WHITE PAPER

Data Mesh

By David Jerrim, Senior Director, Teradata
and Martin Willcox, Vice President of Technology, Teradata

08.21 / DATA MESH / WHITE PAPER

teradata.

Table of Contents

2 Introduction

3 Aligning data products with real-world business requirements

4 Understanding the characteristics of great data products

4 Infrastructure is not the real challenge

5 Six features of successful approaches to Data Mesh

10 Architectural considerations and deployment best-practices

13 Conclusions

14 About Teradata

Introduction.

There is clear value in knowledge workers being able to access, combine and share data from across the enterprise - but data integration can be complex, time-consuming and requires that stakeholders from across the organization come together. The Data Mesh concept proposed by Dehghani[1] proposes a divide and conquer approach to the delivery of data and analytic products by aligning implementation with domain-driven design principles and patterns.

We are enthusiastic about the Data Mesh concept because we observe that there are essentially three strategies for successfully delivering data and analytic products faster in large and complex organizations:

1. Intelligent use of decomposition, so that large and complex products can be created in parallel by multiple, loosely coupled development groups;
2. The use of agile, business-led development methods and teams to eliminate unnecessary work;
3. The automation of as much as possible of the necessary development and testing work that remains.

These approaches are not mutually exclusive – and by employing all three strategies, organizations with mature "DataOps" processes and toolsets can build complex data products in 4–6-week sprints. This is true even when those data products require the acquisition and integration of multiple datasets.

Whilst dividing large problem spaces into smaller, more tractable components is fundamental to success, the unthinking application of "divide and conquer" approaches to data product development is creating a new generation of overlapping, redundant data silos in some organizations.

This risks creating more technical debt precisely when organizations seek the rapid rollout of digital transformation programmes and increased agility.

1 https://martinfowler.com/articles/data-monolith-to-mesh.html

teradata.

It is also the case that despite several decades of intensive academic research, distributed query optimization is still relatively complex and unproven at scale – and that improvements in the performance of multi-core CPUs continue to outpace increases in performance of network and storage sub-systems. Whilst development of usable and useful data products is invariably business-led, understanding and respecting these engineering fundamentals when architecting and designing domains remains critical to success.

Finally, although "Data is the new oil" has become a cliché, when Clive Humby coined the phrase in 2006 he was also pointing out that raw data – the crude oil in the analogy – must be refined before high-value data products are created. Raw, un-sessionized weblogs are, by themselves, neither terribly interesting nor remotely comprehensible to most business users. However, when the raw data have been refined through the removal of the web-bot traffic and the identification of user sessions, the resulting data are a powerful predictor of customer intent – let's call this 'diesel fuel.' When the sessionized web data are combined with interaction data across channels and re-socialized, we have even more powerful predictors – let's call these 'gasoline.' And when these behavioral data are combined with customer, transaction history and demographic data, yet more powerful predictors – 'kerosene' or 'jet fuel' can be created.

Successful data-driven organizations ensure that data products like these can be discovered and connected so that the jet-fuel that powers their digital transformation initiatives can be created quickly and efficiently, and so that complex value chains can be optimized.

As large enterprises operating across multiple geographies continue to embrace cloud deployment models and multiple service providers, we believe that what we term "the connected data warehouse" model will be fundamental to successful Data Mesh implementation. Co-location of multiple schemas aligned to specific business domains within a single, scalable database instance provides a natural platform for at-scale Data Mesh deployment, with lightweight governance processes providing interoperability.

The remainder of this paper outlines practical steps to deploying the Data Mesh concept as an effective foundation for enterprise analytics.

Aligning data products with real-world business requirements

The development of data and analytic products is inherently complex, for at least three reasons:

1. Data products often require that data is reused for purposes that were not foreseen when the processes that generate it were created – necessitating complex data transformation, cleansing and integration;
2. Requirements are often ambiguous and incompletely defined at the start of the project – and are frequently fluid thereafter;
3. Integrating analytic insights into business processes demands that complex trade-offs are revealed, understood and assessed.

For example, we may be able to improve the predictive accuracy of a fraud detection model by training it on a larger set of features, but at the cost of increased run-times when the model is scored in production. An increase in decision latency from 150ms to 200ms might be an acceptable price to pay for a 20% increase in the lift of a fraud detection model – or it might not. But we are unlikely to know at the start of the project whether an improvement is possible or not – and even less likely to be able to quantify the response time "cost" or the lift "benefit" so that one can be weighed against the other.

"We believe that what we term "the connected data warehouse" model will be fundamental to successful data mesh implementation."

Agile, incremental approaches to the development of data and analytic platforms and products have proven

to be extremely successful in addressing these issues. Embedding business stakeholders in the development process ensures what gets built is what is needed to change the business. Not only is a "minimum viable product" lens critical to avoiding unnecessary development work, but it also reduces testing and maintenance effort. Time-to-market is reduced, and the accumulation of technical debt is slowed.

It is also the case that data products come in a variety of shapes and sizes, ranging from lightly curated source-oriented extract files to denormalized star schemas optimized for slice-and-dice reporting. Good data products address a specific requirement at a specific point in-time; great data products are re-usable and extensible. Successful organizations therefore anticipate that data products will need to be adapted as business requirements evolve, rather than remaining static over the course of their lifetime. Managing the data product development lifecycle effectively requires that data products are designed to support modularity and abstraction, and are packaged with appropriate metadata describing provenance and lineage. Our desire to create "minimum viable products" should not lead us to over-rotate on minimum requirements at the expense of medium-term viability.

Understanding the characteristics of great data products

The principal motivation for moving to Data Mesh based architectures is the desire to deliver high-quality data products more quickly and at-scale. Understanding the characteristics of those data products can help organizations to make intelligent technology choices. Large-scale, enterprise analytics typically share one or more of the following characteristics:

- Stateful processing of historical transaction, interaction and event data;
- Complex processing that requires multiple datasets to be combined in order that sophisticated measured can be derived;
- Repeated execution against data that are continuously changing, so that static caching strategies may have limited value;
- Embedded deployment in mission-critical business processes, so that eventual consistency models may be inappropriate.

Opportunistic technology vendors are rushing to jump on the Data Mesh bandwagon by claiming that their virtualization, federation, and even BI application technologies represent "magic middleware" that will enable data to be discovered, relationships to be inferred, and complex joins across distributed datasets to be executed at-scale.

These claims should be treated with extreme scepticism, especially for use-cases where complex processing and high levels of concurrency are concerned. In this regard, note that even at the low-end a typical data platform in a global 3,000 organization today often supports 50+ analytic applications and 1 billion queries per annum. Many commentators anticipate increases in query volumes of two orders of magnitude as Machine Learning becomes ubiquitous over the next decade.

Infrastructure is not the real challenge

We note also that some of the discussions about the Data Mesh on professional social media appear to suggest that rapid deployment of containerized infrastructure to support domain initiatives is the critical ingredient in a successful Data Mesh. For us, this misses the point. Provisioning infrastructure was rarely the "long pole in the tent" for the development of sophisticated data products – even when that meant procuring and installing physical infrastructure in the data centre. And as organizations migrate to the cloud, provisioning computing infrastructure is anyway becoming even simpler and (much) quicker.

Wrangling disparate data so that it can be reliably compared and combined remains the long pole in the tent when developing analytic data products – and one of us has argued elsewhere that this issue is becoming more critical as organizations seek to deploy Machine Learning more widely.

Meyer and Madrigal's[2] recent experience in managing COVID data in the US is instructive. As they described in a recent essay in The Atlantic, the initial response of the

Federal Government in the US to the COVID epidemic rested on the assumption that COVID infection data were fundamentally sound. In fact, they were not. Different states were collecting what looked like the same data according to different policies and processes – so that what appeared to be the 'same' data could not in fact be reliably compared. Models fed with bad data made bad predictions – and the result was bad public policy. You may not have to deal with 50 states – but you are fortunate indeed if all of the data from your manufacturing plants is created to the same standards and supplied on the same schedule, if you sell products in the same quantity and using the same identifier that you use to order them, etc. These are not problems that, by themselves, distributed architectures, Kubernetes clusters, and CI/CD development pipelines will solve because they are not technology problems in the first place.

Six features of successful approaches to Data Mesh

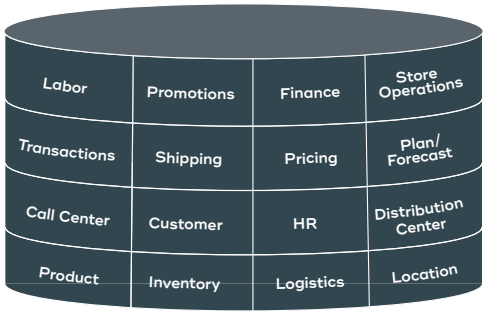
In practice, we observe six critical success factors for reducing time-to-market for the development of new data and analytic products whilst also preserving cross-domain interoperability.

1. Business-driven decomposition; or "subject areas versus domains"

One of the central concepts of domain-driven design that is often misunderstood by organizations pursuing distributed data architectures is "bounded context." Decomposition of a large and complex problem space into a collection of smaller models is not a new idea in science, technology or software engineering – and is central to domain-driven design (DDD). But good DDD also requires precise definition of explicit boundaries and interrelationships between domains. It is tempting to consider the decomposition of the building of complex data products on a subject-area-by-subject-area basis, for example 'event,' 'agreement' or 'product.' Whilst this approach can simplify data re-use, in many organizations it can imply lengthy negotiation and discussion in pursuit of a common understanding of data elements and products that in practice may be shared only infrequently. It often makes more sense to decompose the problem space into domains that are aligned with key business processes and to allow each domain to implement the subject areas applicable to its own activities. This is illustrated below in figure 1.

Subject Areas

- Database concept
- Spans multiple domains
- ELDM overwhelming to business
- Data centric



Domains

- Business unit concept
- Self-contained from business point-of view
- Small-to-medium size schemas
- Business area centric

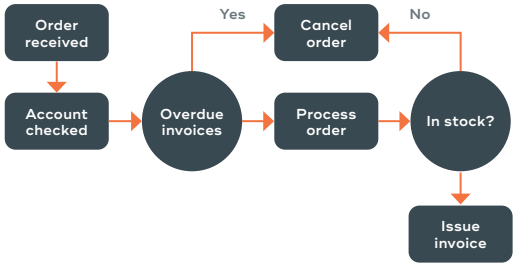


Figure 1: Data subject areas vs business domains

2 https://www.theatlantic.com/science/archive/2021/03/americas-coronavirus-catastrophe-began-with-data/618287/

teradata.

This model works well where each domain is defined with an explicit boundary and all users within that domain are working towards a common purpose and use a consistent business vocabulary. This can be further enhanced using global standards, for example, for the use of surrogation to obfuscate PII data in natural keys. Identification, definition, and sizing of domains is also a critical consideration. If domains are defined with too large a context, agility is sacrificed due to the number of products that must be built and maintained within the domain, and the number of people required to do so. Conversely, where domains are drawn with too narrow a focus, organizations find themselves forever creating additional cross-domain, enterprise teams that risk redundancy and duplication. For us, the "two pizza rule" of agile systems development remains a good guide; if the team building a data product cannot be fed with two extra-large pizzas, it may be too big – and further decomposition should be considered.

All of this implies some degree of management and co-ordination between different development teams. Lightweight governance processes ensure minimum levels of co-ordination across domains, providing bounded context. Published meta-data ensure that data products' high-value data elements, refined at significant cost to the organization, can be discovered and re-used in other contexts.

A simplified retail banking scenario of business-driven decomposition

Consider a simplified retail banking scenario. There are multiple attributes of a mortgage product that are of limited value outside of the mortgage domain. Loan-to-value ratios, the type of survey on which a property valuation was based, and the date of that valuation all represent important information to the mortgage function. But they have limited value in other domains from across the Bank. Decisions about how these data are captured, cleansed, modelled, transformed, managed, and exploited should therefore be delegated exclusively to the mortgage domain. By contrast, information about customer salaries and mortgage debt will be highly relevant in other domains including unsecured loans, credit card, and risk. Ensuring that these data can be shared and combined across domains is not only highly desirable, but probably essential to ensure regulatory compliance in most geographies. And if delinquency codes can be standardized across all the domains that extend credit to customers, then the task of understanding which customers have or are likely to default across multiple product lines will be greatly simplified.

2. Separate schemas by domain to provide agility

One of the primary advantages of embracing domain-driven design is agility: loosely coupled teams working more-or-less independently, and each focusing within their specific areas of business expertise, are able to deliver data products in parallel.

Our recommended approach to implementation of Data Mesh based architectures is to create separate schemas for each domain. Responsibility for data stewardship, data modeling, and population of the schema content is owned by experts with business knowledge about the specific domain under construction. This approach removes many of the bottlenecks associated with attempting to implement a single, centralized consolidation of all enterprise data into a single schema. The domain-oriented schemas provide a collection of data products aligned to areas of business focus within the enterprise. In our simplified retail bank scenario, for example, the mortgage domain may have a legitimate and urgent requirement to create a new data product to understand and measure the impact of the COVID-19 pandemic on the demand for larger suburban properties. At a minimum, this new data product will probably require the roll-up of mortgage product sales by a new and different geographical hierarchy from that used by the rest of the organization.

A domain-aligned development process and schema makes this possible without lengthy discussion and negotiation across the rest of the organization, so long as interoperability standards that also enable total sales of loan products to be rolled-up according to corporate reporting hierarchies exist and are respected.

3. Integration across domains

Many business outcomes can be optimized in the context of a single domain. Many end-to-end business process optimization opportunities, however, require data to be combined across geographical, functional and domain boundaries and these analytic use-cases often drive disproportionately high business value. Organizations need to have an explicit strategy for enabling cross-domain sharing that includes:

- Understanding, defining and enforcing the minimum set of PK / FK relationships required to join and compare data across different domains reliably and accurately.
- Defining appropriate business, technical and operational meta-data that enables data and data products to be discovered and re-used.
- Appropriate Master Data Management that ensures critical attributes, frequently reused and shared across multiple domains, are consistently defined and updated.
- A role-based access control policy and framework that ensures data are accessed and shared appropriately, internally and externally.

4. Support for enterprise data products

Enterprise data products present a multi-domain view of aggregated data products or encapsulate a common enterprise standard. They support optimization of end-to-end business processes, such as customer lifetime value, demand-driven forecasting, and network planning. They are often cross-functional by design, typically require the aggregation of multiple sources of data – and often have value across multiple use-cases and applications. Consequently, they will be frequently reused across multiple domains, as illustrated in figure 2.

Many organizations continue to strive to deliver a 360-degree view of customers and operations to support cross business activities. For a telecommunications provider this would include understanding a customer's recent network experience in terms of streaming behavior, coverage areas, mobility – but also their value to the organization in terms of product subscriptions, out of bundle charges and likelihood to churn. Factor in behavioral indicators regarding channel interaction, sentiment and changes to calling circles and you have a requirement to be able to build and manage a data product that is sourced from multiple domains.

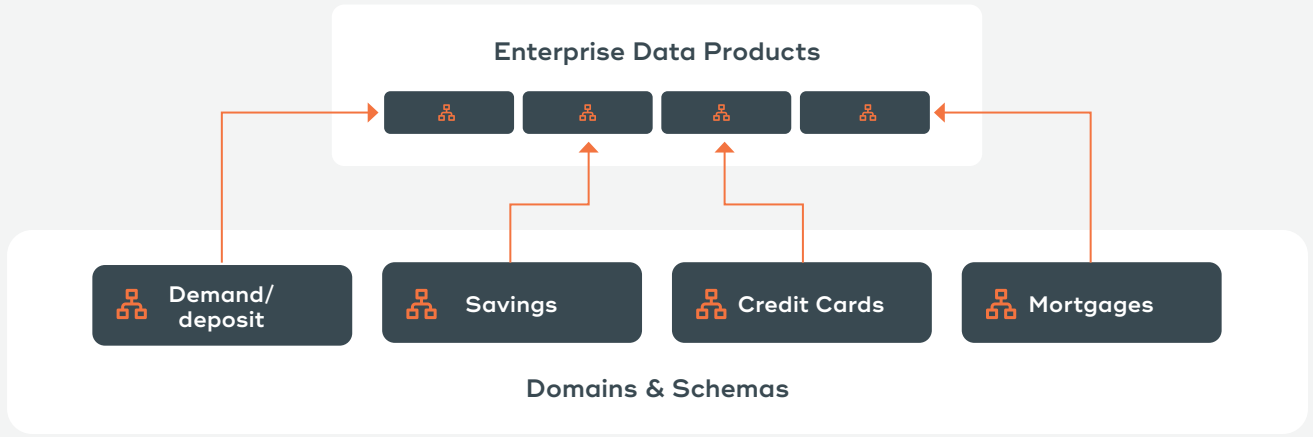


Figure 2: A multi-domain view of aggregated data products

teradata.

Re-use is therefore about much more than merely avoiding the creation of 99 different versions of a nearly identical report. Ultimately, it is about creating layered data architectures that enable high-value data elements to be discovered and re-used to efficiently create multiple data products that support multiple business processes. Interoperability across the domains requires the definition of consistent primary and foreign key relationships and global standards for data typing, naming conventions, and quality metrics.

Layered data architectures that enable highly refined, cross-domain data products to be built from the products created by "downstream" domains can introduce dependencies. But the alternative is that multiple domains are forced to acquire the same data and to create redundant pipelines and overlapping data products. This is not just expensive and error prone, it also creating significant technical debt that will act as a serious drag on innovation and digital transformation programmes in the near future. Effective re-use of data products created in different domains can slash the time-to-market for the development of these enterprise data products whilst also improving quality and consistency, which is why successful organizations place such a high premium on ensuring that data products can be discovered and re-used.

In theory – and assuming adequate PK / FK relationships have been defined – it is possible to implement a "union" operator across separate business domains to get an enterprise view of the data. In our experience however, joining data across LAN and WAN segments with virtualization technologies seldom scales or performs well for complex workloads, with exponential degradation not uncommon as the number of federated systems increases. Instead, it will often be appropriate to create enterprise domains to support the realization of these enterprise data products with the active support and collaboration of domain product owners.

Robust governance and agile, incremental approaches to delivery can co-exist. Where they do, combined with the use of appropriate automation tools and "DataOps" processes, they often lead to an order of magnitude improvement in the time taken to acquire and integrate data and to deliver complex, enterprise data products.

5. Supertypes and subtypes

As we have already discussed, to deliver enterprise data products successfully and efficiently we need the domain teams concerned to be able to reliably combine and aggregate data across multiple domains. In the banking scenario described earlier it would be useful to have an enterprise schema to capture information about customers accounts across all the products they have with the bank.

One approach to achieve this would be to create a "Supertype" account enterprise data product that is populated from across all of the domains. This data product contains attributes that are common across all the domains. Each domain then manages its own subtype account data product with additional attributes specific to the business domain data product. This approach drives a degree of consistency across domains since primary keys to support the join back to the supertype table must be enforced, but also allows for flexibility for business area domains to extend the subtype data product as required. This is illustrated in figure 3 below:

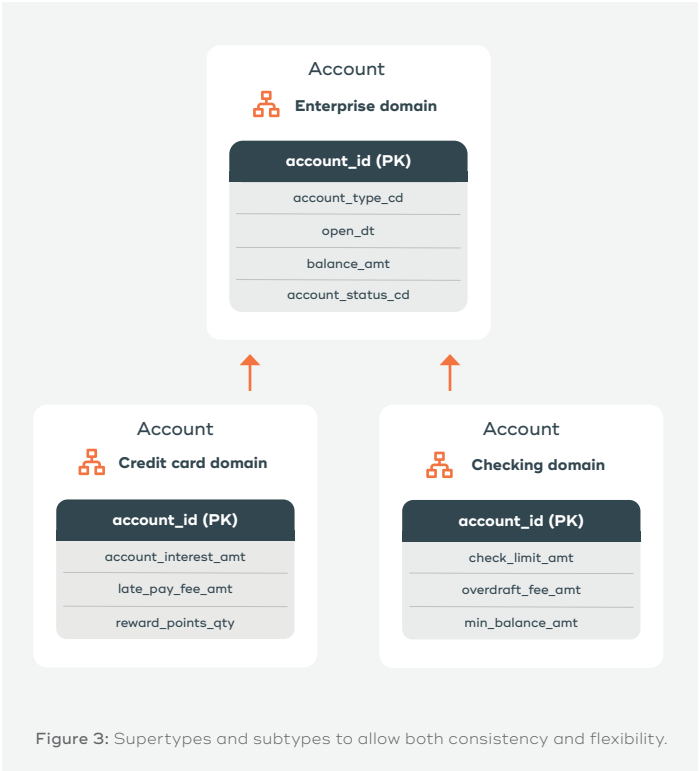


Figure 3: Supertypes and subtypes to allow both consistency and flexibility.

teradata.

Options for the actual implementation of enterprise data product needs careful consideration to take into account, amongst other factors, frequency of access, location of data products, and performance expectations. An enterprise domain could provide a consolidated data product by processing each domain's individual account data products either through virtualization or copying of data out of the domain. However, in the case of a database hosted data product, virtualized SQL UNION ALL across several data platforms may not perform as desired when taking into account factors such as network bandwidth. In this case, it may make more sense to have (denormalized) copies of the supertype information duplicated into a business domains data product, especially if file-based access is relevant. This consideration is an exercise for physical design optimization and will need to be considered for each potential enterprise data product.

Customer and prospect information embedded in each business domain almost always increases in value when promoted to an enterprise domain to facilitate a customer 360 view for enterprise marketing, risk, and other analytics. It is important that in our desire to embrace increased productivity in analytic teams we do not back-slide to the bad old days of the 80s and 90s, when siloed information systems prevented many B2C organizations from treating customers holistically.

6. Good governance – and the right standards

One of the hotly debated topics when adopting Data Mesh principles is how much – or how little – governance should be applied centrally, rather than delegated to individual domains.

Digital transformation and modern business initiatives are driving the need for more, not less, integration across domains. Providing the coherent, cross-functional view across operations required by modern businesses requires that data are not merely technically connected, but also that they are semantically linked. The consistency in implementation across domains required to make this happen does not just spontaneously emerge; rather it requires a co-ordinated, business-driven approach to data governance. Furthermore, it is still the case that specialist expertise

is required to successfully leverage data management tools and effectively implement cross-domain data governance. Each domain will need support to develop data quality processes, data structure design, data reconciliation, and other elements of the architecture in a way that not only works for the near-term application and analytics use cases, but also enables scalability and extensibility. Doing this well takes specialized training and more than a little experience.

There is, nevertheless, an important balance to be struck here. Strong governance and standards by themselves do not guarantee success. Precisely because the semantic alignment that underpins data integration is complex and time-consuming, it is important to be selective – both about which data are aligned and integrated and the extent to which they are modeled. Integration costs serious time and effort – and the game is not always worth the candle.

'Engineered' levels of modeling and integration should be deferred until there is a sophisticated understanding of which data will need to be frequently and reliably compared and/or joined with one another. Since this level of understanding exists only rarely when the first few MVP data products are being developed, organizations should take care to avoid over-investing in data modeling and data engineering during the early stages of a new programme or project by adopting a 'Light Integration' approach, like Teradata's LIMA framework.

teradata.

A recent data platform development programme at a large Asian Telco provides a simple example of the application of a lightweight, de-centralized governance model. Multiple development teams at the telco were able to work largely autonomously whilst ensuring that inter-domain relationships and dependencies were modeled and understood. Agile delivery teams building-out the data platform worked in parallel domain-oriented teams. The teams managed cross-domain impacts via a weekly planning meeting. Each team would add Post-It notes on a shared planning board for each data subject area they were working with for that sprint. Where multiple teams were leveraging the same subject areas the impacts and changes were discussed so that all teams were clear on the changes that were being made. There are many good tools available to those seeking to digitize this process, but this example highlights that collaborative development processes and practices that prioritize knowledge capture and sharing are the real keys to success. By contrast, in our experience far too many organizations lack even basic Wiki pages (or similar) to describe their data platforms, so that the vast majority of organizational knowledge about data products walks out the door at the end of each project and each contract.

Architectural considerations and deployment best-practices

Federating the development of complex data products does not automatically imply the federation of their deployment. In fact, a spectrum of deployment options is available to organizations deploying Data Mesh solutions.

Different strategies are associated with fundamentally different engineering trade-offs, so it is important that organizations frame these choices correctly and are intentional about their decisions.

In general terms, there are three different strategies for deploying schemas within a Data Mesh:

1. Co-location,
2. Connection,
3. Isolation.

These are not mutually exclusive and man real-world implementations use a combination of these approaches.

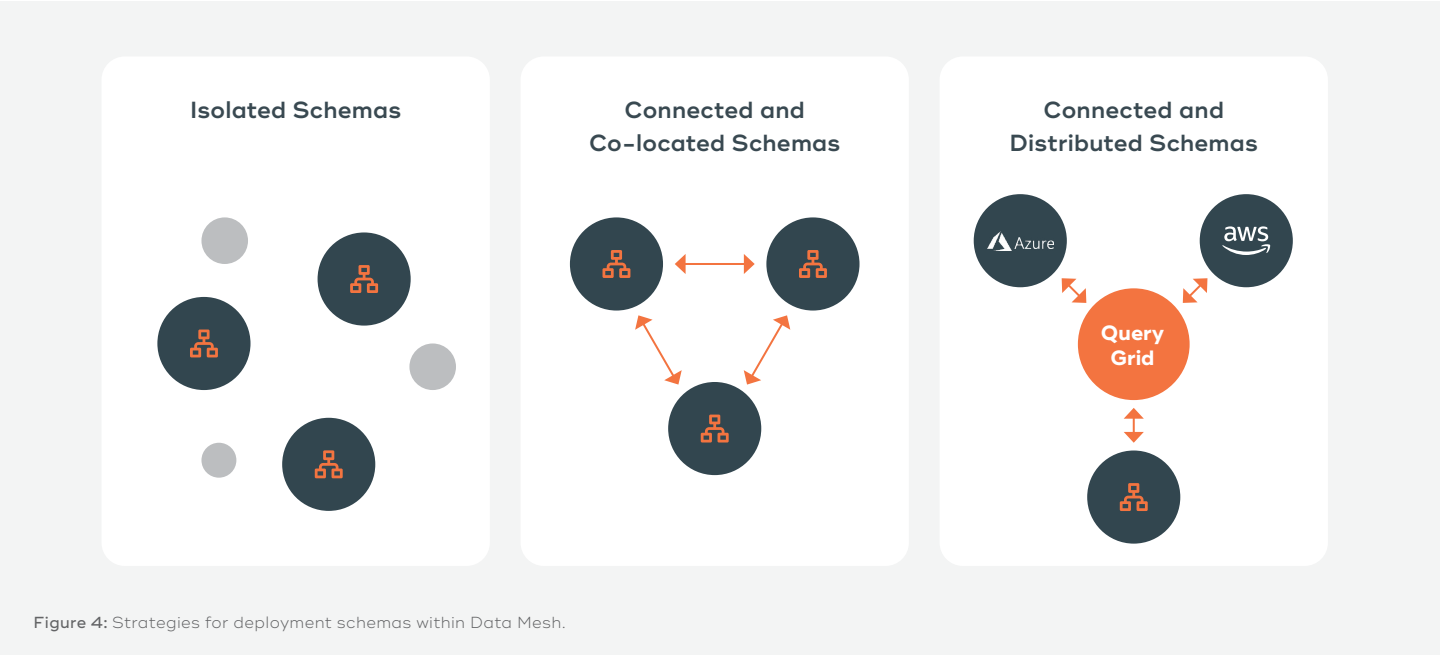


Figure 4: Strategies for deployment schemas within Data Mesh.

teradata.

Co-located schemas

The co-located approach to deployment places domains aligned to different schemes under the management of a single database instance on a common technology platform. Domains are free to manage and evolve their schemas independently according to whatever policies and standards have been agreed centrally.

By itself, co-location does not guarantee interoperability – that must still be designed-in using the approaches outlined earlier in this paper. But deploying on a single platform can have very important performance and scalability advantages, especially for cross-domain workloads that otherwise require data to be assembled across multiple database instances or even multiple clouds. Specifically, co-location can eliminate large-scale data movement operations across relatively slow WAN, LAN, and storage networks and can permit improved query optimization. Even marginal improvements in query optimization can mean very significant improvements in query performance and concurrency.

Where data platforms are supporting billions of queries per annum, the cumulative impact of optimization is substantial improvement in query throughput and reduced total-cost-of-ownership.

Connected schemas

Whilst the physical co-location of schemas generally contributes to better performance and lower TCO when data across multiple domains is combined, there are many cases where it does not make sense to co-locate all schemas under a single database instance. For example, sovereignty laws may require that data created by a business unit within a specific country must remain in that country. For a multi-national company this means that there will be multiple schemas deployed across different geographies – even if the database technology used is the same. There are other reasons why schemas may not be optimally co-located under the umbrella of a single database instance. These include the data gravity created by applications producing data in different clouds, and the use of fit-for-purpose database technologies optimized for particular data types or workloads. Virtualization and federation technologies can still enable cross-domain analytics in the connected scenario. But by themselves these technologies do not guarantee interoperability any more than co-location does.

It is the reconciling and aligning of data so that they can be reliably combined and compared that makes data integration complex. These hard yards must be run whether the data are co-located and joined in a database management system or are physically partitioned across multiple platforms and joined in an application or middleware tier.

Isolated schemas

When using the isolated technique, the implication is that a data product is completely self-contained within a single domain. The schemas used with the isolated technique are usually narrow in scope and service operational reporting requirements rather than enterprise analytics. Isolated domains typically have more autonomy in their deployment – both in data modeling and technology selection. Sometimes isolated domains are proposed based on the need for extreme security (e.g., HR data). However, more often than not, the real reason has to do with politics and the desire for organizational independence. It is rare that truly useful data does not amplify its value when combined with other data, so even where isolation is desirable or necessary consideration should be given to using agreed message formats and pub/sub frameworks and/or APIs to enable the exchange of critical data.

A simplified example

This simplified retail scenario illustrates these different approaches and choices. In a retail business, analyzing sales data enables understanding of product performance; analyzing order data enables understanding of supplier performance; and analyzing inventory data enables costs to be managed. But by putting detailed sales, order, and inventory data together and sharing it with partners and suppliers, Wal-Mart dominated grocery retail in the 90s by creating a demand-driven supply chain that simultaneously improved on-shelf availability, sales, and customer experience whilst also crushing costs.

Amazon similarly dominates retail today by combining purchase data with behavioral data to understand what customers want better than its competitors do. By enabling partners to leverage the platform it has created, it generates even more data about even more customers in a virtuous circle. Data integration is critical to both business models.

Product performance, supplier performance, and cost management domains can build out dedicated data products in parallel, but unless these domains engage in the kind of lightweight collaboration and governance described earlier in this document, development of one of the high-value analytic applications – demand forecasting – will be significantly more complex and more time consuming. Furthermore, because that application will require large volumes of product, sales, and order data to be routinely and frequently combined, co-locating these data products on the same platform to avoid unnecessary data movement - and so improve performance and scalability - is likely to be highly desirable.

Figure 5 shows a grossly simplified schematic representation of a grossly simplified Retail architecture, illustrating the concepts of co-location, connection and isolation in the development of data products. The Sales, Orders, and Inventory products are domain-oriented and developed in parallel, but domain interrelationships are defined and the products are co-located on the same platform so that data can be combined to create a scalable and high-performance Demand Forecast data product.

The Customer Experience data products are also domain-oriented and are also built-out in parallel, but on a separate platform that enables these data products to be run-time connected to improve both the Customer and the Demand Forecast data products; whilst integration is deferred until run-time, it still requires interrelationships to be defined and modeled. A conscious decision is taken to de-couple the Activity Based Costing data product from the Product Performance data product, at the expense of potential inconsistency in Sales reporting and increased technical debt.

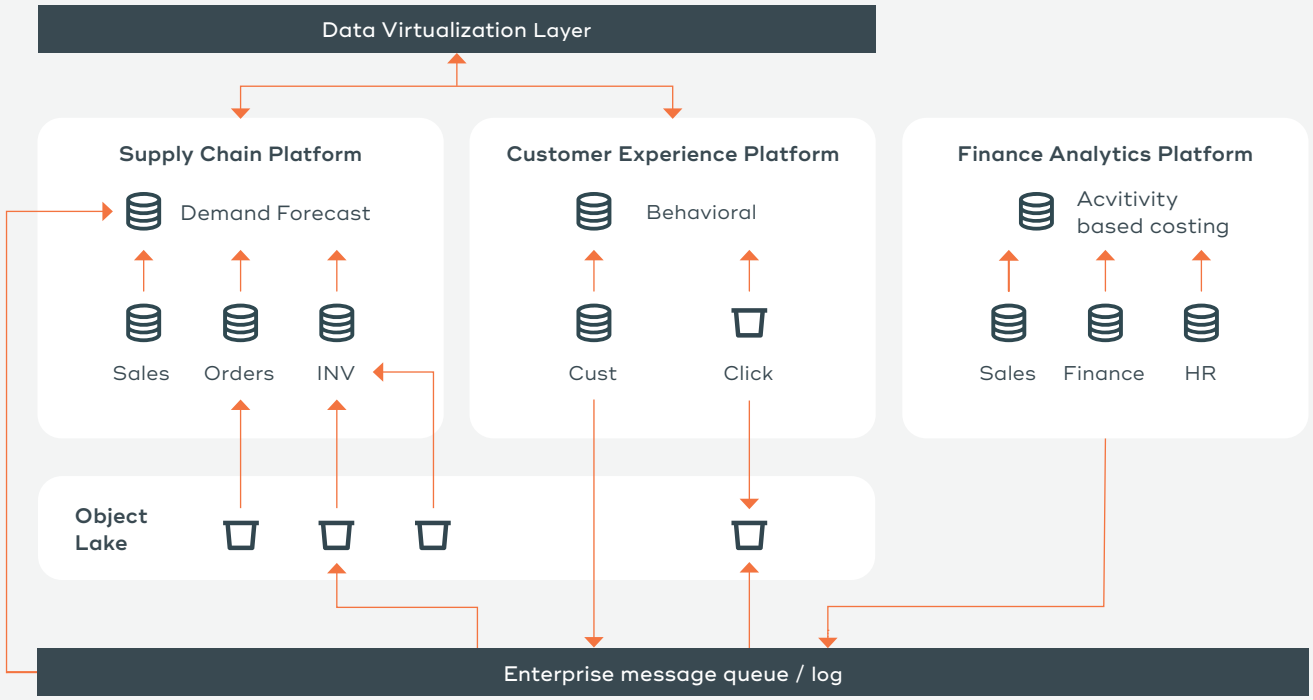


Figure 5: A simplified architecture illustrating co-location, connection, and isolation.

teradata.

Additional domains and workstreams can build out the customer experience analytic products, also in parallel. But it is hard to imagine a useful customer analytics product that does not involve combining customer demographic data with sales data. The Demand Forecast product will be substantially improved if it can also leverage behavioral data from online channels – a leading, rather than a lagging indicator.

For the purposes of our stylized example, let us assume that these data will need to be combined less frequently so that co-location is less important. However, even if this were the case, it is much harder to argue that the products created by these domains should not be linked at all. At a minimum, we should conclude that these domains need to be connected. As we have already discussed, connecting these data products assumes that the domains concerned have defined and implemented the necessary interrelationships and that a scalable, high-performance virtualization layer – like Teradata's QueryGrid framework – has been deployed to enable them to be run-time connected.

For the purposes of our simplified example, let us assume that another set of domain-oriented teams are tasked to build-out finance and HR data and analytic products, including an Activity Based Costing (ABC) data product. The ABC data product will also need to leverage sales data. If the ABC domain acquire this sales data from the product performance data product, they become a consumer of that product – and a dependency is introduced into the architecture. If they acquire that data themselves then they eliminate that dependency at the expense of additional technical debt – because their redundant sales data product and pipeline will need to be developed and maintained – and they also risk introducing inconsistency, so that sales numbers can no longer be reliably compared across the organization. Neither of these choices is automatically right or wrong, but clearly organizations should be intentional about them to avoid inadvertently deploying hard-to-change and-maintain 'accidental' architectures.

Conclusions

We are enthusiastic about the Data Mesh concept because it places intelligent decomposition front-and-centre in the rapid development of complex data products and platforms. Our own experience leads us to conclude that when smart decomposition is combined with agile, incremental development methods and appropriate use of DevOps processes and automation tools, time-to-market for the development of complex data products can in some cases be reduced from months to weeks.

The single most critical success factor for the development of data products remains alignment with funded business initiatives. Agile, incremental, business-led approaches to data and analytic product development matter because they help organizations to focus on delivering high-quality data products that solve a real-world business problem. They therefore avoid unnecessary development and testing work. That adds up to less cost, reduced maintenance overhead, and greater business benefit.

The ultimate "eliminate unnecessary work" play is to extend and reuse existing data products. Designing data products for reuse, creating discoverable data services to enable those data products to be accessed, and ensuring that useful, usable, and searchable "crowd-sourced" business meta-data are available to end-users are all critical to avoiding the constant re-invention of similar, overlapping data products.

Data love data and are frequently exponentially more valuable when they are aligned and combined across domains. Lightweight governance models can provide for the interoperability that is required to optimize end-to-end business processes. Done right, this amounts to another reuse strategy, because it ensures that existing domain data products are leveraged in the creation of aggregate data products. This is a faster and cheaper approach to creating these data products. It is better too, since it reduces the data duplication that can otherwise drive inconsistency, complexity and increased technical debt.

Whilst we believe that development of data products should be federated along domain lines by default, we encourage organizations to proceed more cautiously before federating the deployment of those data products.

In general, both the co-location and connection patterns provide for better performance, scalability, and TCO than does the isolation pattern, with the co-located pattern scaling and performing best of all in the vast majority of circumstances. All three of the deployment strategies have a place – and most large and complex organizations will find that they need to deploy all of them in different parts of the business. Doing so intentionally, and with a clear-eyed understanding of their strengths and weaknesses, is key to avoiding "accidental architectures" that inhibit, rather than enable, change.

Most large enterprises already operate across multiple geographies – and are increasingly leveraging multiple cloud service providers. That makes the connected data warehouse fundamental to at-scale Data Mesh implementation. Within a Cloud Service Provider and within a geography, co-location of multiple schemas aligned to specific business domains within a single, scalable database instance gives the best of two worlds: agility in implementation and high-performance in execution.

About Teradata

Teradata is the connected multi-cloud data platform company. Our enterprise analytics solve business challenges from start to scale. Only Teradata gives you the flexibility to handle the massive and mixed data workloads of the future, today. The Teradata Vantage architecture is cloud native, delivered as-a-service, and built on an open ecosystem. These design features make Vantage the ideal platform to optimize price performance in a multi-cloud environment. Learn more at **Teradata.com.**

About the Authors

By David Jerrim, Senior Director, Teradata
and Martin Willcox, Vice President of Technology, Teradata

The authors would like to acknowledge the contributions of: Jean-Marc Bonnet, Jim Bougatsias, Stephen Brobst, Kevin Lewis and Nathan Green. With grateful thanks to Stephen Brobst for providing the illustrations.

17095 Via Del Campo, San Diego, CA 92127 Teradata.com

teradata.