## Product Development

**Summary:**
Product development can be seen as a funnel that **narrows ideas into a launched** product, moving through **four** main **phases**: **requirements, concept, design, and testing**. Every project falls somewhere between simple support tasks, incremental derivative projects, next-gen platforms, or radical breakthroughs. The **leader's role it to align the right** team type (functional, lightweight, heavyweight, or autonomous) and **process** (Stage-Gate, CPM, or Agile) with the project's **level of complexity** and uncertainty. Metrics like time-to-market, capacity, and productivity help track how well the process is working.

**Key Learning:**
To be strong **engineering leaders**, we must **know** how to **match the project's complexity** with the right **structure** and **process**, giving our team the **best chance to succeed** and produce an impactful product.

## Alignment Matrix

**Summary:**
When big **projects fail**, it's often not just because of the technology but **because** teams don't **talk when they should.** The alignment matrix is a tool that compares where communication *should* happen (based on technical interfaces) with where it *actually* happens (based on team input). This exposes two problems: **unattended interfaces** (where teams need to talk but don't) and **unidentified interfaces** (where unexpected but useful conversations happen). By surfacing these gaps, leaders can **fix communication issues** before they turn into expensive mistakes.

**Key Learning:**
The alignment matrix **shifts focus** from blaming individuals **to fixing the communication system** and it helps leaders make sure that critical conversations happen at the right time, so technical work moves forward without hidden breakdowns.

## I4 Framework Video Summary

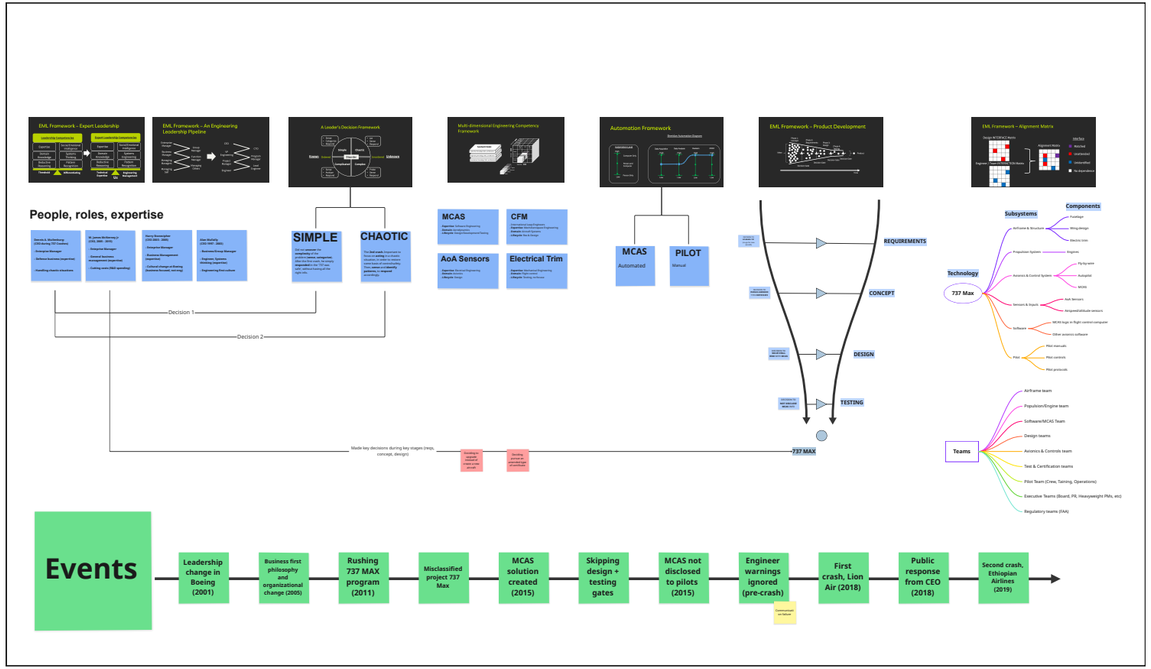
**Summary:**
Prof. Steve showed how the alignment matrix, an important engineering tool, can help (technical) teams understand their shortcomings (in terms of communication) and what they need to do to improve. He mentions the **main difficulty** he **faced** when he inherited the team as a **Program Manager**, with a **heavyweight structure and 30+ engineers**, was the missing **communication** between the two lead engineers. The best part was seeing how the **prof made them understand the issue** by **speaking their language** and using a **technical framework**.

**Favorite moment:**
Minute 4:44, when prof is discussing **how engineers talk a technical language**. It's another important skill to have as an engineering leader, knowing what tools/frameworks to use to effectively communicate to your team.

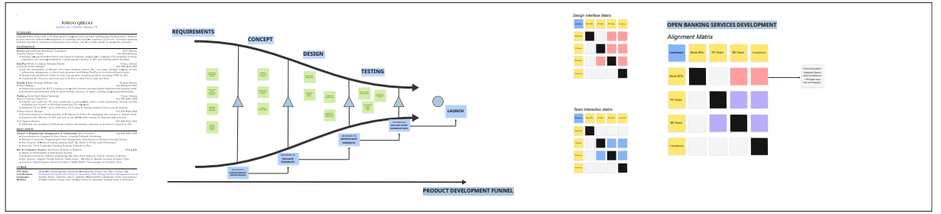
## Boeing Case Study

**Summary:**
Rereading the Boeing 737 Max after reading this week's papers helped me see the importance of **product fundamentals and communication**, and what could happen when they are ignored. Boeing **rushed the Max through the funnel**, relying on an amended certification to avoid costly pilot retraining. Instead of **slowing down at key gates (like design and testing)**, leadership cut corners, treating a **complex project like a simple** derivative **one**. At the same time, the "alignment matrix" problem was clear: the technical system (MCAS depending on AoA sensors) demanded communication between software, mechanical, and pilot training teams but those conversations didn't happen. The result was a hidden system, poor coordination, and ultimately two fatal crashes.

**Key learning:**
Successful product development is **less about speed** and more about **matching the right process and team structure** to the project's **complexity**, and making sure **communication lines match technical dependencies.** When leaders ignore these fundamentals, unattended interfaces (like pilots not being told about MCAS) and misclassified projects (treating a breakthrough-level redesign as a derivative) can lead to catastrophic failure.




**Connection Visual Analysis**




**Boeing Case Study**