform encodes them as checks: a product cannot be published without an owner, a domain and a contract on each output port; a contract cannot be merged if it fails linting or breaks compatibility without a new version; and a product whose contract checks fail is flagged in the catalog. Paper 2 described the policy engines and tag-driven controls that enforce access and use; contracts add the producer's commitments on shape, quality and service.

The evidence suggests where to be careful. The move to federated governance is the most frequently reported difficulty [16], governance needs incentives and change management as well as tools [17], and practitioners report a lack of guidance on implementing governance on the platform [19]. Automated checks reduce the burden on domains, but someone must still own the rules, arbitrate disputes between producers and consumers and decide when a product should be retired.

## 11. Comparative analysis

Table 3 gives a qualitative judgment of maturity, drawn from the sources cited. It describes the state of practice, not any single product.

| Area                                      | State of practice                                                                              | Maturity |
|-------------------------------------------|------------------------------------------------------------------------------------------------|----------|
| <b>Contract format</b>                    | One graduated open standard; the main alternative deprecated in its favor                      | Mature   |
| <b>Schema evolution and compatibility</b> | Formal rules in serialization formats; enforced modes in registries; versioning in build tools | Mature   |
| <b>Product descriptors</b>                | Several open specifications with common fields; no dominant one                                | Maturing |

| Area                                     | State of practice                                                                                | Maturity     |
|------------------------------------------|--------------------------------------------------------------------------------------------------|--------------|
| Product-to-contract links in platforms   | Present in some standards and catalogs; contracts often attach to data sets rather than products | Maturing     |
| End-to-end enforcement                   | Split across build tools, registries, quality engines and catalogs; some features commercial     | Maturing     |
| Impact analysis                          | Lineage standards and catalog impact views available; depends on lineage coverage                | Maturing     |
| Federated governance operating model     | Recognized as the hardest part; little outcome evidence                                          | Open problem |
| Evidence that contracts improve outcomes | Reviews of practitioner literature; no controlled evaluations found                              | Open problem |
| Contracts for AI agents                  | Newly specified in ODCS 3.2.0                                                                    | Early        |

The formats and the schema mechanics are ready. The open questions are organizational and empirical: how federation works over time, and what difference products and contracts make.

## 12. Open challenges and future opportunities

### 12.1 Evidence of value

**Challenge.** Peer-reviewed work on data contracts mostly synthesizes practitioner writing, and implementation studies do not measure outcomes [31], [16]. **Research direction.** Longitudinal studies of incidents, change lead times and consumer trust before and after contracts.

**Opportunity.** Evidence would let organizations decide where contracts are worth their cost.

### 12.2 Making federation work

**Challenge.** The transition to federated governance and domain ownership is the most reported difficulty [16], [17]. **Research direction.** Operating models, incentives and steering structures for federated governance [19], [20]. **Opportunity.** Proven models would reduce the risk of moving ownership to domains.

### 12.3 Converging product descriptions

**Challenge.** Several product specifications coexist, two of them with the same abbreviation [22], [23], [24], [25]. **Research direction.** Mappings between specifications and a DCAT-based core [21]. **Opportunity.** A common product description would let catalogs and marketplaces exchange products without custom integration.

### 12.4 Contracts beyond schema

**Challenge.** Schema is checked well; quality, service levels and meaning are checked unevenly and in different tools [45], [48], [47]. **Research direction.** Executable quality and service-level sections and semantic definitions tied to business glossaries [35]. **Opportunity.** Contracts that test meaning as well as shape would catch errors that types cannot.

### 12.5 Contracts at the product level

**Challenge.** Catalogs often attach contracts to individual data sets rather than to products, and record rather than enforce them [49]. **Research direction.** Product-level contracts that

cover all output ports [23]. **Opportunity.** Product-level commitments would match how consumers think about what they depend on.

## 12.6 Change at scale

**Challenge.** Breaking changes require knowing every consumer, and lineage coverage is rarely complete [50], [51]. **Research direction.** Automatic consumer discovery and migration support, borrowing from API evolution research [34]. **Opportunity.** Safe, planned change would let producers improve products without fear of breaking others.

## 12.7 Fabric metadata and mesh ownership

**Challenge.** Fabric architectures emphasize central metadata and integration, meshes emphasize distributed ownership, and research on combining them is early [6], [15], [11].

**Research direction.** Knowledge-graph metadata layers that carry ownership, contracts and policies from domains. **Opportunity.** A combined design would offer integration without taking ownership away from domains.

## 12.8 Data for AI consumers

**Challenge.** AI agents consume data without reading documentation, and contract sections for them are new [38]. **Research direction.** Machine-readable context, constraints and permitted uses for agent consumers, connected to the controls in Papers 1 and 2. **Opportunity.** Contracts that agents can read would make AI use of data products safer and more predictable.

# 13. What this means for platform decisions

Decide first which decisions are federated and which stay central, and write them down as rules the platform can check. Give every data product an owner, a domain, a stated purpose, defined output ports and a lifecycle status, and describe it in a machine-readable form aligned with DCAT. Put an ODCS contract on each output port, kept in version control with the code that produces it. Write evolution rules into the contract, add with defaults, deprecate before removing and version breaking changes, and enforce them with compatibility checks in build tools and registries. Run quality checks on the data and publish the results to the catalog. Use lineage and impact analysis before every breaking change. Measure incidents and change lead times, so that the value of products and contracts can be judged on evidence.

For federated query and data products, Bassirah works with Starburst as a Strategic Technology Partner; Starburst Enterprise documents data products with domains, owners, a lifecycle and definitions managed as code, and contracts can be checked alongside them with open tools, since contract checks are not among the features documented on the pages reviewed. For metadata, data products, lineage and impact analysis, Bassirah provides independent support for DataHub Core, the open-source edition of DataHub; this support is Bassirah's own and is not vendor support from DataHub's maintainers. DataHub Core records data products and contracts, while scheduled contract assertions are documented for DataHub Cloud. Other products, projects and specifications named in this paper are ecosystem references. In each case, validate the design against your own data, organization and versions before relying on it.

Paper 4 turns to data sovereignty and cross-border compliance, where the location and jurisdiction of each product and its consumers become part of what must be governed.

## References

1. Foidl, H., Golendukhina, V., Ramler, R., and Felderer, M. (2024). Data pipeline quality: Influencing factors, root causes of data-related issues, and processing problem areas for developers. *Journal of Systems and Software*, 207, 111855. (<https://doi.org/10.1016/j.jss.2023.111855>)
2. Scherzinger, S. and Sidortschuck, S. (2020). An Empirical Study on the Design and Evolution of NoSQL Database Schemas. In *Conceptual Modeling, ER 2020, Lecture Notes in Computer Science 12400*, Springer, pp. 441–455. ([https://doi.org/10.1007/978-3-030-62522-1\\_33](https://doi.org/10.1007/978-3-030-62522-1_33))
3. Dehghani, Z. (2019). How to Move Beyond a Monolithic Data Lake to a Distributed Data Mesh. *martinfowler.com*, article published 20 May 2019. (<https://martinfowler.com/articles/data-monolith-to-mesh.html>)
4. Dehghani, Z. (2020). Data Mesh Principles and Logical Architecture. *martinfowler.com*, article published 3 December 2020. (<https://martinfowler.com/articles/data-mesh-principles.html>)
5. Hasan, M. R., & Legner, C. (2023). Understanding Data Products: Motivations, Definition, and Categories. *Proceedings of the European Conference on Information Systems (ECIS 2023)*, Kristiansand, Research Paper 229. ([https://aisel.aisnet.org/ecis2023\\_rp/229/](https://aisel.aisnet.org/ecis2023_rp/229/))
6. Blohm, I., Wortmann, F., Legner, C., & Köbler, F. (2024). Data products, data mesh, and data fabric: New paradigm(s) for data and analytics? *Business & Information Systems Engineering*, 66(5), 643–652. (<https://doi.org/10.1007/s12599-024-00876-5>)
7. Dehghani, Z. (2022). *Data Mesh: Delivering Data-Driven Value at Scale*. O'Reilly Media, 270 pp., ISBN 978-1-492-09239-1. (<https://openlibrary.org/isbn/9781492092391>)
8. Machado, I. A., Costa, C., & Santos, M. Y. (2022). Data Mesh: Concepts and Principles of a Paradigm Shift in Data Architectures. *Procedia Computer Science (Elsevier)*, 196, 263–271. (<https://doi.org/10.1016/j.procs.2021.12.013>)
9. Goedegebuure, A., Kumara, I., Driessen, S., Van Den Heuvel, W.-J., Monsieur, G., Tamburri, D. A., & Di Nucci, D. (2024). Data Mesh: A Systematic Gray Literature Review. *ACM Computing Surveys*, 57(1), 1–36 (published online 7 Oct 2024; issue dated Jan 2025). (<https://doi.org/10.1145/3687301>)
10. Priebe, T., Neumaier, S., & Markus, S. (2021). Finding Your Way Through the Jungle of Big Data Architectures. *2021 IEEE International Conference on Big Data (Big Data)*, 5994–5996. (<https://doi.org/10.1109/BigData52589.2021.9671862>)
11. Trattnig, S., Bauer, J. C., Geng, P., Tanz, L., Heuss, L., & Daub, R. (2026). Scalable data fabric for smart manufacturing: A connector-based approach to industrial data integration. *Computers in Industry*, 182, Article 104561 (online 2 Oct 2026). (<https://doi.org/10.1016/j.compind.2026.104561>)
12. Hasan, M. R., Finkel, B., & Legner, C. (2026). The Data Product Canvas: Designing Data Products for Sustained Value From Enterprise Data. *Information Systems Journal*, 36(1), 144–158 (published online 21 June 2025). (<https://doi.org/10.1111/isj.12603>)
13. Eichler, R., Gröger, C., Hoos, E., Stach, C., Schwarz, H., & Mitschang, B. (2023). Introducing the enterprise data marketplace: a platform for democratizing company data. *Journal of*

- Big Data, 10(1), Article 173 (published 24 Nov 2023; open access).  
(<https://doi.org/10.1186/s40537-023-00843-z>)
14. Wilkinson, M. D., et al. (2016). The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data*, 3, Article 160018. (<https://doi.org/10.1038/sdata.2016.18>)
  15. Gieß, A., & Hutterer, A. (2025). The Future of Data Management: A Delimitation of Data Platforms, Data Spaces, Data Meshes, and Data Fabrics. *Information Systems and e-Business Management*, 23(4), 971–997. (<https://doi.org/10.1007/s10257-025-00707-4>)
  16. Bode, J., Kühn, N., Kreuzberger, D., & Holtmann, C. (2024). Toward Avoiding the Data Mess: Industry Insights From Data Mesh Implementations. *IEEE Access*, 12, 95402–95416. (<https://doi.org/10.1109/ACCESS.2024.3417291>)
  17. Joshi, D., Pratik, S., & Rao, M. P. (2021). Data Governance in Data Mesh Infrastructures: The Saxo Bank Case Study. *Proceedings of the International Conference on Electronic Business (ICEB 2021)*, Nanjing, paper 52. (<https://aisel.aisnet.org/iceb2021/52/>)
  18. Vestues, K., Hanssen, G. K., Mikalsen, M., Buan, T. A., & Conboy, K. (2022). Agile Data Management in NAV: A Case Study. In *Agile Processes in Software Engineering and Extreme Programming (XP 2022)*, *Lecture Notes in Business Information Processing*, Springer, 220–235. ([https://doi.org/10.1007/978-3-031-08169-9\\_14](https://doi.org/10.1007/978-3-031-08169-9_14))
  19. Wider, A., Verma, S., & Akhtar, A. (2023). Decentralized Data Governance as Part of a Data Mesh Platform: Concepts and Approaches. *2023 IEEE International Conference on Web Services (ICWS)*, 746–754. (<https://doi.org/10.1109/ICWS60048.2023.00101>)
  20. van Eijk, T., Kumara, I., Di Nucci, D., Tamburri, D. A., & van den Heuvel, W.-J. (2024). Architectural Design Decisions for Self-Serve Data Platforms in Data Meshes. *2024 IEEE 21st International Conference on Software Architecture (ICSA)*, 135–145. (<https://doi.org/10.1109/ICSA59870.2024.00021>)
  21. W3C — Albertoni, R., Browning, D., Cox, S. J. D., Gonzalez Beltran, A., Perego, A., & Winstanley, P. (Eds.) (2024). *Data Catalog Vocabulary (DCAT) - Version 3*. W3C Recommendation, 22 August 2024. (<https://www.w3.org/TR/2024/REC-vocab-dcat-3-20240822/>)
  22. Object Management Group (OMG) (2026). *Data Products Ontology (DPROD)*, Version 1.0 beta 2. OMG specification, September 2026, doc. ptc/26-05-40 (OWL ontology ptc/26-06-29; SHACL ptc/26-06-30). (<https://www.omg.org/spec/DPROD>)
  23. Bitol (LF AI & Data Foundation) (2026). *Open Data Product Standard (ODPS)*, v1.1.0. Specification, GitHub release v1.1.0 dated 8 Sep 2026; Apache 2.0. (<https://bitol-io.github.io/open-data-product-standard/latest/home>)
  24. Open Data Product Initiative / Open Data Products (a Series of LF Projects) (2025–2026). *Open Data Product Specification (ODPS)*, version 4.1. Specification, Apache 2.0. (<https://opendataproduts.org/v4.1/>)
  25. Open Data Mesh Initiative / Quantyca s.r.l. (2024). *Data Product Descriptor Specification (DPDS)*, Version 1.0.0. Official specification, accessed October 2026. (<https://dpds.opendatamesh.org/specifications/dpds/1.0.0/> ; repository <https://github.com/opendatamesh-initiative/odm-specification-dpdescriptor>)
  26. Starburst Data, Inc. (2026). *Data products overview; Manage data products* (Starburst Enterprise 483-e STS documentation; latest LTS 482-e). Official documentation, accessed October 2026. (<https://docs.starburst.io/latest/data-products/> ; <https://docs.starburst.io/latest/data-products/manage-data-products.html>)

27. DataHub project (2026). Data Products. Official documentation, accessed October 2026. (<https://docs.datahub.com/docs/dataproducts>)
28. Atlan (2026). Data Products (concepts) and Create data products (how-to). Official documentation, accessed October 2026. (<https://docs.atlan.com/product/capabilities/data-products/concepts/what-are-data-products> ; <https://docs.atlan.com/product/capabilities/data-products/how-tos/create-data-products>)
29. Jones, A. (2023). Driving Data Quality with Data Contracts: A Comprehensive Guide to Building Reliable, Trusted and Effective Data Platforms. Packt Publishing, Birmingham, ISBN 978-1-83763-500-9 (print). (<https://openlibrary.org/isbn/9781837635009>)
30. Freeman, M. and Sanderson, C. (2025). Data Contracts: Developing Production-Grade Pipelines at Scale. O'Reilly Media, ISBN 978-1-0981-5763-0 (print; ebook 978-1-0981-5759-3). (<https://openlibrary.org/isbn/9781098157630>)
31. Wasser, J., Kumara, I., Monsieur, G., Van Den Heuvel, W.-J., and Tamburri, D. A. (2025). Data Contracts in Data Mesh: A Systematic Gray Literature Review. In Business Modeling and Software Design (BMSD 2025), Lecture Notes in Business Information Processing, Springer, Cham, pp. 21–38. ([https://doi.org/10.1007/978-3-031-98033-6\\_2](https://doi.org/10.1007/978-3-031-98033-6_2))
32. Polyzotis, N., Zinkevich, M., Roy, S., Breck, E., and Whang, S. (2019). Data Validation for Machine Learning. Proceedings of Machine Learning and Systems 1 (MLSys 2019). ([https://proceedings.mlsys.org/paper\\_files/paper/2019/hash/928f1160e52192e3e0017fb63ab65391-Abstract.html](https://proceedings.mlsys.org/paper_files/paper/2019/hash/928f1160e52192e3e0017fb63ab65391-Abstract.html))
33. Schelter, S., Lange, D., Schmidt, P., Celikel, M., Biessmann, F., and Grafberger, A. (2018). Automating Large-Scale Data Quality Verification. Proceedings of the VLDB Endowment, 11(12), 1781–1794. (<https://doi.org/10.14778/3229863.3229867>)
34. Lamothe, M., Guéhéneuc, Y.-G., and Shang, W. (2021). A Systematic Review of API Evolution Literature. ACM Computing Surveys, 54(8), 1–36. (<https://doi.org/10.1145/3470133>)
35. Bitol project, LF AI & Data Foundation (2026). Open Data Contract Standard (ODCS), v3.2.0. Official specification, accessed October 2026. (<https://bitol-io.github.io/open-data-contract-standard/latest/> ; repository <https://github.com/bitol-io/open-data-contract-standard>)
36. Bitol / AIDA User Group (2024). Bitol project overview and incubation announcement. Bitol project website, last updated 2024-10-09, accessed October 2026. (<https://bitol.io/?p=79>)
37. Bitol project (2026). Three Years, 59 RFCs, One Vote: Bitol Graduates. Bitol project blog post, last updated 2026-07-28, accessed October 2026. (<https://bitol.io/bitol-graduates/>)
38. Bitol project (2026). Data Contracts & Data Products for AI (release announcement of ODCS v3.2.0 and ODPS v1.1.0). Bitol project blog post, last updated 2026-09-08, accessed October 2026. (<https://bitol.io/data-contracts-data-products-for-ai/>)
39. Christ, J., Harrer, S., et al. / datacontract.com (2023–2026). Data Contract Specification, v1.2.1 (deprecated). Official repository, accessed October 2026. (<https://github.com/datacontract/datacontract-specification>)
40. Negele, S., Christ, J., and Harrer, S. (creators) / datacontract organization (2026). Data Contract CLI. Official documentation and repository, accessed October 2026. (<https://github.com/datacontract/datacontract-cli> ; <https://cli.datacontract.com/>)
41. The Apache Software Foundation (2024). Apache Avro Specification, version 1.12.0, section "Schema Resolution". Official specification, accessed October 2026. (<https://avro.apache.org/docs/1.12.0/specification/>)

42. Google / Protocol Buffers project (2026). Language Guide (proto 3), section "Updating A Message Type". Official documentation, accessed October 2026.  
(<https://protobuf.dev/programming-guides/proto3/>)
43. Confluent, Inc. (2026). Schema Evolution and Compatibility for Schema Registry on Confluent Platform. Official documentation, accessed October 2026.  
(<https://docs.confluent.io/platform/current/schema-registry/fundamentals/schema-evolution.html>)
44. dbt Labs (2026). Model versions. Official documentation, accessed October 2026.  
(<https://docs.getdbt.com/docs/mesh/govern/model-versions>)
45. dbt Labs (2026). Model contracts. Official documentation, accessed October 2026.  
(<https://docs.getdbt.com/docs/mesh/govern/model-contracts>)
46. Confluent, Inc. (2026). Data Contracts for Schema Registry on Confluent Platform. Official documentation, accessed October 2026.  
(<https://docs.confluent.io/platform/current/schema-registry/fundamentals/data-contracts.html>)
47. Soda Data (2026). Soda Core (v4) repository README, and Contract Language reference (Soda v4 docs). Official documentation, accessed October 2026.  
(<https://github.com/sodadata/soda-core> ; <https://docs.soda.io/soda-v4/reference/contract-language-reference>)
48. DataHub project (Acryl Data / DataHub, Inc.) (2026). Data Contract (DataHub Cloud "Observe" documentation). Official documentation, accessed October 2026.  
(<https://docs.datahub.com/docs/managed-datahub/observe/data-contract>)
49. DataHub project (2026). Data Contract entity (dataContract), metadata model reference, DataHub 1.8.0 docs. Official documentation, accessed October 2026.  
(<https://docs.datahub.com/docs/generated/metamodel/entities/datacontract>)
50. OpenLineage project, LF AI & Data Foundation (2026). OpenLineage documentation (overview) and Schema Dataset Facet. Official documentation, accessed October 2026.  
(<https://openlineage.io/docs/> ; <https://openlineage.io/docs/spec/facets/dataset-facets/schema>)
51. DataHub project (2026). Lineage Impact Analysis (DataHub 1.8.0 docs). Official documentation, accessed October 2026. (<https://docs.datahub.com/docs/act-on-metadata/impact-analysis>)

Product and project names are trademarks of their respective owners. Their mention describes publicly documented technology and does not imply a business relationship, except where Bassirah names a Strategic Technology Partner.