**Data engineering failure — Why is it almost impossible to meet deadlines?**

**“I love deadlines. I love the whooshing noise they make as they go by.”**



Engineering is an art (credits: me).

Failure. I would like to write about failure. Why is it so hard for data engineers to evaluate projects duration but also to meet deadlines?

Over my last 7 years of experience I had the opportunity to work with multiple data engineering teams. A common pattern I noticed projects after projects was the difficulty to define and meet milestones for engineers.

Teams after teams we tried all the trendy methodologies in order to create roadmaps and milestones. Jumping from Scrum to Lean, trying to size tasks using poker planning or with peer evaluation. That didn’t stop us from failing. But, this is normal.

In this post I’ll share the 3 main reasons of our failure as data engineers with solution proposals to open the discussion with the community.

**Data Engineers are bad at prioritization**

We are bad at prioritization, which means lots of interruptions and unrealistic timelines. You said interruptions? Yes, data engineering deeply suffer of context switching. We often jungle between tasks on a daily basis.

Here is the exaggerated (is it really?) daily planning of a data engineer:

* morning — fix pipelines which broke during the night
* noon — answer a public Slack question on #data-public regarding a warehouse column
* afternoon — help one analyst with a tricky issue you chatted about at the coffee break in the morning
* afterwork — you start working on your daily task

Because data engineers are historically seen as a support team, they need (or they want) to answer as fast as they can which obviously conflict with project deadlines. It’s hard to say no. I know. But to preserve your mental health you’ll need to learn.

On the other hand we also have too much recurrent pipeline issues. Writing error-less pipelines is a Herculean task. But why? Because we often are between data providers and consumers (cf. data mesh), facing too many moving parts. We move data from a database to storage or to third party platforms. Database schema, storage, third party API or business requirements can change. And then generate new issues.

But let’s not blame only the other teams, we also sometime build crappy pipelines because we do not have enough time or because we just botch. It is hard to prioritize between planned projects, unplanned projects and daily bugs & questions.

**Solutions**

In order to solve this prioritization issue I offer you 4 paths to explore. You’ll see that the three first ones are organizational, the last one is more about removing tech debt to avoid paying it daily.

1 — Start your journey to become a product team. Make people understand you build tools with them rather than only provide a service. Then, open a Product Management position in order to start considering data like a product. This bullet is a big trend in the industry right now with data mesh and everything around it. *Warning: do not hire a PM to manage a tasks backlog.*

2 — If you manage a data team, create a trust climate that will allow engineers to say no if necessary. **Saying no is not an easy thing to do when a lot of pressure comes from the top.**

3 — Create a Totem holder role. Within the team someone should be responsible on a weekly basis to answer all possible external interruptions. Creating the weekly rotations and the role is easy but don’t forget to check if the role really works. Does this generate less interruptions? If not, find why and improve.

3 — Apply [software engineering best practices to your pipeline](https://maximebeauchemin.medium.com/functional-data-engineering-a-modern-paradigm-for-batch-data-processing-2327ec32c42a) or app development. Concepts like idempotency, reproducibility, read/write schema, etc. are a must-do. **When you develop a pipeline, it will most likely fail one day.** So do yourself a favor for the future, make it easy to debug and relaunch. And remember: *it is simple to build complex, but complex to build simple.*



Every day a new crossroad ([credits](https://unsplash.com/photos/BkkVcWUgwEk))

**Data Engineering projects are on another schedule**

Data Engineering tasks are rarely small ones compared to other data tasks. You probably already faced this situation where an analyst asks for a table to answer a business question and by the time you add the table the analyst has already found a solution to get the data.

The previous situation illustrates the different schedules data people are in and it also depicts the complexity of syncing them up.

* Data Analysts are on daily-based schedules, as they mainly answer questions (business or instincts).
* Data Scientists are on weekly-based schedules, they transform questions into models thanks to feature engineering.
* Data Engineers are on monthly-based schedules, they build platforms in order to support past, current and future use-cases and growth.

The main reason we work on a monthly-based schedule is that, as of today, tools are still hard to operate (do you remember Hadoop?), we have too many tools and a lot of viable solutions for the same task, without adding the fact that each time we work on something we discover unplanned stuff.

If we add to the tools complexity the low quality inputs data platforms gets. It brings a lot of edge cases: *the normal cases are becoming the edge cases, and the bad input the norm.* Let’s be honest the data quality path has not yet be covered in many data platforms and data engineers are still facing on a daily basis edge cases.

This mix between tools complexity and input quality is something that brings delay on every task we work on. We finally understand why it takes us so long to move forward.



A data engineer late on his project because of a flood ([credits](https://unsplash.com/photos/4dSMCFIcE1A))

**Solutions**

1 — Do baby steps. First time sizing a project? Do not try to size a 1 year project. Slice it up to 3 weeks or 3 months smaller projects and create small understandable tasks. You will be wrong the first times, but after some tries you will be better for sure. *Without practice, you cannot be a master*.

2 — We are all in the same boat. In data teams our North Star should be to empower colleagues with data. Let’s be teammates and understand each others constraints. Engineers are here to create the best platform for the company, Analysts and Scientists are here to create the visibility, the usage and the interface on the data.

3 — Watch for technical innovations and embrace them. A lot of new products are coming out of the ground (Airbyte, Meltano, Singer — to name a few) added to cloud data storage (BigQuery, Snowflake, Databricks). They aim to simplify data engineering complexity. Use them. More importantly: *adapt them to your stack.*

**Data engineering discipline is still young**

The two previous points are accentuated by the fact that the discipline is so young. Every choice or small mistake could lead to technical debt easily.

The industry was wrong for several years and with the data science hype a lot of companies hired data scientist and asked them to do data engineering jobs.**It resulted in two major issues: we had frustrated data scientists and bad engineered platforms.**

Do not get me wrong here, I think that data scientists can build wonderful platforms but when you are doing a job you are not supposed to do the results could be deceptive.

Data engineering teams are often integrated within the “traditionnal” product team but because the discipline is new, other teams are not able to figure out what data engineers are capable of. This is a issue. Sometimes data engineers could bring more adapted solutions to the table to avoid unnessecary work.

We also lack of data product managers or product managers with data platform understanding. It happens often that product managers forget totally the data ecosystem in their scoping adding unplanned work to data teams later.

Also, because of his youth data engineering skillset as of today is quite wide, except for the big corporations that have enough resources to specialize teams, smaller companies need to hire 5-legged sheep data engineers. **Data engineers are asked to jump from DevOps techniques to production-grade code writing then to SQL debugging.**



A data engineer reading “Data Engineering Magazine” ([credits](https://unsplash.com/photos/0qmXPnZKeLU))

There are not that many engineers with multiple experiences building data platforms out there. It is hard for a team on a new project to evaluate how long it will last if no one in the team did it before. If we add to that tools and trends that are changing every 2 years, how can we be great at evaluation?

**Solutions**

1 — Ask for help. You may not have the chance to have a senior in your team, but I think there are people out there that are willing to help or challenge you if you ask.For instance ask other companies with the same size or within the same market, be creative. I did it in the past and I met awesome people.

Challenge your choices and vision by presenting at public conferences and/or meetup. From this you will get questions and feedback that will help you to rate your platform.

2 — Read what others are writing and get inspired. But do not forget that you might not be Netflix, Uber or Airbnb. You probably do not need to create a new open-source database technology to compute your metrics along the way.

3 — Hire teams with diverse skills. If it suits your use-case prefer hire 2 generalists DE with 2 specialized ones (DevOps and streaming for instance) rather than 4 generalists. But be careful, specializing your team too early will also create issues. You don’t want to have only person in charge of streaming. Otherwise holidays will be hard for him.

**Conclusion**

If you are a data engineer, please, be kind to yourself. It takes time to be a great data engineer. If this is the first time you create a platform alone it is normal to struggle.

If you are working with data engineers, please, be kind with them. In their daily professional life engineers love to see people using the tools they build.

But do not forget to put (challenging) deadlines on your projects. It is only like this that you will improve and start building trust with your colleagues. Also have in mind that each solution could have his own article, we have many actionable levers to improve the situation.

This article can also apply more generally to software engineering. Is data engineering really so particular that it has specific needs? **Shouldn’t we consider data engineers like another engineering team?**

*Special thanks to Augustin, Charlotte, Emmanuel and Pierre for the review.*

*Subtitle citation by Douglas Adams.*

*This article has been originally published on* [*blef.fr*](https://www.blef.fr/data-engineering-failure-why-is-it-almost-impossible-to-meet-deadlines/) *where I publish a weekly curated newsletter about data. Engineering, fundraising, analytics, organization everything goes is.*