agilelab

How to identify Data Products? Welcome “Data Product Flow”

Nov 22, 2021 5:56:54 AM

TECH ARTICLE KNOWLEDGE BASE

We all know that Data Mesh is inspired by DDD principles (by Eric Evans), transposing them from the operational plane (microservice architecture) to the analytical one (data mesh).

Since one year and a half, we have been helping many big enterprises to adopt the data mesh paradigm, and it’s always the same old story. The concept itself is really powerful and everybody can catch the potential from the early days because it addresses real problems, so it is not possible to ignore it.

Once you have the buy-in of high-level concepts and principles, it is time to draft the platform capabilities. This step is game-changing for many companies and is often becoming challenging because it revolutionizes processes and tech stacks.

But the most complex challenge is another one; it is something that blows people’s minds away.

What is a Data Product? I mean, everyone understands the principles behind that, but when it comes to defining it physically… is it a table? Is it a namespace? How do I map it with my current DWH? Can I convert my Data Lake to Data Products? These are some of the recurring questions… and the answer is always “no” or “it depends”.

When we start to introduce concepts like bounded context and other DDD elements, most of the time is getting even harder because they are abstract concepts and people involved in Data Management are not familiar with them. We are not talking with software experts; DDD until now has been used to model software, online applications that need to replicate and digitalize business processes. Data Management people were detached from this cultural shift; they typically reason around tables, entities, and modeling techniques that are not business oriented: 3NF, Dimensional modeling, Data Vault, Snowflake model… all of them are trying to rationalize the problem from a technical standpoint.

So after a while, we arrive at the final question: **How do we identify Data Products?**

For DDD experts, the answer could seem relatively easy…but it is not !!!

Before to deep dive into our method to do that, let’s define an essential glossary about DDD and Data Mesh (coming from various authors):

Domain and Bounded Context (DDD): Domains are the areas where knowledge, behaviour, laws and activities come together. They are the areas where we see semantic coupling and behavioural dependencies. It existed before us and will exist after us; it is independent by our awareness.

Each domain has a bounded context that defines the logical boundaries of a domain’s solution, so bounded contexts are technical by nature and tangible. Such boundaries must be clear to all people. Each bounded context has its ubiquitous language (definitions, vocabulary, terminology people in that area currently use). The assumption is that the same information can have different semantics, meanings and attributes based on the evaluation context.

Entity (DDD): Objects that have a distinct identity running through time and different representations. You also hear these called «reference objects».

Aggregate (DDD): It is a cluster of domain objects or entities related to each other through an aggregate root and can be treated as a single unit. An example can be an order and its line items or a customer and its addresses. These will be separate objects, but it’s useful to treat the order (together with its line items) as a single aggregate. Aggregates typically have a root object that provides unique references for the external world, guaranteeing the integrity of the Aggregate as a whole.

Transactions should not cross aggregate boundaries. In DDD, you have a data repository for each Aggregate.

Data Product (Data Mesh): It is an independently provisionable and deployable component focused on storing, processing and serving its data. It is a mixture of code, data and infrastructure with high functional cohesion. From a DDD standpoint, it is pretty similar to an Aggregate.

Output Port (Data Mesh): It is a highly standardized interface, providing read-only and read-optimized access to Data Product’s data.

Source-aligned Data Product (Data Mesh): A Data Product that is ingesting data from an operational system (Golden Source)

Consumer-aligned Data Product (Data Mesh): A Data Product that is consuming other data products to create brand new data, typically targeting more business-oriented needs

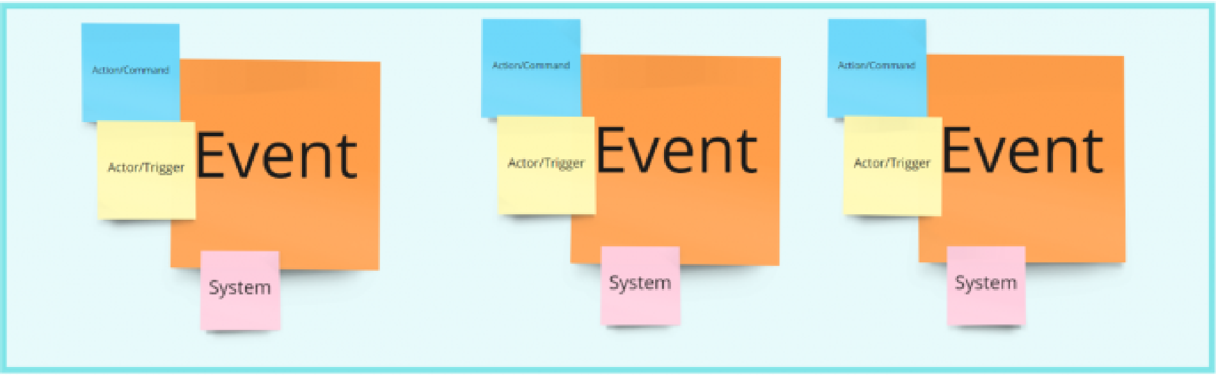
So, how do we identify Data Products?

Because a Data Product, from a functional standpoint, is pretty similar to an Aggregate, we can say that the Data Product’s perimeter definition procedure is pretty similar. Keep in mind that this is valid only for Source Aligned Data Products because drivers are changing when we enter into the data value chain.

In Operational Systems, where the main goal is to “run the business”, we define Aggregates by analyzing the business process. There are several techniques to do that. The one I like most is **Event Storming** (by Alberto Brandolini). It is a kind of workshop that helps you decompose a business process into events (facts) and analyze them to find the best aggregate structure. The steps to follow are straightforward:

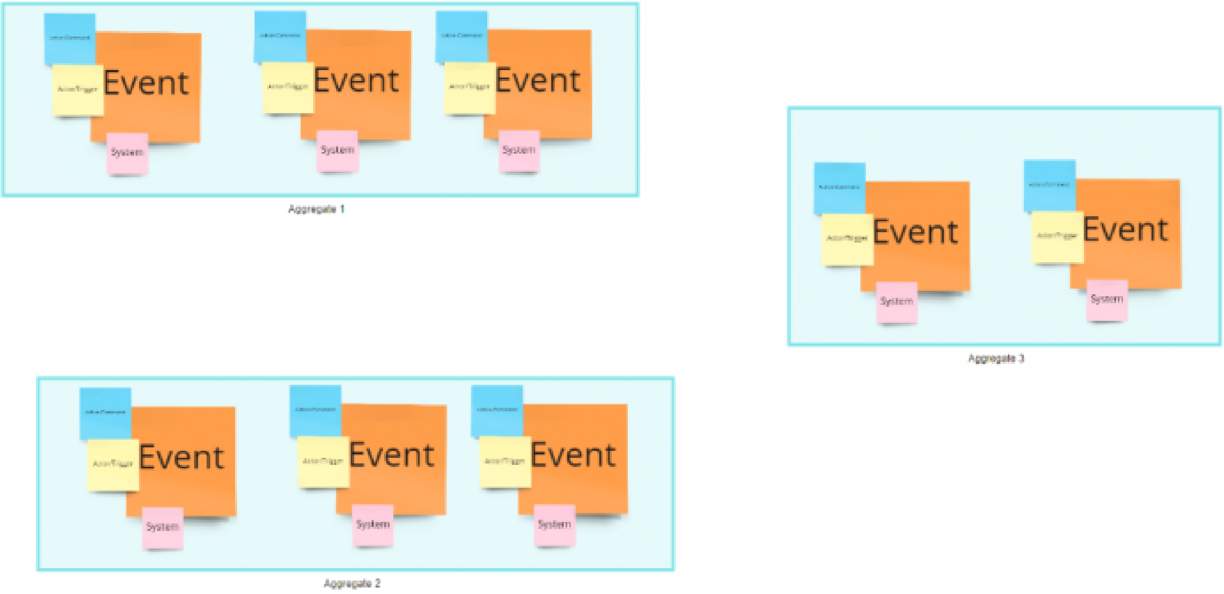
1. Put all the relevant business events in brainstorming mode
2. Try to organize them with a timeline
3. Pair with commands and actions
4. Add actors, triggers, and systems

5. Try to look for emerging structures
6. Clean the map and define aggregates



Aggregate

The process itself is simple (I don’t want to cover it because there is a lot of good material out there), but what makes the difference is who and how is driving it. The facilitator must ask the domain experts the right questions and extract the right nuances to crunch the maximum amount of domain information.



There are other techniques like bounded context canvas or domain storytelling, but, if your goal is to identify Data Products, I would suggest using Event Storming because, in Data Mesh, one of the properties that we want to enforce is the immutability and bi-temporality of data, something that you mainly achieve with a log of events.

So, Event Storming is the preliminary step of our process to discover aggregates.

If your operational plane is already implemented with a microservice architecture with DDD, probably you already have a good definition of aggregates but, if you have legacy operational systems as golden sources, you need to start defining how to map and transform centralized and rationalized data management practice into a distributed one.

Once you have your business processes mapped, it's time to think about data and how they can impact your business. The analytical plane is optimizing your business, and this comes with better and informed decisions.

Now we enter into our methodology: **Data Product Flow.**

Data Product Flow

Each event is generated by a command/action and a related actor/trigger. To make an action, people in charge of that step need to make decisions. The first step of our process is to let all the participants put a violet card describing the decision driving the creation of that event and related actions. This decision could be the actual one or how it could be in a perfect world. People participating should also think about the global strategy of their domain and how those decisions will help to move a step forward in that direction. We can now introduce the concept of Data Opportunities:

A **Data Opportunity** is:

- A more intelligent and more effective way to get a decision supported by data. You could be able to automate a decision fully or create some suggestions for it.
- A brand new business idea that is going to be supported by Data. In that case, it will not be part of the identified aggregates.

At this stage, let's start to talk about Data Products to emphasize that, from now on, the focus will entirely be on the data journey and how to be data-driven.

In this phase, the facilitator needs to trigger some ideas and do some examples to let the participants realize that now we are changing direction and way of thinking (compared with the previous part of the workshop). We are not focusing anymore on the actual business process, but we are projecting into the future. We are imagining how to optimize and innovate it, making it data-driven.

When people realize that they are allowed to play, they will have fun. It is crucial to let people’s creativity flow.

Once all the business decisions are identified, we start to think about what data could be helpful to automate (fully or partially) them. Wait, but until now, we didn’t talk about data so much. Where the hell are they?

Ok, it’s time to introduce **the read model** concept. In DDD, a read model is a model specialized for reads/queries and it takes all the produced events and uses them to create a model that is suitable to answer clients’ queries. Typically we observe this pattern with CQRS, where starting from commands, it is possible to create multiple independent read-only views.

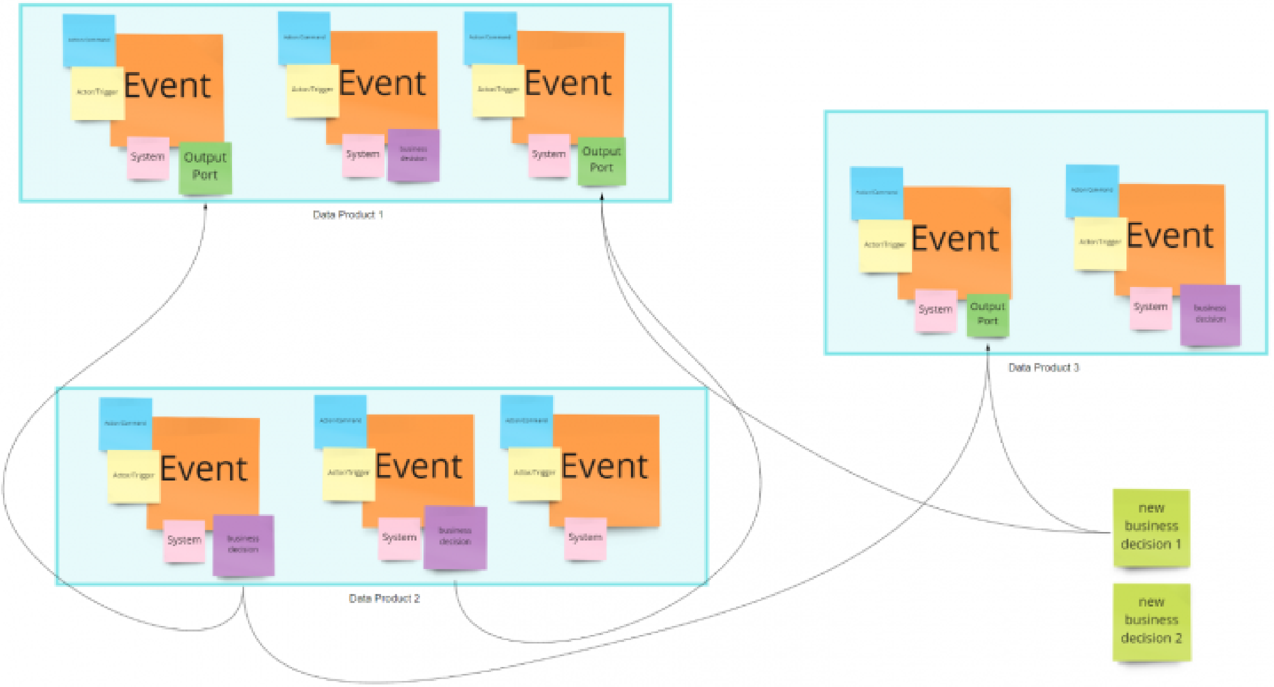
When we transpose this concept in the Data Mesh world, we have Output Ports.

For each Business Decision, let’s identify datasets that can provide help in automating them. Create a green post-it in each data product where we believe it is possible to find the data we need and position them close to the events that probably generate the data underlying such output port.

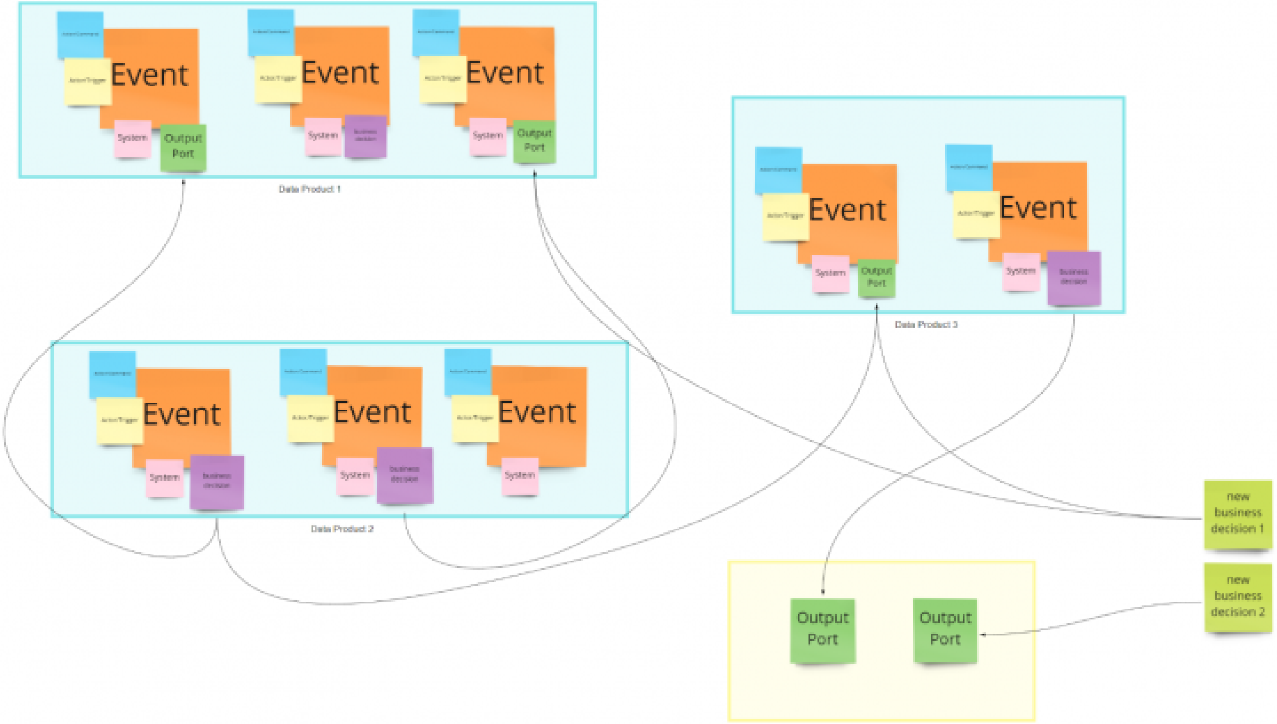
Please, pay attention to giving a meaningful name to the output port, respecting ubiquitous language and not trying to rationalize it (there is always time to clean the picture, early optimization is also not good in this field). I repeat, because it is vital, please provide domain-oriented and extensive output port names, don’t try to define a “table name”.

Do the same for “new business decisions”, those that are not part of the actual business process.

At this stage, your picture should be more or less like the following one.



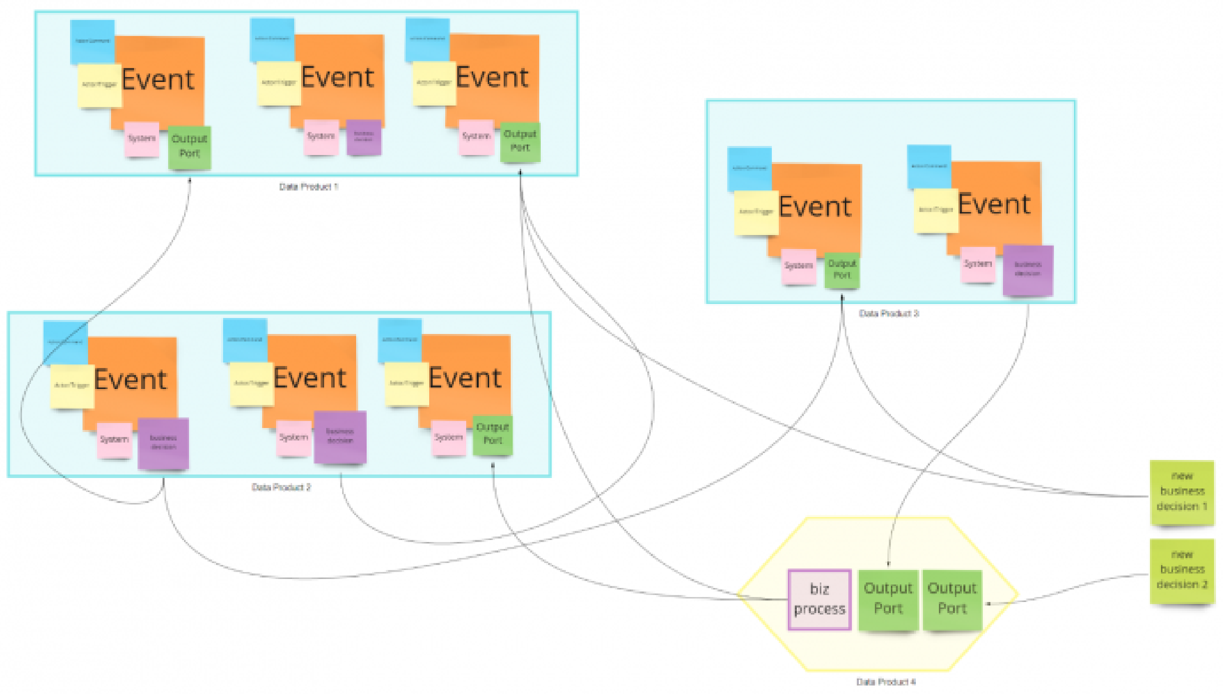
If you are working in a real environment, you will soon realize something scary: while you are searching for datasets that can help with your business decisions, you would be able to imagine what information could be helpful, but you would not be able to find the right place for it on the board, because simply that kind information does not exist and you need to create it.
Probably this is your first consumer-aligned Data Product !!
Start creating a placeholder (in yellow) and create the output ports needed to support your business decisions.



These new output ports can be part of the same data product, or you may need to split into different data products. To define it, we need to apply some DDD principles again. This time we need to think about the business processes that will generate such data:

- Is the same business process generating both datasets at the same time?
- Do we need consistency between them?
- Can we potentially apply for different long-term ownership on them or not?

There are many other rules and ways to validate data product boundaries, but it is not the focus of this article. Fortunately, in this case, we can unify the process under unique ownership and have two different output ports for the same data product because they have high functional cohesion. Still, we want to provide optimized read models to our customers.



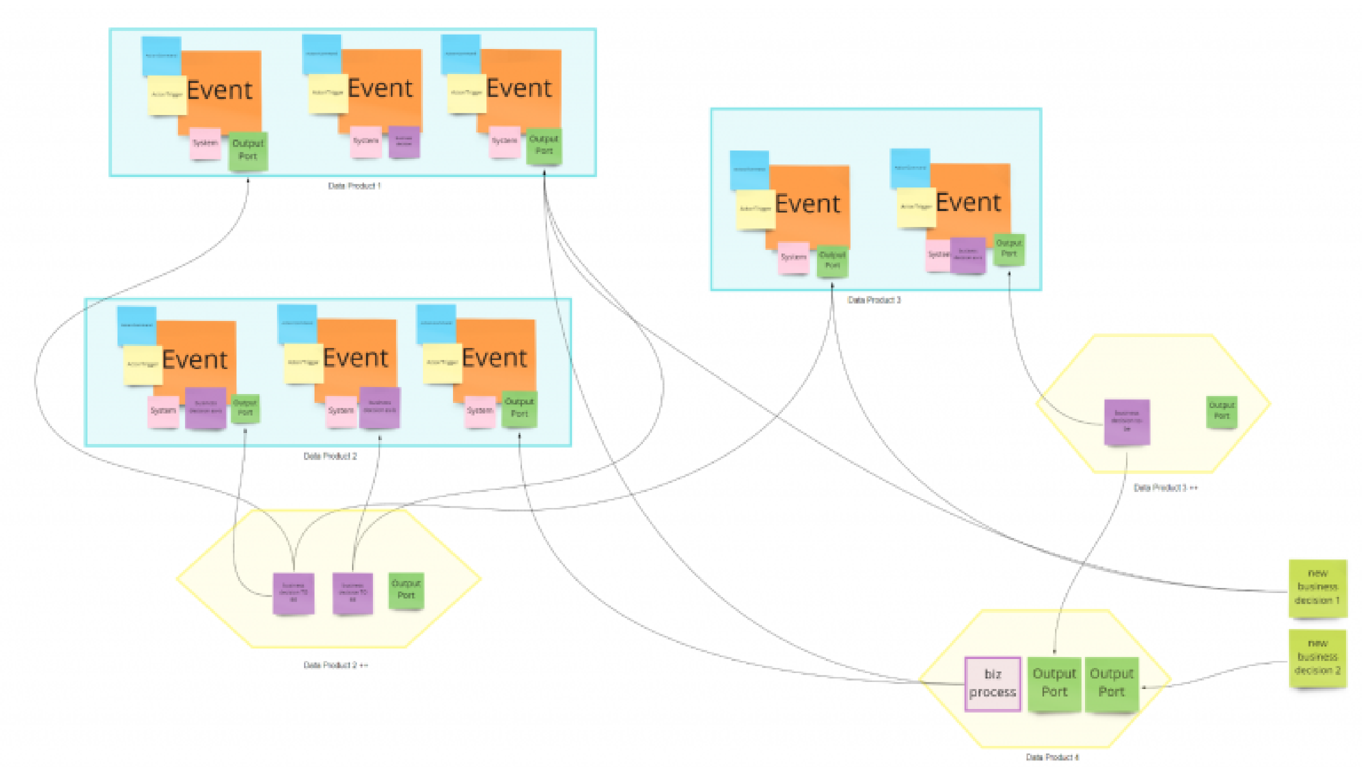
This process can also have several steps, and you need to proceed backwards, but always starting from business decisions.

In this phase, if it is becoming too abstract and technical for domain experts, simplify and skip it; you can work on it offline later. It is essential to don’t lose domain expert attention.

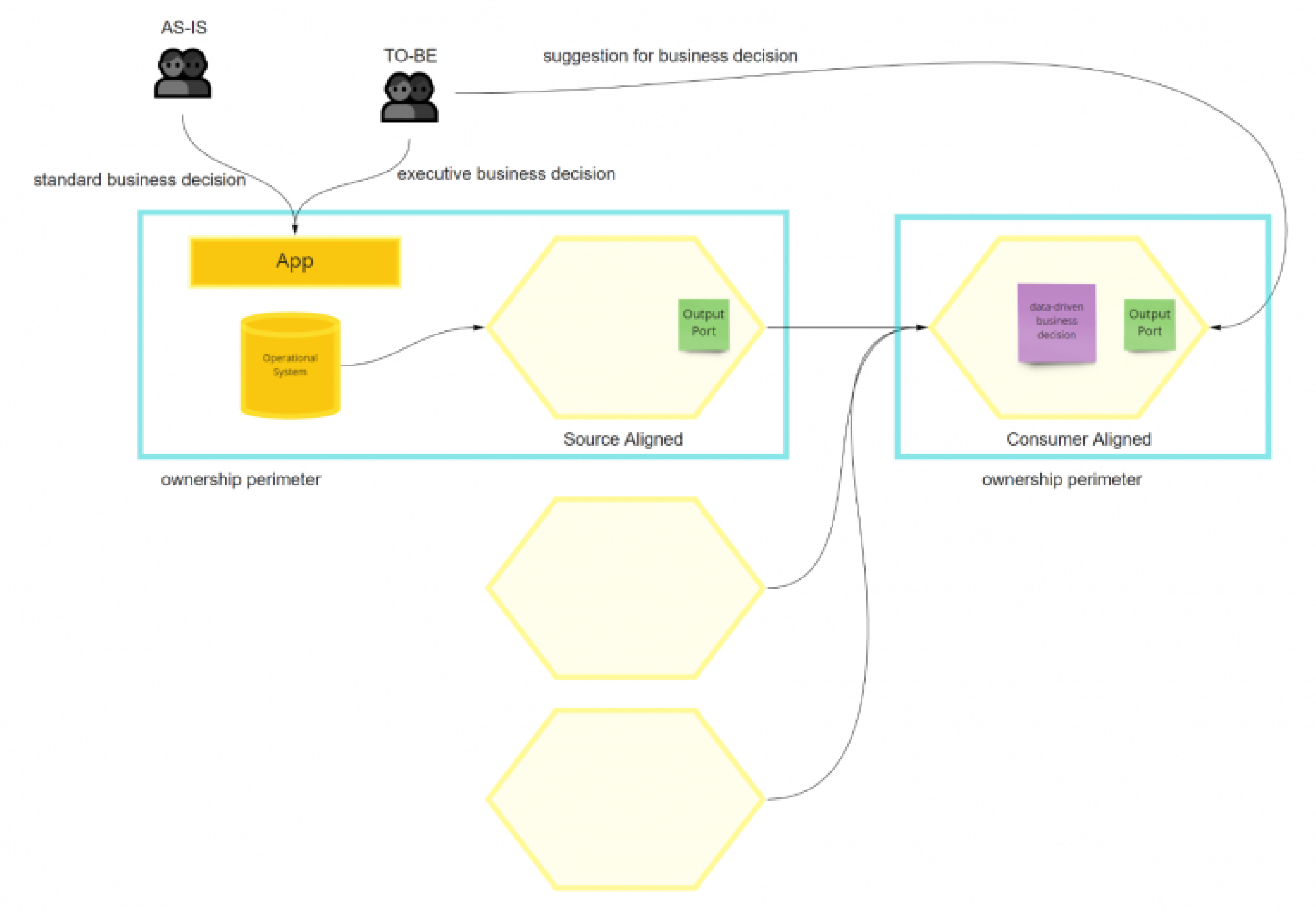
...

Because Data Mesh is not embracing the operational plane of data, we now need to do something fundamental to stay consistent and model authentic data products.

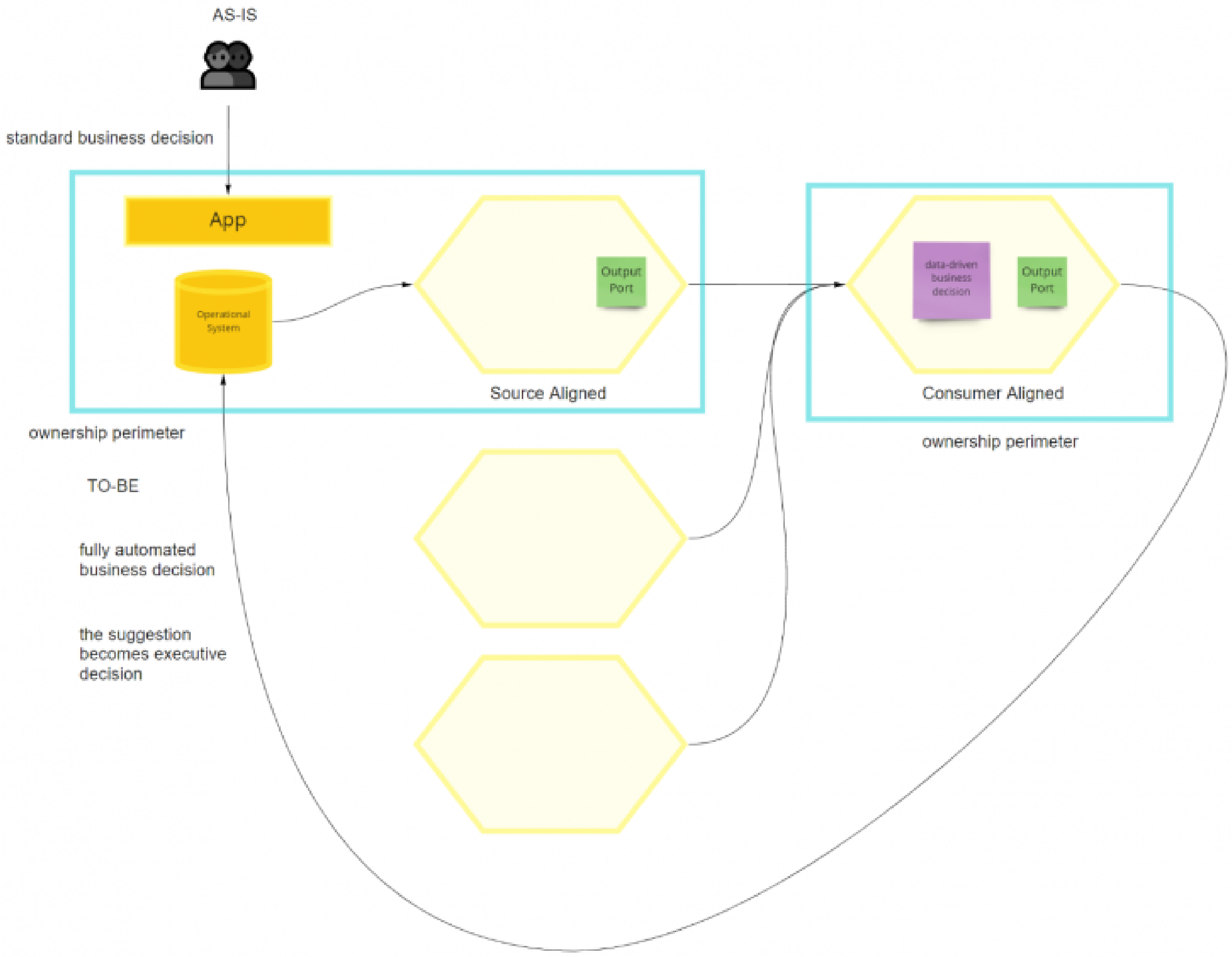
At this stage, if on your board you have a data product that is taking business decisions leveraging output ports of other data products and also has the “system” post-it, it means you are mixing operational and analytical planes. It cannot be accurate because the business process is happening only in one of the two. Data Product 1 is purely operational because it is not going to make data-driven decisions. Data Product 2/3 processes are happening on operational systems for sure because we started from there. Still, we would like to automate or support some business decisions by reading data from the analytical plan. When you detect this situation is better to split the data product by keeping a source-aligned data product that maps on the operational system’s data and then create an additional DP to include the newly added value business logic. These are more consumer-aligned DPs because they are reading from multiple data products to create something entirely new and new ownership.



To understand better, this last step could be helpful to visualize it differently. With the standard business process (AS-IS), users make decisions, propagating them in the source-aligned DP. Suppose we want to support the user to make smarter or faster decisions. In that case, we can provide some data-driven suggestions through a consumer-aligned data product that is also mixing-in data from other DPs. In the end, it is also the same business decision. Still, it is better to split it into executive decisions and suggestions to separate the operational plane and the analytical one. If you are thinking of shrinking everything in the source-aligned DP, keep in mind that it includes the executive business decision (facts). You will end up mixing operational and analytical planes without any decoupling. It is also dangerous from an ownership standpoint because once you consume data from other domains, the DP will not be “source-aligned” anymore.



Also, if you can automate such business decisions fully, I strongly recommend keeping them as separate Data Products.


AS-IS
standard business decision
App
Operational System
Output Port
Source Aligned
ownership perimeter
data-driven business decision
Output Port
Consumer Aligned
ownership perimeter
TO-BE
fully automated business decision
the suggestion becomes executive decision

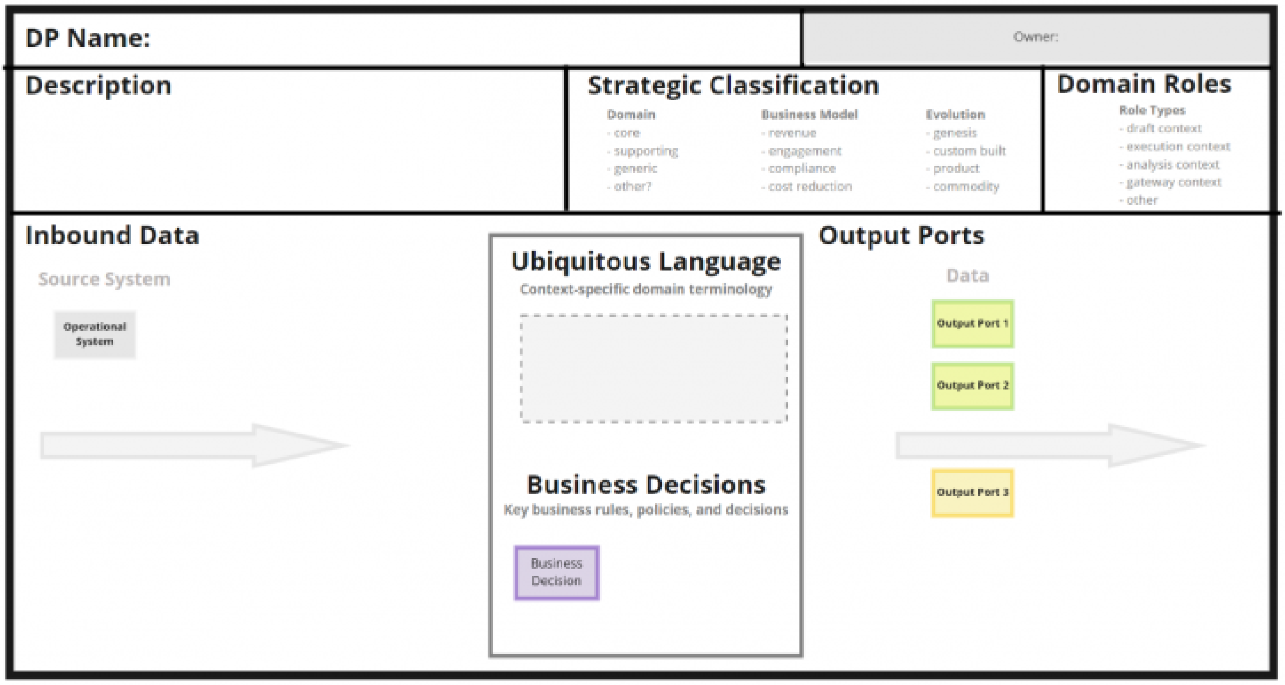
...

How to formalize all this stuff? Can we create a deliverable that is human readable and embeddable in documentation?

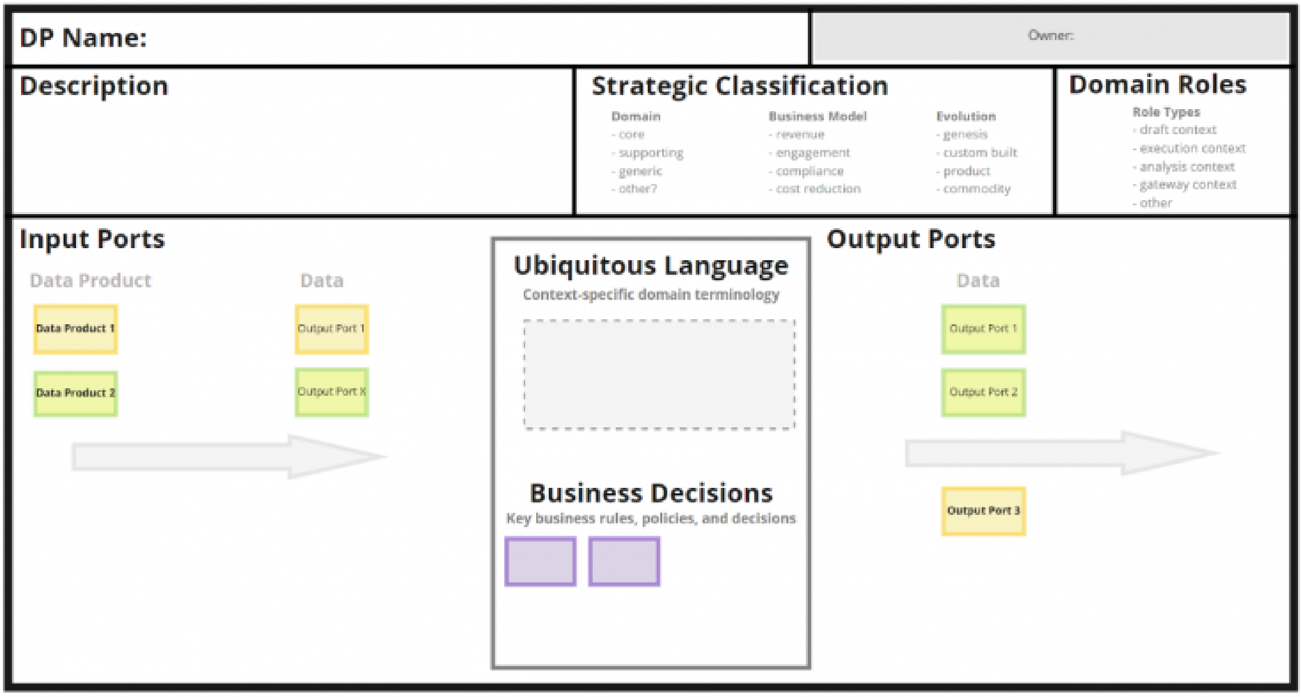
DDD is helping us again. We can use Bounded Context Canvas to represent all the Data Products.

I propose two different templates:

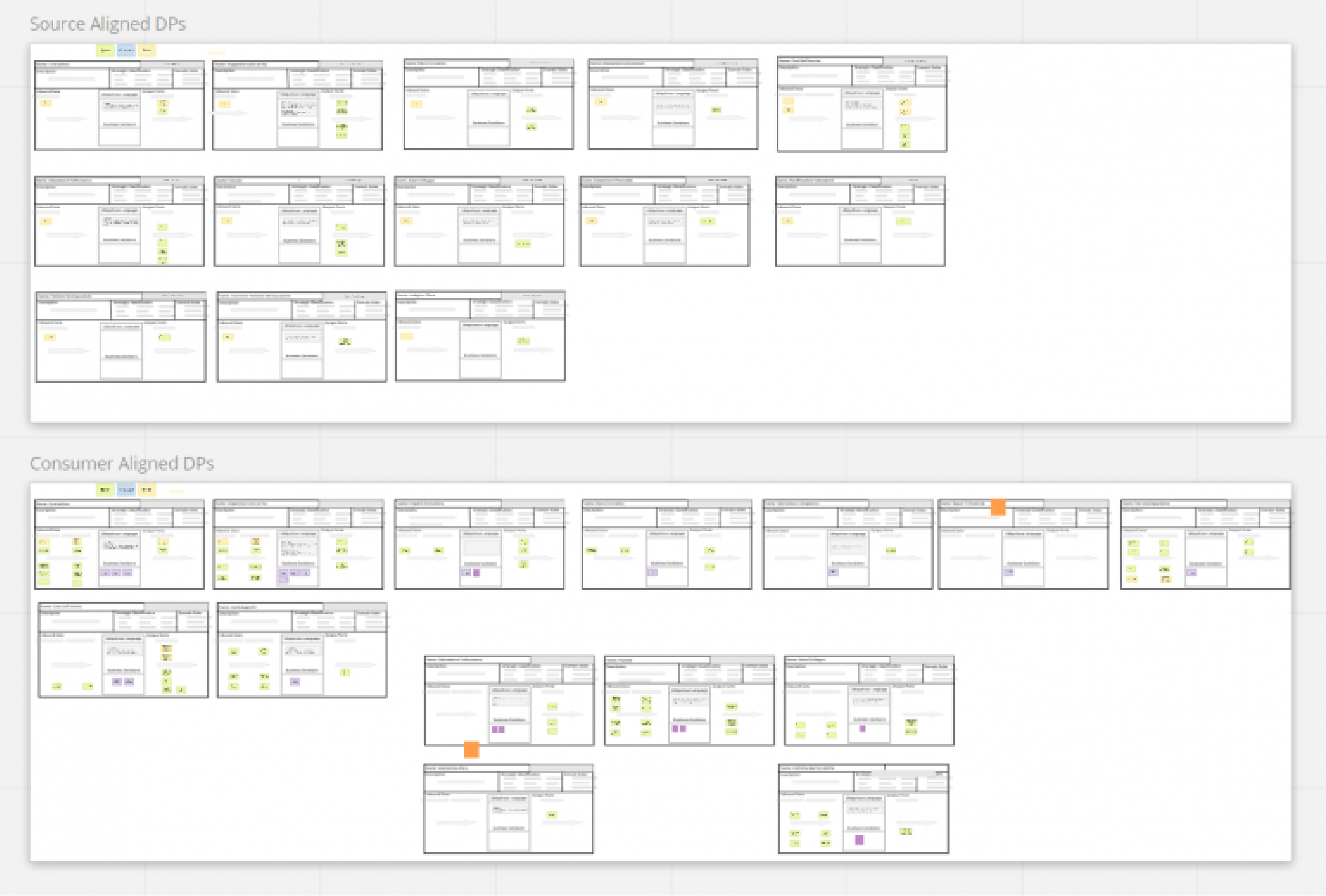
- Source-aligned DP



- Consumer-aligned DP



The final result is quite neat and clean.
Welcome, Data Product Flow as practice and workshop format.



LEARN MORE ABOUT DATA MESH BOOST

References:

- Data Mesh Principles and Logical Architecture (martinfowler.com)
- Remote EventStorming (avanscoperta.it)
- domain driven design (martinfowler.com)

Thanks to Roberto Coluccio. For a deep focus on Data Mesh, you might also be interested in reading more about Customer 360 and Data Mesh: friends or enemies?

STAY TUNED!

If you made it this far and you’re interested in more info about the Data Mesh topics, or you’d like to know more about Data Mesh Boost, get in touch below!

Posted by Paolo Platter

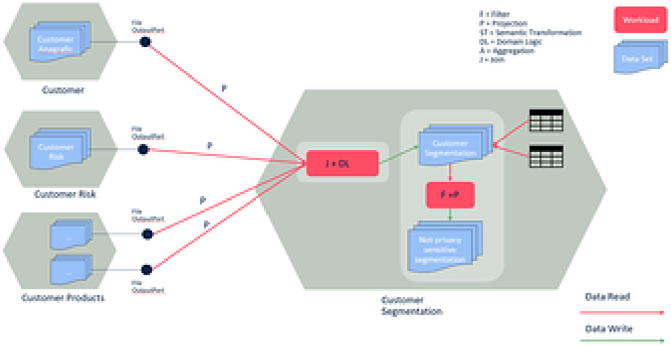
CTO & Co-Founder. Paolo explores emerging technologies, evaluates new concepts, and technological solutions, leading Operations and Architectures. He has been involved in very challenging Big Data projects with top enterprise companies. He's also a software mentor at the European Innovation Academy.
LinkedIn

Talk to our Experts

Related articles

Cos’è il Data Mesh e perché risolve problemi - 01 Net

Data Product Specification: power up your metadata layer and automate your Data Mesh with this practical reference



How to model Data Products

Founded in 2014, Agile Lab specialises in bespoke, cutting-edge technology solutions to provide its Clients with a concrete competitive advantage by creating efficiency and optimising all data-intensive business processes.

Latest news

- 01net - Data, cloud, skills: what awaits us in 2023
- Agile Lab acquired by Poste Italiane
- Technology Abstraction in the Data Mesh