Executive Summary:

1. **What we are being asked to do**

|  |  |  |
| --- | --- | --- |
| **Business Area** | **Request** | **Purpose/ Use Case** |
| Business A | Onboard all market data to support   current or new pricer  EEE Full Reval  New curves  Credit Vol   In House Curves | Support regs Support Hedging  Enable consistent valuation  Reduce dependency on VVVV |
| Business B | Provide historical time series | Probablity analysis  Enhanced Business B analytics |
| Business C | Onboard business C issued data | Support Liquiity and funding Strategy Build business C curves |
| Business D | On board new Curves |  |
| Audit Support | Provide Audit trail and evidence of data controls | Satisfy Regs and audit scrutinity  Support Explain options |
| Business E | Tech storate issue | Enable downstream to X users |

**1.2 , new roles and responsibility ( FO Data Strats)**

Advanced Analytics and Pre Onboarding Validation

|  |  |  |
| --- | --- | --- |
| **Area** | **New Responsibility** | **Why it matters** |
| Onwership of Team A and B functionality | take over the prepnboarding validation  Leverage team A and B knowledge | aligns data onboarding with … |
| Pre Onboarding Testing | Perform structured validation before onboardng | Prevents poor quality data from entering.. Reduce downstream .. And audit |
| OnGoing Monitoring | maintain continue validation of DQ checks | Support defensible risk modeling |
| Build Data Validation Engine | correlation, pricing consistency, validate behavior during bad period |  |

1. **Business Value from doing so**

|  |  |  |
| --- | --- | --- |
| **Benefit Area** | **Value Delivered** | **Why It Matters** |
| Regs Defensibility | System A is Referenced in reg Artifacts | Reduce risk of and audit findings |
| Strategic Enablement | Enable Onboarding of new products Facicilates metric A and B | Align with business Long term area Position .. As single truth |
| Audit Readiness | Provide Evidence of controls on  Centralized data lineage  Enable Point in time views | Respons to increasing audit |
| Businss Impact | Support Profitability analysis Improve Liquity Management | Directly improve risk manamgent  Enhance team A and B collaboration |

1. **Consequence to InAction**

|  |  |  |
| --- | --- | --- |
| **Consequence to InAction** |  |  |
| **Risk Area** | **Impact if we do nothing** | **why it matters** |
| RRR exposure | Failure to comply with Risk of …in future exams |  |
| Audit Failures | system A is ungovernec | Audit team already flag this as  Work A requires Platform A for Defensibility |
| Strategic misaligment | No Road Map for full migration to system A | Leavs gaps in  Undermines the goal of .. As single source |

**Introduction**

"Good morning, everyone. Thank you all for taking the time to join this discussion. I’m excited to share our vision for the expansion of the MDSOR platform and to outline the resources we need to make this vision a reality."

**The Ask**

"We are here today to secure your support and the necessary resource allocation for the expansion of MDSOR. This expansion is crucial for transforming how we manage and leverage our market data, ensuring that we stay ahead of the curve and remain competitive in this rapidly evolving landscape."

**Why It Matters**

"The importance of this expansion cannot be overstated. By investing in MDSOR, we are setting the stage for a more streamlined, efficient, and reliable market data infrastructure. This means faster onboarding of new products, improved regulatory compliance, and more accurate and timely insights for our teams. In short, MDSOR will be the backbone that supports our data-driven decision-making and innovation."

**The Risks of Inaction**

"If we choose to ignore this opportunity, we risk falling behind our competitors who are already leveraging advanced data solutions. Delays in product launches, inefficiencies in data processing, and potential compliance issues could become significant roadblocks. This is a pivotal moment for us to act and ensure that we are not only keeping up but leading the way.

**What We Are Being Asked to Do**

"In essence, we are being asked to transform how we handle market data from end to end. This means several key things:

1. **Breaking Down Data Silos**: We need to unify our data sources so that all teams have access to consistent, reliable data. This will eliminate inefficiencies and ensure everyone is working with the same, high-quality information.
2. **Creating Curated Data Products**: Instead of simply storing data, we’ll be turning it into curated, validated products that are easy to access and use. This ensures that our analysts, risk managers, and other stakeholders can derive insights quickly without spending time on data wrangling.
3. **Embedding Governance and Quality Control**: We’re implementing strong, embedded governance to ensure data integrity, traceability, and compliance. This means less manual oversight and more automated controls, making the entire process smoother and more reliable.
4. **Delivering Data in the Preferred Format**: We aim to make data easily accessible in whatever format our teams need, ensuring that they can integrate it seamlessly into their workflows and tools.
5. **Adopting a Flexible, Scalable Architecture**: Inspired by the principles of Data Mesh, we’ll ensure that our architecture is modular and adaptable, so we can scale effortlessly as our business grows and as new data needs emerge

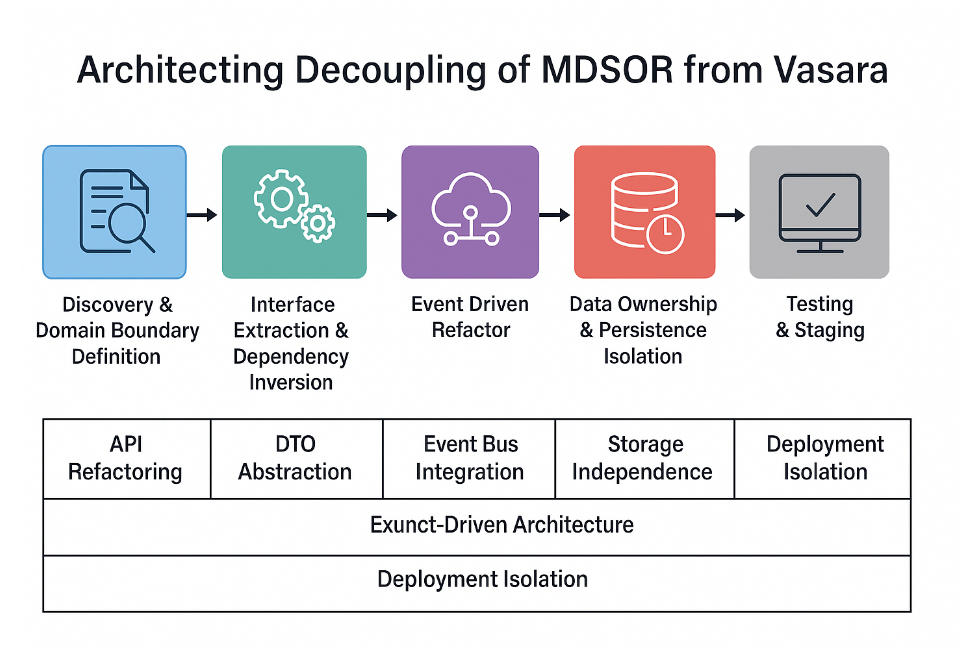
🧠 **Suggested Visuals for Your Next Slides**

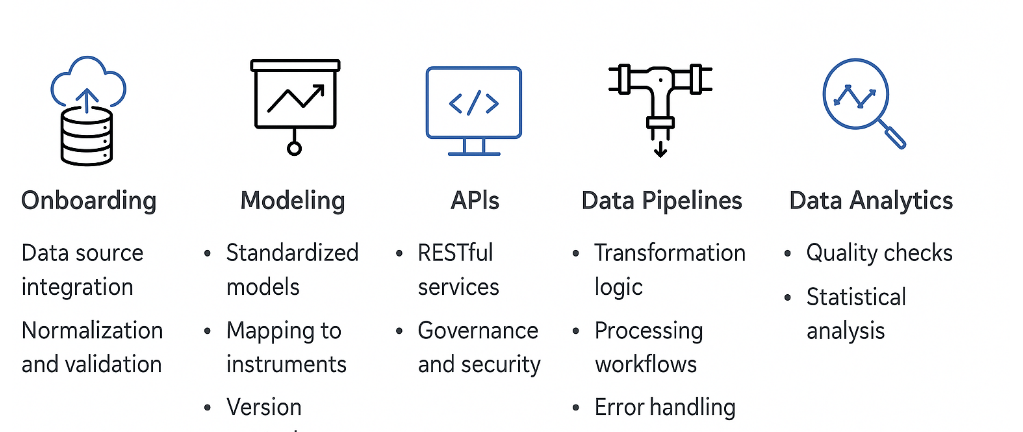
1. **Current State vs Future Vision**
   * Side-by-side flow: current pain (manual, stale, siloed) vs. target (modular, governed, self-service)
2. **Time Series Data Lifecycle**
   * Flow: Ingest → Normalize → Stitch → Validate → Store → Query → Derive → Share
3. **Modern Architecture Blueprint**
   * Layers: Ingestion → Storage → Modeling → Analytics → Access → Governance
4. **Market Data Mesh**
   * Federated domains (IR, FX, EQ, Credit) as producers
   * Central governance plane (Lake Formation or FinSpace)
5. **AWS-Powered Stack Example**
   * Icons of Timestream, Glue, Athena, Redshift, SageMaker, S3, Lambda in modular flow

**⚙️ Technical Capabilities You Want to Build**

| **Area** | **Desired Capability** |
| --- | --- |
| Ingestion | Real-time and batch pipelines with AWS Glue, Kafka, AppFlow |
| Storage | Time series DB (Timestream, or hybrid S3+Iceberg/Delta) |
| Modeling | Bitemporal versioning, flexible risk factor schemas |
| APIs | REST/GraphQL APIs and Python SDK for programmatic access |
| Governance | Fine-grained entitlements, version control, audit |
| Discovery | Self-service tools like FinSpace, Athena, Data Catalog |
| UI/UX | Exploratory dashboards like FLDS or FinSpace |
| Compute | Serverless compute for backfill, analytics, transformations |

**API**





**🔁 Summary of Key Technical Workstreams**

| **Workstream** | **Description** |
| --- | --- |
| API Refactoring | Replace synchronous Vasara calls with interfaces |
| DTO Abstraction | Redefine POJOs and serialization boundaries |
| Event Bus Integration | Move from tightly coupled calls to async events |
| Storage Independence | Own lifecycle, history, and audit trail |
| Contract/Component Tests | Build regression suite to validate independence |
| Deployment Isolation | Standalone build, CI/CD, config, fault isolation |

**🧱 Phase 1: Discovery & Domain Boundary Definition**

1. **Identify coupling points**
   * Static references to Vasara modules, utility classes, shared libraries
   * Direct data access via Vasara-specific service layers
   * Event subscriptions using Vasara-specific event buses (e.g., custom pub-sub)
   * Hardcoded configuration files tied to Vasara namespaces
2. **Perform dependency graph analysis**
   * Use tools like JDepend, ArchUnit or IntelliJ structural analysis
   * Categorize dependencies into:
     + data contract coupling (schemas, DTOs)
     + behavior coupling (synchronous service calls, shared state)
     + environment coupling (classpath/config injections)
3. **Define Bounded Contexts**
   * Apply DDD (Domain-Driven Design) to isolate MDSOR from upstream Vasara domains
   * Tag shared classes that violate SRP or LSP (Single Responsibility Principle / Liskov Substitution)
   * Design clear APIs/interfaces at context boundaries (e.g., RiskFactorService, DataCatalogClient)

**⚙️ Phase 2: Interface Extraction & Dependency Inversion**

1. **Extract interfaces** from all Vasara services MDSOR depends on
   * Functional interfaces if using Java 8+ (for lambda injection support)
   * Example: MarketDataLookup, ShiftAdjustmentPolicy, RiskFactorMapper
   * Create an SPI (Service Provider Interface) module
2. **Apply the Dependency Inversion Principle (DIP)**
   * Inject abstractions using constructor injection (Spring Boot or Guice)
   * Replace direct Vasara imports with service provider bridge modules
3. **Decompose shared objects and value types**
   * Avoid referencing Vasara MarketData, CurvePoint, Namespace, etc. directly
   * Create MDSOR-native POJOs or DTOs
   * Use Jackson MixIns or custom serializers to interop at boundary

🚦 **Phase 3: Event-Driven Refactor**

1. **Decouple synchronous behavior** using domain events
   * Move from function call → return result to emit event → subscribe + handle
   * Examples:
     + MarketDataPublishedEvent
     + SchemaDefinitionUpdatedEvent
     + DerivedCurveComputedEvent
2. **Introduce event bus abstraction**
   * e.g. Reactor (Project Reactor), Akka, Spring ApplicationEventPublisher, or Kafka
   * Enable async processing of onboarding, QA suspecting, derivation flows
3. **Back-pressure, retry, and fault tolerance**
   * Use reactive streams principles or circuit breakers (e.g., Resilience4j, Hystrix)
   * This allows MDSOR to operate independently from failures in Vasara’s downstream compute

**🗃️ Phase 4: Data Ownership & Persistence Isolation**

1. **Move MDSOR data ownership to its own stores**
   * HBase → switch to isolated namespace or TSDB backed by Kafka for ingestion
   * Avoid reusing Vasara-curated state (e.g. derived curves, valuation inputs)
2. **Introduce versioned data model + bitemporal support**
   * Own market data lifecycle: loaded, validated, transformed, derived
   * Manage data revisions without relying on Vasara data promotions
3. **Governance and audit trace**
   * Build independent lineage, access control, and suspect tagging
   * Remove reliance on Vasara's internal QA controls

**Special Considerations for Time Series**

* Data structure must support:
  + asOfDate, effectiveDate, updateTimestamp
  + Roll-forward/roll-back navigation
* Seamless stitching for back-testing, forecasting, scenario analysis
* Time-local indexing: partitioning by asset/date range
* Ability to store both raw and derived views
* Needs dual-access paths:
  + High-speed query for point-in-time analysis (in-memory layer)
  + Long-tail archive for historical backtests (cold S3 + SQL)

🧱 **Modern Patterns to Adopt**

| **Concept** | **Example/Tool** |
| --- | --- |
| Data Mesh | Domain-owned ingestion, productized datasets |
| Federated Query | Athena with connectors (S3, Redshift, RDS, etc.) |
| Data Modeling | Legend, FinSpace vocabularies, asset class taxonomies |
| Real-Time Processing | Kinesis + Lambda, or Kafka + Flink |
| No-Code for Ops | Glue DataBrew, SageMaker Canvas |
| Governance | AWS Lake Formation, FinSpace access rules |
| Machine Learning Ready | SageMaker Feature Store, AutoML, Pipelines |
| Modularity | Microservices (like GS Financial Cloud or JPM Athena) |

**Resourcing (why business)**

No worries! It's definitely important to emphasize that waiting could create more challenges down the line. You can explain that while there are other priorities like PLA and Vasara, investing in MDSOR now ensures that it becomes a robust, independent application capable of supporting all business needs. This proactive approach will prevent larger issues in the future and streamline our operations, ultimately benefiting everyone in the long run.

having resources from the business side ensures that MDSOR is built with a deep understanding of the actual needs and challenges. Without that insight, it’s easy for a project to become misaligned with the real business goals, resulting in delays and less impactful outcomes. By involving people who truly understand the front office, the risk landscape, and the nuances of our market data, we can ensure that MDSOR will be scalable, manageable, and truly aligned with our long-term business strategy. This way, we’re not just creating a short-term fix, but a sustainable solution for the future

. You can emphasize that the resources we need are not just any developers, but highly skilled individuals who have a deep understanding of our business needs and the technical expertise to create scalable, reusable solutions. These individuals will help us design the architecture, identify the right tools, and ensure that MDSOR is future-proof. By having the right talent from the start, we can ensure that the platform not only meets our current needs but also evolves with us, reducing the risk of having to constantly hire more people or redo work in the future. Let’s drive this initiative proactively and set ourselves up for long-term success

**Potential Questions and Concerns:**

1. **Why can’t we leverage existing tech resources instead of hiring new ones?**
   * **Response:** While our current tech teams are incredibly capable, they’re already fully committed to other critical initiatives like PLA and Vasara. The expertise we need for MDSOR is highly specialized, combining deep business insight with advanced technical skills. By bringing in dedicated resources, we ensure that MDSOR gets the focused attention it needs without compromising other projects.
2. **How do we know this investment will pay off in the long run?**
   * **Response:** Investing in MDSOR now means we’re building a future-proof platform. This will streamline data processes, reduce inefficiencies, and improve compliance and risk management. In the long run, this reduces costs, accelerates product launches, and ensures that we remain competitive. Delaying this investment could lead to bigger challenges and higher costs down the line.
3. **What if the technology landscape changes? How adaptable will MDSOR be?**
   * **Response:** One of the key principles in our roadmap is to ensure that MDSOR is built with flexibility and scalability in mind. We’ll adopt a modular architecture that can integrate with new technologies and evolve as the landscape changes. This ensures that MDSOR remains relevant and continues to add value, no matter how the tech environment shifts.
4. **How will we ensure collaboration between business and technology teams?**
   * **Response:** Collaboration is at the heart of this initiative. We’ll have dedicated liaisons from the business side working closely with the tech team to ensure that development is aligned with business needs. Regular check-ins, joint planning sessions, and continuous feedback loops will ensure seamless integration and mutual understanding.

**Decoupling**

"When it comes to decoupling MDSOR from Vasara, we need a strategic, phased approach — one that ensures **resilience, reuse, and independence**. On the Vasara side, we start by isolating the specific market data services that MDSOR currently consumes — pricing engines, instrument mappings, or curve construction logic. These will be exposed through **clean, versioned, and contract-driven APIs**, allowing MDSOR to interface without being embedded within Vasara’s core.

Meanwhile, MDSOR will **build its own ingestion, modeling, validation, and distribution pipelines** — decoupled from Vasara’s deployment cycles and priorities. Think of MDSOR as becoming its own “mini-Fusion”: a domain-agnostic data product platform that **serves pricing, risk, and audit needs across desks**, independently.

This decoupling lets us scale MDSOR to serve not just Vasara-aligned users, but also Risk, Finance, Quants, and even external clients or regulators — **without waiting on upstream code releases or coordination bottlenecks.**

The transformation is not just technical — it’s architectural and operational.  
We’re embracing a **modular, federated data mesh model**: MDSOR becomes the foundation layer where each asset class (IR, FX, EQ, Credit, Structured Products) owns their logic, pipelines, and controls — while leveraging common shared services for ingestion, lineage, QA, and APIs.

So to summarize: this isn’t just about removing tight integration.  
It’s about **unlocking speed, ownership, auditability, and agility** — using Fusion-like principles, AWS-inspired architecture, and Data Mesh thinking to position MDSOR as the future-ready market data backbone.

Excellent — with the **Fusion principles**, **AWS key strategies**, and now the **“From Data Mess to Data Mesh”** guidance from Nicki Watt, you are now fully equipped to present **a transformative vision for MDSOR**.

Below is a **consolidated and enhanced set of MDSOR principles**, structured in plain business language and layered across:

* ✅ AWS Strategy (technical foundation)
* ✅ Fusion (operational & user-centric)
* ✅ Data Mesh (organizational mindset and scalability)

**🔷 MDSOR STRATEGIC PRINCIPLES**

**From Data Mess to Data Mesh**  
(*Powered by AWS-like architecture + Fusion-like delivery + Mesh-inspired ownership*)

**1. Start with a Solid Data Foundation**

*“The engine of insight is clean, connected, governed data.”*  
MDSOR builds a consistent, normalized, and complete dataset as a starting point. This unlocks everything else: valuation, risk, analytics, AI.

* ✅ AWS: Store + integrate + govern
* ✅ Fusion: Clean, ready-to-use datasets
* ✅ Mesh: Foundational prerequisite

**2. Break the Silos: Systems, People, Process**

*“Silos slow down every project, every decision, every audit.”*  
MDSOR unifies fragmented data, decouples teams from tribal knowledge, and builds shared data practices across the bank.

* ✅ AWS: Zero-ETL & federation
* ✅ Fusion: Central API access
* ✅ Mesh: Decentralize usage, not chaos

**3. Deliver Data Products, Not Just Data Dumps**

*“Stop sharing spreadsheets. Start sharing products.”*  
Each desk, team, or domain produces validated, structured, versioned data sets that are queryable, governable, and discoverable.

* ✅ AWS: LakeFormation, Glue, FinSpace
* ✅ Fusion: ESG example — ready-to-use normalized views
* ✅ Mesh: Domain teams expose **data as a product**

**4. Push Quality Upstream (Shift Left)**

*“Normalize and clean at the source — not after the fact.”*  
MDSOR applies mapping, tagging, identifier resolution, and QA as part of the ingestion layer — not passed down to each consumer.

* ✅ AWS: Glue DataBrew, streaming validation
* ✅ Fusion: Unified schema and opinion management
* ✅ Mesh: Data producers own data quality

**5. Enable Self-Service, Not Service Tickets**

*“Let users find, trust, and use data without a helpdesk.”*  
MDSOR includes search, APIs, metadata, dashboards, and SDKs to make data access frictionless and intuitive.

* ✅ AWS: Athena, QuickSight, API endpoints
* ✅ Fusion: Query by intent, not schema
* ✅ Mesh: Self-service platform is essential to scale

**6. Embed Governance, Not Just Control**

*“Every dataset has a contract. Every override has lineage.”*  
Governance isn’t an afterthought. Validation rules, lineage, PII tagging, SLA monitoring — all embedded by design.

* ✅ AWS: LakeFormation, DataZone
* ✅ Fusion: Track data opinions, propagate with audit
* ✅ Mesh: Federated computational governance

**7. Support Domain Ownership + Platform Enablement**

*“Let Credit, IR, EQ, and others own what they know best — without reinventing the wheel.”*  
MDSOR provides shared pipelines, templates, validation frameworks, and tools, while domain teams own their datasets and SLAs.

* ✅ AWS: Shared but modular architectures
* ✅ Fusion: Flexible onboarding with controls
* ✅ Mesh: Decentralized ownership, centralized guardrails

**8. Deliver Data Where It’s Needed, When It’s Needed**

*“Power risk models, pricing engines, and audit queries from one trusted platform.”*  
MDSOR lets users subscribe to data in their format — JSON, CSV, APIs — with consistency and scale.

* ✅ AWS: Redshift, Timestream, Step Functions
* ✅ Fusion: Deliver to any tech of client’s choice
* ✅ Mesh: Data consumers pull data via contracts

**9. From Weeks to Days: Accelerate Every Workflow**

*“Faster onboarding. Faster analysis. Faster compliance.”*  
MDSOR streamlines vendor onboarding, internal data loading, risk factor expansion, and exception handling.

* ✅ AWS: Serverless integration
* ✅ Fusion: From months to days
* ✅ Mesh: Start small, iterate fast

**10. No One-Size-Fits-All: Adapt to Your Context**

*“MDSOR doesn’t force one architecture — it adapts.”*  
We combine lakehouse for storage efficiency, mesh for domain ownership, and modular APIs for flexibility.

* ✅ AWS: Modular, not monolithic
* ✅ Fusion: Elastic and composable
* ✅ Mesh: Hybrid models — lakehouse + federated governance

**11. Make Progress Without a Reorg**

*“You don’t need to boil the ocean to modernize.”*  
Start small. Pick a use case (like VIX, RMBS, FX Spot), build your first domain product, learn, adapt, expand.

* ✅ AWS: Plug-and-play components
* ✅ Fusion: ESG use case first
* ✅ Mesh: Crawl-walk-run model of implementation

**12. Technology Is Only Half the Answer**

*“Data Mesh is not a product — it’s a mindset.”*  
MDSOR requires shifts in culture, incentives, ownership, and process. We’re pairing platform evolution with an operating model redesign.

* ✅ AWS: Well-architected framework
* ✅ Fusion: Business ownership of data usage
* ✅ Mesh: Socio-technical shift (not a software install)

**🧩 Summary for Executives**

**MDSOR isn’t just a data platform — it’s our path out of the data mess.**  
By combining technical strategy (AWS), operational delivery (Fusion), and organizational design (Data Mesh), we will:

* Accelerate product and model onboarding
* Eliminate waste in data wrangling
* Strengthen audit defensibility
* Empower teams to build with confidence
* Scale data usage, not just data storage

Excellent — here is your **complete list of MDSOR Executive Pitch Principles**, now enhanced with **Fusion-style business logic** layered *on top of* the AWS-derived strategic principles. Each is phrased in **plain business terms**, ready for inclusion in your deck, script, or executive one-pager.

**🔧 MDSOR STRATEGIC PRINCIPLES**

**(Blending AWS and Fusion best practices — in your voice)**

**🔹 1. Start with a Solid Data Foundation**

*Everything else depends on it.*

Before you can forecast risk, train AI, or run analytics — the data must be complete, clean, and trusted.  
MDSOR establishes a single, solid foundation for all market data needs.

**🔹 2. Break Down Data, People, and Process Silos**

*Make market data easier to find, understand, and use.*

Market data today is fractured — across files, teams, systems, and knowledge silos.  
MDSOR unifies this into a **shared platform** that brings transparency, consistency, and accessibility to everyone.

**🔹 3. Turn Raw Feeds into Curated Data Products**

*Stop passing around files. Start delivering structured, validated datasets.*

Like Fusion, MDSOR shifts from dumping tables to producing curated data products — modeled, versioned, and aligned with how each desk or function uses them.

**🔹 4. Push Normalization and Cleansing Upstream**

*So downstream users don’t waste time fixing data.*

Today, every team reinvents the same mapping and joins. MDSOR **normalizes on ingest**, applying common structure and identifiers once — not repeatedly by every consumer.

**🔹 5. Deliver Data That’s Ready to Use, Not Just Stored**

*APIs, not file drops. Business-ready formats, not raw tables.*

MDSOR focuses on usability. Users **describe what they need**, and the system delivers it in the format, structure, and location they prefer — with traceability and context built-in.

**🔹 6. Replace Manual Review with Embedded Data Controls**

*Flag suspect points. Track overrides. Embed governance.*

Instead of relying on email and spreadsheets to manage data issues, MDSOR includes **automated validation rules**, override workflows, and full lineage tracking.

**🔹 7. Give Users Choice, Not Complexity**

*No need to know where the data lives or how to join it.*

With MDSOR, users **don’t need to understand tables or schemas.** They can query by intent (“Give me 10y curves”) and MDSOR assembles the result, validated and delivered on time.

**🔹 8. Enable Fast Discovery, Versioning, and Access**

*No more “where’s the latest?” or “who has access?”*

Data is discoverable via metadata catalog, versioned with history, and **delivered through controlled, governed access** — cutting time-to-discovery and audit burden.

**🔹 9. Move Compute to the Data (Not Data to the Compute)**

*Avoid unnecessary ETL and duplication.*

MDSOR supports **query-in-place** and streaming where possible — enabling real-time pipelines without waiting for batch processes or replication steps.

**🔹 10. Support Federated Ownership with Central Governance**

*Let domains own their data, with platform-wide consistency.*

Each asset class (Credit, IR, EQ, Commodities) owns onboarding and quality. MDSOR provides the **shared tooling, controls, and delivery framework** to make it scalable and safe.

**🔹 11. Reduce Time from Months to Weeks**

*Accelerate onboarding, reduce data remediation, unlock analytics.*

Just like Fusion’s clients, MDSOR users should **spend 90% less time wrangling data** — and more time running models, onboarding products, and responding to regulators.

**🔹 12. Drive Better Decisions, Faster**

*Clean data. Trusted data. Faster time to insight.*

Whether it's valuation explain, regulatory reporting, or backtesting, MDSOR provides **ready-to-use data pipelines** that reduce delays and improve outcomes.

**🔹 13. Adaptable, Modular, and Built for Change**

*Scale fast, plug in new sources, respond to evolving needs.*

MDSOR is modular — built on flexible ingestion, modeling, and delivery components. This allows us to grow, adapt, and integrate **without rebuilding from scratch.**

Would you like me to now:

* Convert this list into an editable **PowerPoint slide** or set of slides?
* Write a **speech-ready version** to open your C-suite meeting?
* Create a **1-page summary** with visuals to hand out?

Just say the word — I’ll prepare it immediately.

Thank you — I’ve now reviewed the **Fusion overview from JP Morgan** along with your **internal AWS exploration summary**.

Here’s how we can **enhance your MDSOR executive pitch** by weaving in the **core Fusion principles** from JPMorgan — without directly naming AWS or Fusion — but using **the same compelling language, positioning, and logic** to frame your ask.

**🔧 FUSION-INSPIRED BUSINESS LANGUAGE FOR YOUR PITCH**

These additions will elevate your ask by showing strategic thinking, agility, and customer centricity — and align with how industry leaders like JPMorgan are solving the same problems.

**🔹1. Data is the Bottleneck, Not the Models**

“The biggest drag on speed, analytics, and AI isn’t modeling or infrastructure — it’s the **effort to prepare, normalize, and trust the data**. Everyone wants better forecasting and PnL explain, but the foundation — clean, ready-to-use data — is what’s missing.”

🧠 *Inspired by Fusion: “Analytics and AI are powerful — but what slows everybody down is the data.”*

**🔹2. Normalize Early, So Consumers Don’t Have To**

“Instead of every team rebuilding the same joins, mappings, and identifiers, we’re shifting normalization upstream.  
**MDSOR does the hard work early — applying structure, common identifiers, and conventions — so consumers can focus on insight, not data wrangling.**”

🧠 *Inspired by Fusion: “Rather than every consumer dealing with normalization, shift left — normalize on the way in.”*

**🔹3. From Tables to Datasets to Insights**

“Today, our market data lives in tables and logs scattered across systems.  
**MDSOR turns raw feeds into structured, discoverable, query-ready datasets — with lineage, context, and APIs.**  
That’s how we enable faster onboarding, easier scenario runs, and clean audit responses.”

🧠 *Inspired by Fusion: “Data is buried in thousands of tables. Fusion turns it into usable datasets, easy to query and consume.”*

**🔹4. Shift From Manual Review to Embedded Controls**

“We’re embedding validation rules, override workflows, and lineage directly into MDSOR.  
**Users can flag issues, trace changes, and express overrides in a governed flow** — enabling transparency, accountability, and custom business views.”

🧠 *Inspired by Fusion: “Users can opine on data points, propose overrides, and once approved, those become official.”*

**🔹5. No More ‘Where is the Data?’**

“Today, analysts and quants spend hours just finding data.  
MDSOR aims to fix that by becoming **the trusted place to find and access all market data — with versioning, lineage, and semantic search**. Not email. Not CSVs. Not tribal knowledge.”

🧠 *Inspired by Fusion: “Data must be easy to discover, easy to access, and ready to use via API.”*

**🔹6. Deliver Data to the Destination of Choice**

“With MDSOR, users won’t need to learn the system.  
They’ll **describe what they want — credit curves, VIX series, audit snapshots — and MDSOR will deliver it to their preferred environment**, from notebooks to risk engines.”

🧠 *Inspired by Fusion: “Users express their intent and we deliver the data to their technology of choice.”*

**🔹7. Save Months of Manual Work**

“By integrating ingestion, validation, and curation, MDSOR can **reduce data onboarding and remediation from months to weeks** — enabling faster product rollout and faster regulatory responses.”

🧠 *Inspired by Fusion: “Go from months and years to days and weeks.”*

**🔹8. Managed Data, Not Just Managed Infrastructure**

“This is not just about storage. It’s about providing **a fully managed market data experience** — from ingestion to quality to delivery — so teams can move fast with confidence.”

🧠 *Inspired by Fusion: “Fusion is a managed data service — combining onboarding, modeling, and delivery across all vendor and internal data.”*

**✳️ Optional Final Slide Title**

**MDSOR = Clean Data, Trusted Delivery, Business Acceleration**  
Like Fusion, MDSOR offers:

* Clean, ready-to-use data
* Embedded controls and governance
* Fast onboarding of vendors and internal risk factors
* Data that’s discoverable and delivered wherever it’s needed

Would you like me to:

* 🎨 Update your actual **PowerPoint slides** with this enriched pitch?
* 🎤 Write the **2-minute spoken opener** for your meeting?
* 🧾 Draft a crisp **one-pager for the C-Suite** to summarize the ask?

Just say the word.

Here’s a structured **hiring plan for MDSOR** to onboard market data across all asset classes and elevate the platform to Bloomberg (BBG)-like standards. This includes ETL, platform modernization, UI/UX, governance, and scalability.

**🧩 1. Executive Summary**

To successfully onboard and manage market data across all asset classes while modernizing MDSOR into a Bloomberg-like platform, we recommend a phased hiring plan spanning 6 key pillars:

* Data Ingestion & ETL
* Data Modeling & Engineering
* Platform Modernization (Infra & Storage)
* Analytics & Derivations
* UI/UX & APIs
* Governance & Program Oversight

**👥 2. Full Hiring Plan by Function**

| **Role** | **Headcount** | **Location** | **Core Responsibilities** |
| --- | --- | --- | --- |
| **Lead ETL Engineer** | 1 | Onshore | Architect and oversee batch + real-time data pipelines |
| ETL Developers | 3 | Nearshore/Offshore | Build ingestion pipelines for vendors, trades, referential data |
| **Data Modeler** | 1 | Onshore | Define standard taxonomy, schema, normalization layers |
| Data Engineers | 2–3 | Onshore/Offshore | Time series processing, large-scale data wrangling, HBase/TSDB |
| **Cloud Infrastructure Engineer** | 1 | Onshore | Modernize storage (e.g. S3, parquet, Delta Lake), caching, compute |
| DevOps & Platform Engineers | 2 | Nearshore | CI/CD, environment mgmt, observability, failover readiness |
| **Quant Data Analyst** | 1 | Onshore | Help design derived risk factors, transformations (e.g. curves) |
| Python Quant Developer | 1 | Onshore | Implement quant analytics and pricing tool integrations |
| **UX Designer / Product Designer** | 1 | Onshore | Build intuitive market data discovery tools and dashboards |
| React/Angular Frontend Devs | 2 | Nearshore | Build GUIs for search, trace, preview, validation, auditing |
| API Developers | 1 | Nearshore | Build REST/gRPC APIs for Python, Excel, web apps |
| **Data Governance Lead** | 1 | Onshore | Define lineage, controls, entitlements, audit |
| Governance Analyst | 1 | Offshore | Monitor metadata catalog, roles, access requests |
| **Program Manager** | 1 | Onshore | Cross-stream coordination, planning, stakeholder mgmt |
| Scrum Masters | 1 | Nearshore | Sprint orchestration, backlog grooming, reporting |

📌 **Total Headcount**: ~22 FTEs  
📅 **Duration**: 6–12 months initial rollout, with sustained team for BAU and future enhancements

**🗂 3. Key Responsibilities by Stream**

**A. Data Ingestion & ETL**

* Vendor feeds: BBG, Refinitiv, ICE, Markit, internal trades
* Clean pipeline: raw → validated → normalized
* Add bitemporal + versioning

**B. Storage & Infrastructure**

* Move toward scalable object store (S3, GCS) + Parquet or Delta
* Implement HBase/TSDB enhancements for time series indexing
* Enable caching (Redis) and DR

**C. Data Modeling & Derivation**

* Standard risk factor taxonomy
* Cross-asset schema support (IR, FX, EQ, Credit, Commodities, Structured)
* Transformations: derived curves, forward rates, spreads

**D. Platform Modernization**

* Replace rigid flows with modular, metadata-driven design
* Expose Python/REST APIs for quant use
* Elastic compute for curve construction, backfill

**E. UI/UX Modernization**

* Data discovery portal (like BBG’s FLDS)
* GUI for trace, preview, override, validation history
* Interactive dashboards for QA and remediation

**F. Governance, Controls & Lineage**

* Metadata catalog (Snowflake/Athena-like)
* Data quality dashboards
* Access control + audit logs

**🚀 4. Hiring Prioritization (Phased Rollout)**

**Phase 1: Platform & Core ETL**

* Lead ETL Engineer
* 2 ETL Devs
* Infra Engineer
* Program Manager
* Data Modeler
* Governance Lead

**Phase 2: UI/UX + Analytics**

* Frontend Devs
* UX Designer
* Quant Data Analyst
* API Dev
* Python Quant Dev

**Phase 3: Controls & Scale**

* Remaining ETL / DevOps
* Governance Analyst
* Scrum Master
* Additional data engineers

Would you like this packaged as a PowerPoint slide, Excel budget table, or Word proposal?

1. **This is list of work**

* **Decoupled from Vasar**
* **Data Integration and Ingestion**
  + Build Modular ingestion pipleles ( Kakfa,)
  + Enforce Data contract, versionning, SLA,
* **Data Modeling and Transformation**
  + Implement centralized taxnomoy layer
  + Normaliz format, conventions
  + Generated derived curves
  + Enable Bitemporarl modeling
* **Build Advanced Analytics and Validation**
  + Detect anamoloes using advanced ML
  + Build remediation workflows
  + Maintain immutable audit trail
* **Onboard all risk factors**

|  |  |  |
| --- | --- | --- |
| Capability area |  |  |
| Decoupled from A | Refactor B as stanadlone Remove dependecy on A Elimiate Reliance on Team A | Enable indepenedency scaling and  onboarding Reduce Bottle necks and accelerate  delivery Support Braoder use cases Beyong  use case A |
| Data Integration and Ingestion | Build Modular ingestion pipleles ( Kakfa,) Enforce Data contract, versionning, SLA, | Support ream time batch  Support Consistency and relaibity |
| Data Modeling and Transformation | Implement centralized taxnomoy layer Normaliz format, conventions Generated derived curves Enable Bitemporarl modeling | Improve interoperability and tracebility Support factor |
| Validation and Q control | Detect anamoloes using advanced ML  Build remediation workflows  Maintain immutable audit traisl |  |
| Discoverability and User Experience | enhance searching mechanism,  API m SDKs, Plugins Build GUIs | democratize access |
| analtics ad compute | Curve Fitting , boostrapping tools, scenario engies |  |