EDIE

Navigating the Flight Test Data Skies

AIRBUS

Who we are

Arnaud GLIN
Airbus Flight Physics — Digital A/C Engineer
arnaud.g.glin@airbus.com

Adil SOUBKI
Airbus Flight Tests — Data Scientist
adil.soubki@airbus.com

Vincent MALET
Airbus Flight Tests — AI & Big Data Engineer
vincent.v.malet@airbus.com

DISCLAIMER

The authors acknowledge the substantial contribution of Google's™ Gemini™ Flash 2.0 large language model in the drafting, structuring, and refinement of this manuscript. While the core ideas, research, and final validation remain the authors', Gemini's™ assistance was instrumental throughout the writing process.

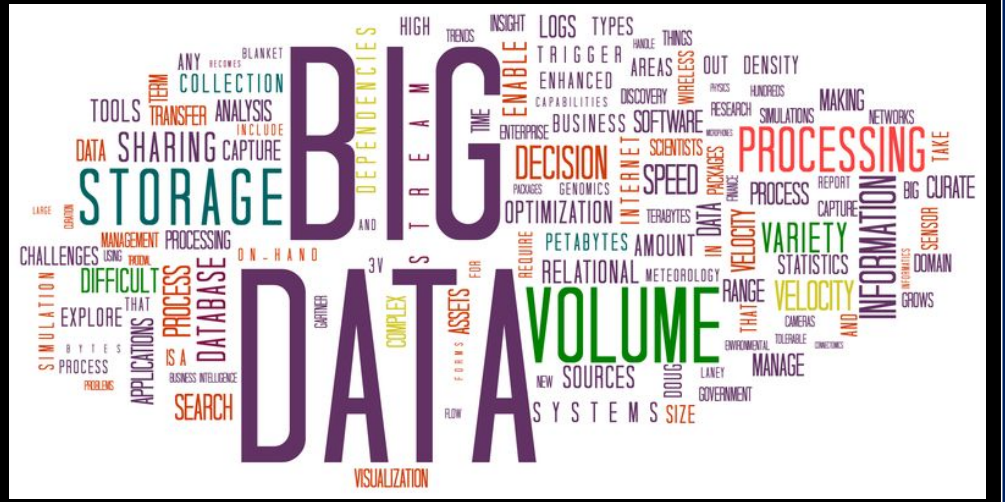
NOTE ON SCOPE

The presentation is not faithful to the article because we allowed ourselves to add recent elements not appearing in the original article.

OBJECTIVE

The goal of this presentation is to illustrate an industry-applied usage and the limitations observed through our perspective.

AIRBUS



Big Data

2 Peta Bytes
Flight Tests data set

End-Users
2 000+

Flight
5 Million+

Time Serie / Flight
+1 Million Signal

5 000+
Flight/Day

AIRBUS

What if finding crucial flight test insights was as easy as asking a question?

AIRBUS

We're Sitting on a Goldmine...

- Flight Test & Design Office systems (TSAS, ECM, etc.) are packed with valuable info.
- Think flight data, sensor readings, important documents, reports...
- This data is crucial for design, testing, and making decisions. HUGE opportunity!

AIRBUS

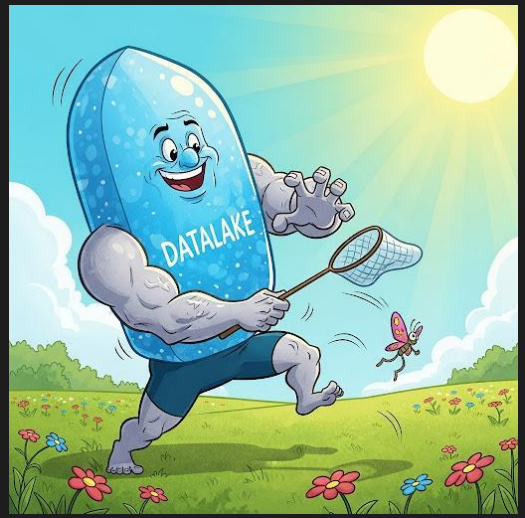
But... Finding the Right Info Can Be a Quest! 😫

- Connecting data between different systems isn't always smooth.
- Getting the exact piece of information you need can feel like solving a puzzle.
- Sometimes, it's just plain PAINFUL and time-consuming!

AIRBUS

Our Mighty Data Lake: Powerful, But a Bit Clunky? 🏋️🐢

- **Datalake Power:** 💪 Handles HUGE data like a champ!
- **The Workout:** 🥵 Feeding & syncing it? A constant marathon!
- **Awesome For:** 🧐 Perfect for those BIG, deep-dive analysis.
- **The Catch?** 🍬 Quick questions? Feels like bulldozer vs. candy wrapper!
- **Result:** 🐢 Small insights can be slow & 💰💰💰 pricey to get.



AIRBUS

Bridging the Gap: AI-Powered Datalakes

The Opportunity: Conversational Data 💬

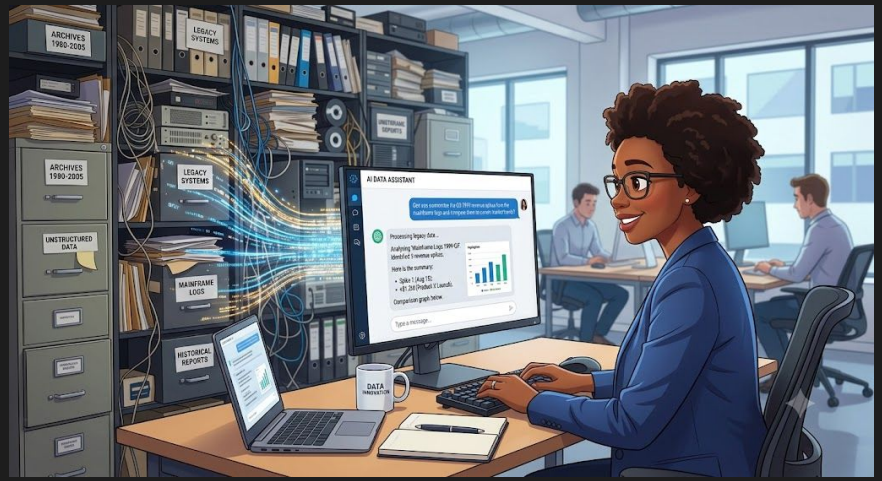
Natural Interaction: Converse directly with data systems.

Instant Insights: Unlock rapid decision-making

The Challenge: Operational Hurdles 🚧

System Diversity: Managing siloed databases

Domain Jargon: Aligning standard LLMs with "Airbus Flight Test" jargon safely.



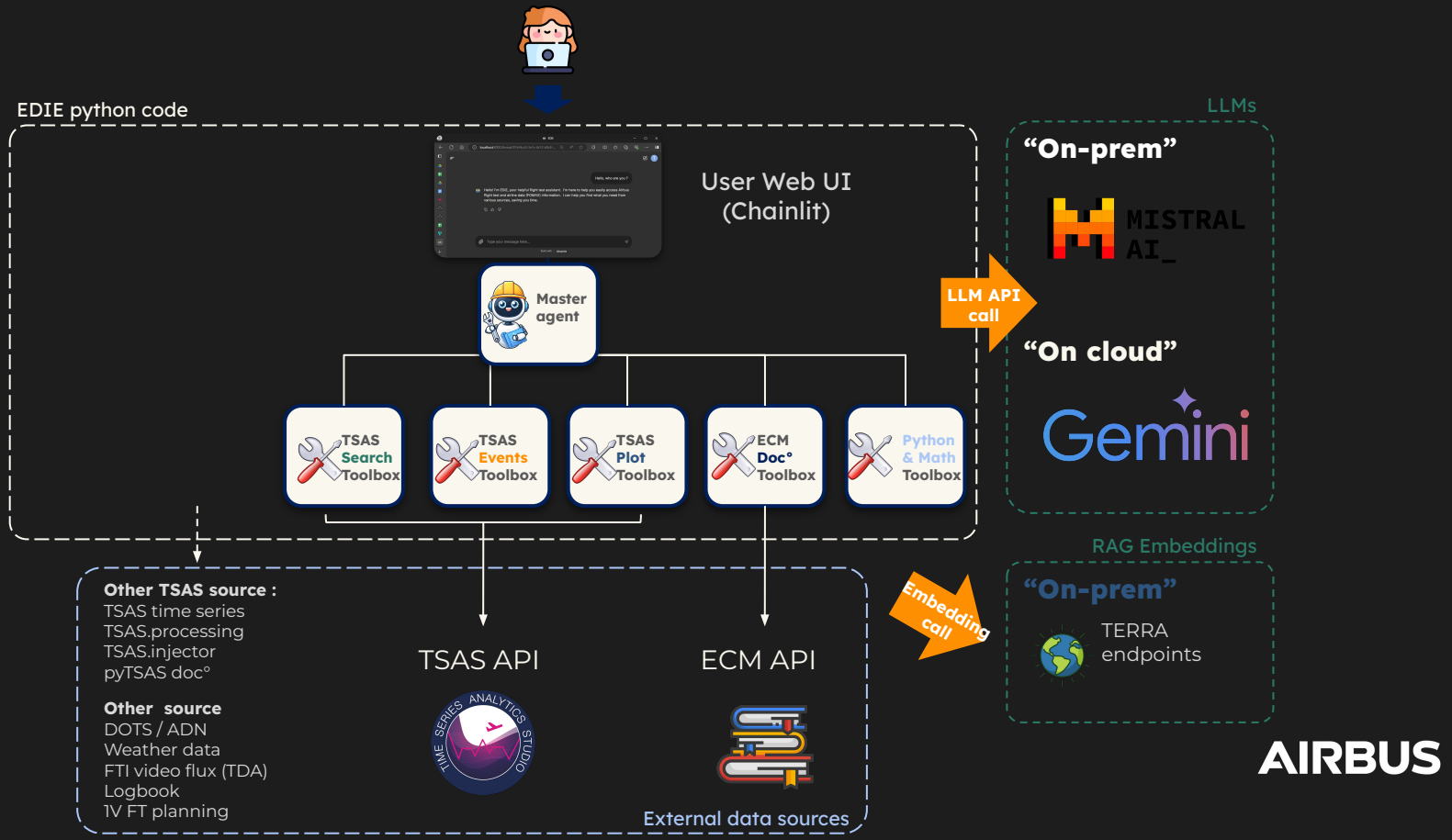
AIRBUS

Chapter 1: Our Expedition Begins: Introducing EDIE

- We started a project called EDIE (Elevating Data Intelligence for Engineering).
- It's a prototype chatbot using AI "agents".
- Our Goals:
  - Make data access EASIER for engineers (less pain!).
  - Learn how this AI stuff actually works in our world.
  - Find cool, new ways to combine data.

AIRBUS

Chapter 1: EDIE's Superpower: The Agentic Mind



Chapter 1: The Right Model, The Right Language

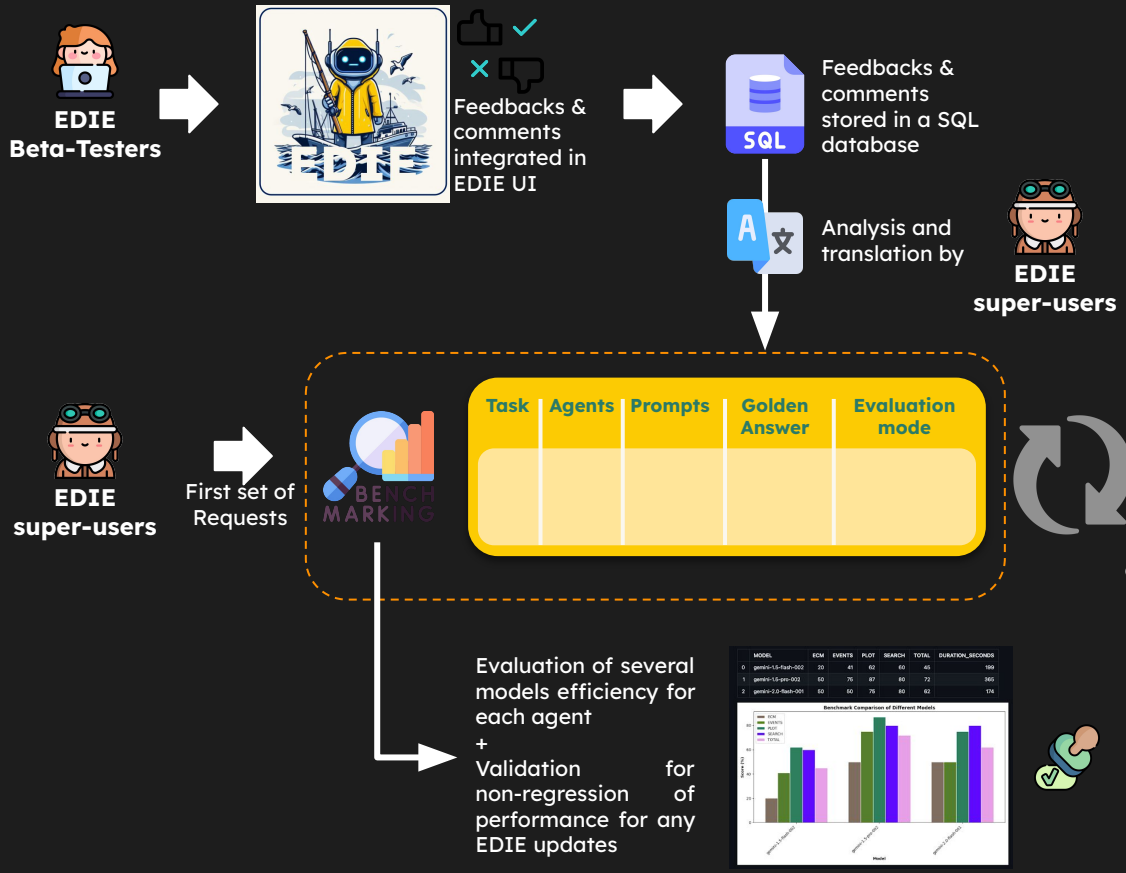
- Model matters: Early smaller models (Mixtral) struggled with multi-step tool use. Gemini Pro was the turning point — better reasoning, reliable tool orchestration.

- Precise APIs: Many small, focused tools beat one monolithic interface.

- Guided databases: Examples + scoped data = accurate queries.

- RAG for aerospace: Handling jargon, complex tables, even handwritten notes — where contextual understanding is critical.

Chapter 1: How EDIE was evaluated



AIRBUS

AIRBUS

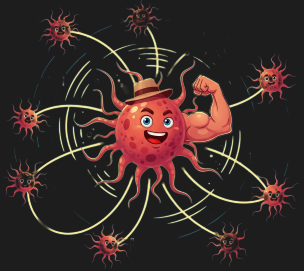
Chapter 1: EDIE's Flight Log - What We Learned

Key Takeaways:

From Unit tests to Users stats !

Users feedback is essential for benchmarking

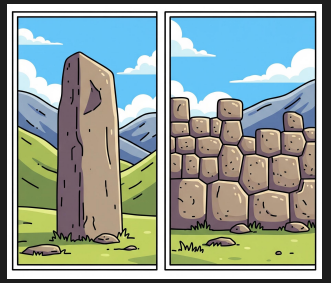
Change of IT Development mindset from deterministic unit tests to user-driven statistical tests



Model Power Matters, Agentic Arch. is crucial

A capable Model like GEMINI Pro makes a big difference

Multiple agents architecture with a standardised protocol exchange is crucial for rapid growth.



Interface Quality is Paramount

How you connect the AI to data is critical for reliability.

Change of IT Development mindset from monolithic to networked of applications

Chapter 2: TSAS assistant : a new story

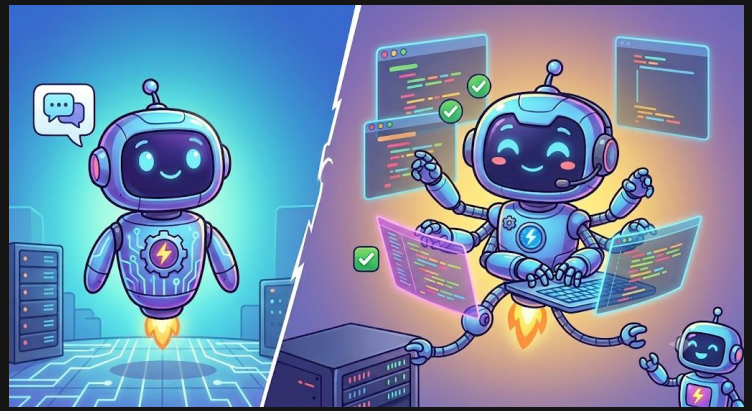
AIRBUS

RAG is hitting a wall

Implementing effective RAG systems is costly and complex, heavily bottlenecked by the need for human experts, and difficult to monitor or reproduce reliably.

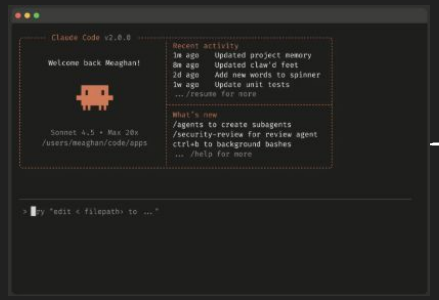
Coding Agents are taking off

Autonomous agents bypass many of these hurdles by using reliable, specialized models that can test their own work, learn from failures, and iteratively correct their own code.



Chapter 2: TSAS Assistant - Tools and skills

Built-in Agent Skills



Coding agent

Metadata : aircrafts, flights, parameters etc

Big data Jobs

Search and extract events

Extract time series

TSAS

Bring Your Own Skills

AIRBUS

Chapter 2: Outcomes

Key Takeaways:

Application Testing Strategy

Logic Tier: Subject to standard testing practices: unit, integration, and functional tests.

Semantic Tier: detailed benchmark ensure non regression.

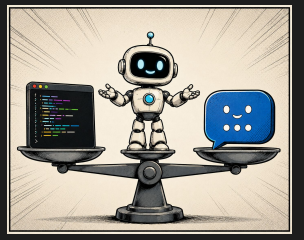
⚠️ **Challenge: Gathering and structuring this benchmark data is difficult.**

Semantic - Logic tiers Interfaces

The quality relies on :

- Functional consistency
- Avoiding functional overlap
- Eliminating ambiguity.

Highlight : integrating AI in our legacy systems pointed relevant functional inconsistencies



Next-Gen UI/UX

New Human-Computer Interfaces are evolving:

The Balance: Striking the right balance between the flexibility of a coding agent and the intuitive ergonomics of a chatbot.

⚠️ **Challenge: Addressing emerging issues related to UX/UI design, cost control, and security**

From Prototype to Production: What Stands in the Way

💸 Cost Control

"When everyone has an agent, who's paying the bill?"

🔒 Privacy

"Your data is in the cloud — but should it be?"

🛡️ Security at Scale

"One prompt injection away from chaos."

AIRBUS

Outcomes (1/2)

AIRBUS

Application Testing Strategy

- **Semantic Tier:** detailed benchmark ensure non regression.
- **Logic Tier:** Subject to standard testing practices: unit, integration, and functional tests.

⚠️ **Challenge: Gathering and structuring this benchmark data is difficult.**

Model Performance

General performance improvements in small open-source models.

- **Reasoning:** Capabilities are now highly capable and practical for real-world use.
- **Orchestration:** Tool orchestration is highly efficient.
- **Limitation:** Still limited for complex queries compared to premium models (Gemini, Claude).

⚠️ **Challenge: Navigating trade-offs in cost, privacy, and dynamic routing based on complexity.**

TSAS Assistant: Navigating AI Challenges & The Rise of Coding Agents

AIRBUS

1. Limitations & Challenges of RAG

SME Bottlenecks: High friction in securing dedicated time from Subject Matter Experts (SMEs) for knowledge curation and validation.

The "Generic RAG" Trap: Attempting to design a one-size-fits-all RAG architecture drastically increases system complexity and development costs.

Engineering Hurdles:

- ***Observability:*** Difficult to trace, monitor, and debug the AI's retrieval and reasoning pathways.
- ***Reproducibility:*** Challenging to guarantee consistent, repeatable outputs across multiple prompts.

2. Emergence & Strengths of Coding Agents

Autonomous Execution: Built-in "fail and retry" capabilities allow agents to test logic and handle exceptions without human intervention.

Iterative Self-Correction: Seamless feedback loops enable the agent to analyze errors, refine its approach, and dynamically fix its own code.

High-Fidelity Models: Leveraging highly reliable, specialized LLMs optimized specifically for robust code generation and syntax accuracy.

Outcomes (1/2)

AIRBUS

Application Testing Strategy

- **Semantic Tier:** detailed benchmark ensure non regression.
- **Logic Tier:** Subject to standard testing practices: unit, integration, and functional tests.

⚠️ **Challenge: Gathering and structuring this benchmark data is difficult.**

Outcomes (2/2)

AIRBUS

Semantic - Logic tiers Interfaces

The quality relies on :

- Functional consistency
- Avoiding functional overlap
- Eliminating ambiguity.

Highlight : integrating AI in our legacy systems pointed relevant functional inconsistencies

Next-Gen UI/UX

New Human-Computer Interfaces are evolving:

- **The Balance:** Striking the right balance between the flexibility of a coding agent and the intuitive ergonomics of a chatbot.

⚠️ **Challenge: Addressing emerging issues related to UX/UI design, cost control, and security.**

Navigating trade-offs in cost, privacy, and dynamic routing based on complexity.

From Prototype to Production: What Stands in the Way

Navigating trade-offs in cost, privacy, and dynamic routing based on complexity.

The Next Frontier: Cost, Privacy & Security at Scale

From Prototype to Production: What Stands in the Way

💸 Cost Control

- LLM inference cost scales with usage — every request has a price
- How do you govern consumption when every user has an agent on their machine?
- Need for smart routing: cheap models for simple tasks, powerful ones for complex ones 🔀

🔒 Privacy

- Foundation LLMs are cloud-hosted — data leaves the perimeter ☁️
- On-premise or private deployment is costly and limits model quality
- What data can we send? What must stay local? 🏠

🛡️ Security at Scale

- Prompt injection: malicious inputs hijacking agent behavior 💉
- Data leakage through model outputs or tool calls 🚨
- Attack surface grows with every new user and integration 📈