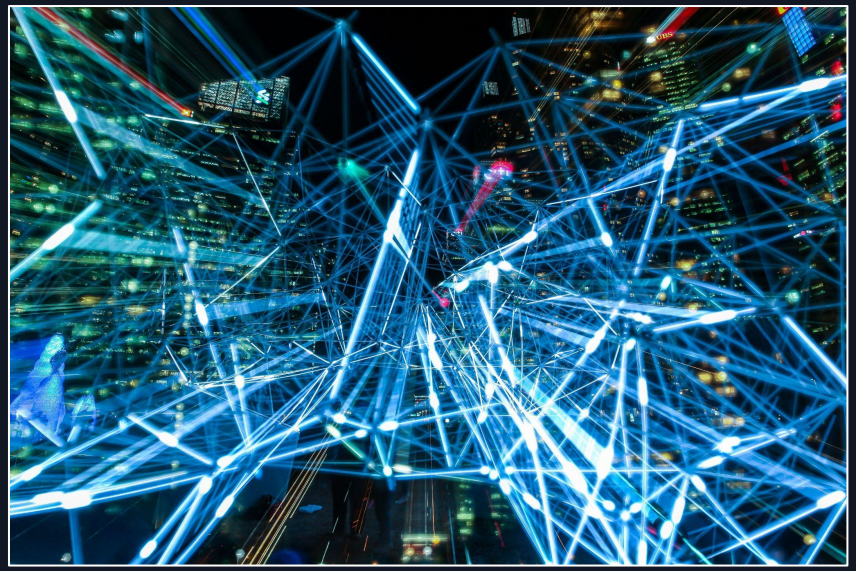
SysML v2 in 2026: Useful Now - Transformative with AI by 2027+

Rigorous systems engineering and the coming AI inflection -
balancing token cost, performance, and system understanding

Presenter: Gregor Schütz, Cob Bar Systems



At Least Three Ways to Build Complex Systems

None is universally "right" >> context and mission define the winner

Agile

Business / software-first

- Financial or product-market target
- Many small bets; kill losers early
- Works well for pure software
- Expensive when hardware is in the loop > feedback comes too late
- No technical north star required
- Risk: optimises for speed, not mission

V-Model

Safety / traceability-first

- Exhaustive requirements upfront
- Organisation optimises for efficiency
  → avoids risk at every level
- "More parts" seen as thoroughness
- Works in highly regulated contexts

- process becomes the goal, not the mission

Mission-Driven

Physics-limit / mission-first

- Goal set at edge of what physics allows
- Leader defines mission + rough roadmap
  so engineers believe it is possible
- Responsible Engineers own requirements
- Build → fly → learn; failures are data
- Risk: needs visionary leadership and tolerance for public failure

SysML v2: Game Changer or Silo?

The tension between structured systems engineering and the capability of AI

The V-Model

- Capture requirements upfront
- Decompose → design → verify
- Full traceability at every level
- Works: NASA, Airbus, etc.
- Critiqued for being slow and document-heavy in practice

vs

→ *Can SysML v2 bridge these two worlds, or do we need something fundamentally different?*

✅ **OMG Finalized — Sept 2025**

What SysML v2 Actually Changes

Formal Semantics + Textual Syntax

v1 was a UML profile — imprecise, tool-dependent.

v2 uses KerML as its kernel: rigorous, unambiguous.

Both graphical AND textual notation — for the first time a machine can read and reason over a SysML model.

→ Opens the door to AI-assisted modelling

Standard API & Interoperability

v1 had no standard data exchange — every tool was an island.

v2 ships a standardized REST API and exchange format.

Connects to PLM, simulation, requirements, and digital twin platforms without custom glue.

→ The "digital thread" could become a reality

14 Diagram Types + Verification

v1 had 9. v2 has 14 — allocation and verification diagrams are now first-class citizens.

Parametric analysis, constraints, and simulation integrated directly into the model.

Traceability from requirement → design → test → evidence.

→ Full lifecycle in one model

⚠️ Open question: Is SysML v2 too SE-centric for AI-Human collaboration, or is its formal structure exactly what WE need as a shared language?

Complement Human Engineering Craftsmanship

"The best part is no part. The best process is no process."

1

Challenge Requirements

Every requirement must have a name attached.
If nobody owns it -> delete it.
Requirements are often wrong — treat them as hypotheses.

2

Delete Parts & Processes

If you're not adding back ≥10% of deleted items, you're not deleting enough.

3

Simplify & Optimise

Optimising a part that should not exist is wasted effort.
Global optimization (systemness) instead of local improvement.

-> Simplicity is a forcing function for quality.

💡 Human-AI collaboration with SysML v2

This methodology starts with organizing, improving and questioning requirements.

People seem to be reluctant of deleting things.

Human cognitive capabilities are limited and often miss the non-obvious interconnection between parts - process.

SysML v2's machine-readable model could let AI agents to help at scale: flag orphaned requirements, detect redundant constraints, and surface hidden assumptions, write complete test procedures, etc

Enter AI: What Changes and What Doesn't

"Compilable Product Development" — agents generate SysML v2 text; models become runnable infrastructure

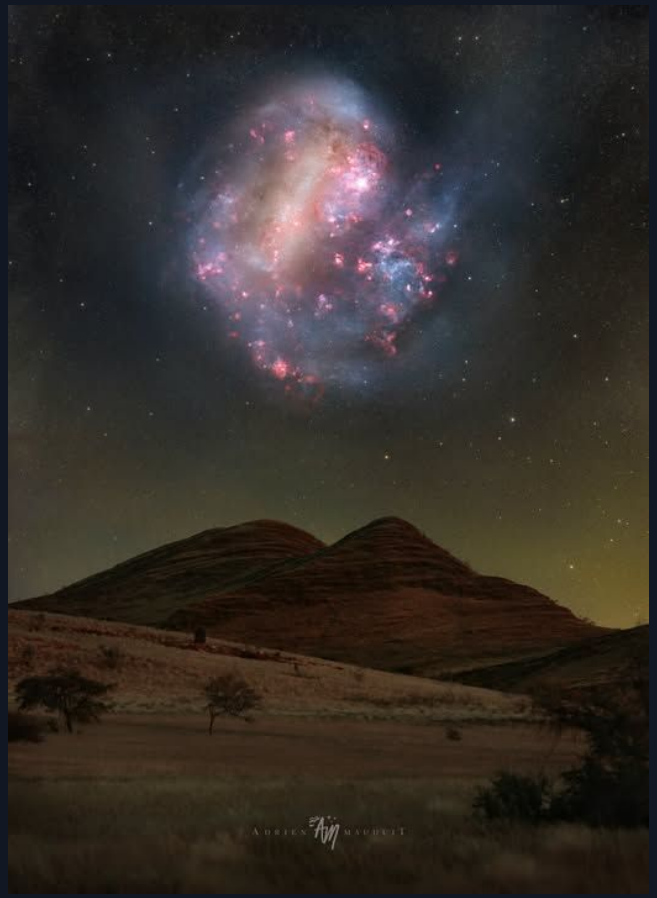
🤖 What AI Changes (especially with SysML v2)

- LLMs can read / write SysML models
- Flag orphaned / contradictory requirements, improve
- Run impact analysis: "if I change this interface, what breaks?"
- Automate verification evidence linking
- Agents traverse the digital thread across tools via the standard API

⚠️ What AI Cannot Replace (yet)

- Visionary mission definition ("attribute purpose and motivate" still needs a human)
- Walk the rough roadmap that makes the impossible feel achievable
- Physical intuition — knowing what the model gets wrong about reality
- Organisational culture change → don’t overestimate organizational speed (SysML v2 doesn't fix bad incentives)
- Hardware-reality feedback loops (some things still need to fly/fail)
- People even do less of the “boring work”, e. g. fundamental requirement understanding and questioning

What is your opinion?



Option A · SysML helps humans

Option A) Useful Step, Not a Paradigm Shift

"SysML v2 is a step forward for humans doing systems engineering, but it does not fundamentally change how projects are executed. AI benefit is marginal or indirect."

✅ Arguments For

- Better notation → clearer human communication
- Interop API reduces integration friction
- Formal semantics catch errors earlier
- Helps large orgs modernise
- Reduces tool lock-in over time
- SysML v2 may help humans collaborate better with AI — a stepping stone

⚠️ Caveats

- AI agents don't need SysML v2 to be useful — they work fine with natural language requirements today
- Organizations won't adopt SysML regardless of v2 improvements
- The bottleneck is culture, not notation
- "Better UML" doesn't change incentives

→ Conclusion under Option A: worth adopting in traditional SE orgs; limited systemic impact on how humanity builds complex systems.

Option B · AI + humans need SysML v2

Option B) The Game Changer Hypothesis

"AI and humans need SysML v2 to work efficiently together. v2 is therefore a game changer in how projects are executed — with impact far beyond the systems engineering community."

🔗 The Mechanism: Why This Could Work

1. SysML v2 textual syntax is the first SE modelling language LLMs can natively read and write
2. The standard API creates a live, queryable model — agents can interrogate the whole system
3. Agents can improve system design (requirements, elements and interfaces) at scale across thousands of elements
4. Automated impact analysis, traceability, and verification
5. Result: enormous iteration speed available to organisations
6. Reach extends far beyond traditional SE-industries

📡 Early Evidence (2025)

- "Compilable Product Development" trend: AI generates SysML text → compiled to CAD/simulation
- Digital engineering labs using SysML v2 API + LLM agents for automated requirement review

→ Option B requires: mature tooling (2027+), culture shift, and AI models that can reason over large formal models. High upside, high friction.

Option C · Humans won't design much longer

Option C) AGI Makes SysML Irrelevant

"AI does not benefit from SysML v2. AGI arrives within a decade. Humans won't be designing challenging systems much longer — so SysML v2 has very limited long-term value."

🤔 The Core Argument

- AGI will design systems far better than human-readable notation allows
- SysML v2 is still a human-centric language — optimised for human brains
- AI doesn't need formal MBSE — it can reason from first principles
- Adoption timeline of SysML v2 (2027+) may collide with AGI timeline
- Investing in SysML v2 training and tooling may have negative ROI if the window is <5 years

💬 Counter Arguments

- AGI timelines are deeply uncertain
- Even with AGI, humans need to understand and certify complex systems
- Safety-critical domains (infrastructure, aviation, etc.) require human-interpretable artefacts
- SysML v2 may be exactly the interface layer between AGI decisions and human oversight
- The "no value" case assumes AGI replaces systems engineers fully — unlikely in regulated industries (safety)

→ Option C is a provocative bet

Verdict & Open Discussion

A — Useful, Not Transformative

SysML v2 improves human SE practice. AI benefit is indirect. Adoption makes sense for large orgs — but does not change how humanity builds complex systems.

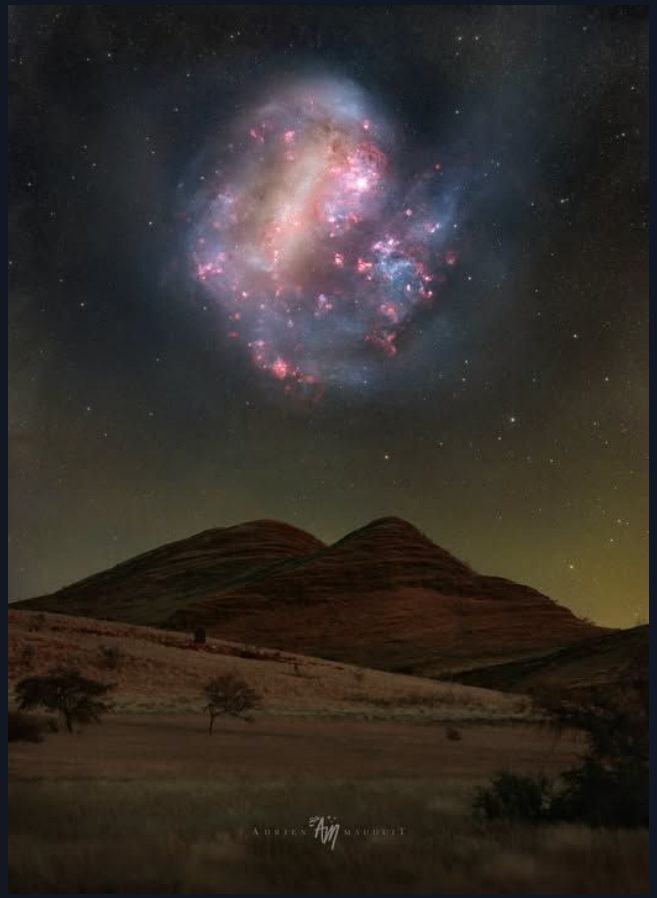
B — Game Changer (with AI)

SysML v2 + LLM agents = the first time a machine can hold and reason over a full system model. This unlocks mission-driven iteration speed for orgs that cannot burn hardware. Impact reaches far beyond SE.

C — Irrelevant (AGI horizon)

AGI makes human-readable modelling notation obsolete before SysML v2 matures. Limited investment window. The safer bet is skipping to AI-native engineering.

What is your opinion?



Thank You

For your time, questions, and participation in the discussion.

→ Find me during the coffee breaks or at the apéro

→ *Time will tell which path was right — and we will find out together.*

For every important project ahead:
→ *Success is probable. Excitement is guaranteed.*