WORLD OF
INDUSTRY
TECHNOLOGY
& SCIENCE

From Prompt to Model

AI-driven requirements engineering in a model-based world

Walter van der Heiden

VDH Informatica BV · IBM Champion 2021–2026

Embedded Software + AI · WoTS 2026 · Jaarbeurs Utrecht · 22 September 2026

FHI Federatie van technologiebranches

THE PATTERN

We have been here before

1865 The motor car

British law: a man had to walk 60 yards ahead of any self-propelled vehicle carrying a red flag. 2 mph in town. Repealed in 1896.

Locomotives Act 1865

1881 The telephone

“I find my telephone almost useless, as I occupy more time in communicating through it than I should do if I went to the offices.”

Clifford Dunn, solicitor, Leeds

1995 The internet

“I predict the Internet will soon go spectacularly supernova and in 1996 catastrophically collapse.”

Robert Metcalfe — who invented Ethernet

2007 The smartphone

“There’s no chance that the iPhone is going to get any significant market share. No chance.”

Steve Ballmer, CEO of Microsoft

Every one of these was a serious person making a reasonable argument at the time.

WoTS
WORLD OF INDUSTRY TECHNOLOGY & SCIENCE

1954

And our own profession did it too

"It is difficult to convey … the strength of the skepticism about 'automatic programming' in general, and about its ability to produce efficient programs in particular, as it existed in 1954."

"… if FORTRAN were to translate any reasonable source program into an object program only half as fast as its hand coded counterpart, then acceptance of our system would be in serious danger."

John Backus, "The History of FORTRAN I, II and III", ACM

— **The objection was competence.** Not "it will take my job" — "it cannot possibly be as good as I am."

— **The compiler had to prove itself** against hand-written assembly, within a factor of two.

— **It did. And then we stopped reading the assembly.**

You already let a machine write your code.
You have since 1957.

WORLD OF INDUSTRY TECHNOLOGY & SCIENCE

13 SEPTEMBER 2026 — FIVE DAYS AGO

But this time the warning comes from inside the building

Dario Amodei	Sam Altman	Demis Hassabis	Elon Musk
Anthropic	OpenAI	Google DeepMind	xAI

All four warning in public in the same week. Amodei's essay argues the field should slow down how fast it improves what these models can do.

WHY THIS ONE IS NOT 1865

- Ballmer was warning about someone else's product.
- Metcalfe was warning about someone else's network.
- Nobody at Ford asked for a man with a red flag.

The people shipping it are asking for the brakes. That is new.

AND WHY PLENTY DO NOT BUY IT

- Critics call it regulatory capture: rules incumbents can afford and startups cannot.
- The US–China race makes "slow down" look unrealistic.
- Many researchers take the doom scenarios less seriously than the headlines do.

That argument is about frontier models and superintelligence. It is not the argument in this room.
Ours is smaller, and answerable: what do you do on Monday with the code it already writes?

WORLD OF INDUSTRY TECHNOLOGY & SCIENCE

THREE REAL DIFFERENCES

So what is actually different this time?

1

It is probabilistic

Ask the same question twice, get two different answers. A compiler is deterministic. A generator is not — and a build step that is not reproducible is not a build step.

2

It has no oracle

A compiler knows the rules of C. A language model has nothing to check itself against. It cannot tell a correct answer from a fluent one, because to it they are the same thing.

3

It produces our artefacts

Not only the code. The requirements, the design, the architecture, the documentation. The things we used to make are now the output.

Those three are real, and worth arguing about. “It will replace me” is the red flag walking in front of the car.

WHERE WE ARE HEADING

The assistance spectrum

1

Requirements

AI as scribe

structures
what you meant

2

Model

AI as co-architect

proposes states,
blocks, links

3

Code

AI as pair programmer

writes the
function body

4

Vibe coding

AI as implementer

you describe,
it builds

5

Autonomous app

AI as author

it works.
nobody read it

what you no longer read

Every step to the right buys speed. Every step to the right spends comprehension.

WORLD OF
INDUSTRY
TECHNOLOGY
& SCIENCE

STEP 1

Level 1 — AI as requirements scribe

WHAT THE STAKEHOLDER SAID

"The system should handle the brake request quickly and safely, taking the road surface into account."

WHAT COMES BACK

REQ-BRK-041

The brake controller shall acknowledge a brake-by-wire request within 15 ms under nominal conditions.

Priority: safety-critical
Source: stakeholder workshop, 14-08-2026
Verified by: HIL test case TC-BRK-041

! **conflicts with REQ-BRK-038:** "… shall acknowledge within 20 ms."

WHAT IT IS GENUINELY GOOD AT

- Splitting one sentence into atomic, testable requirements
- Naming the ambiguity: "quickly" — how quickly?
- Finding the gap you did not write: what happens on low friction?
- Noticing that it contradicts a requirement you wrote three weeks ago

It cannot tell you whether 15 ms is the right number. That is the part of the job that was never typing.

WoTS
WORLD OF INDUSTRY TECHNOLOGY & SCIENCE

STEP 2

Level 2 — the target is the model, not the text

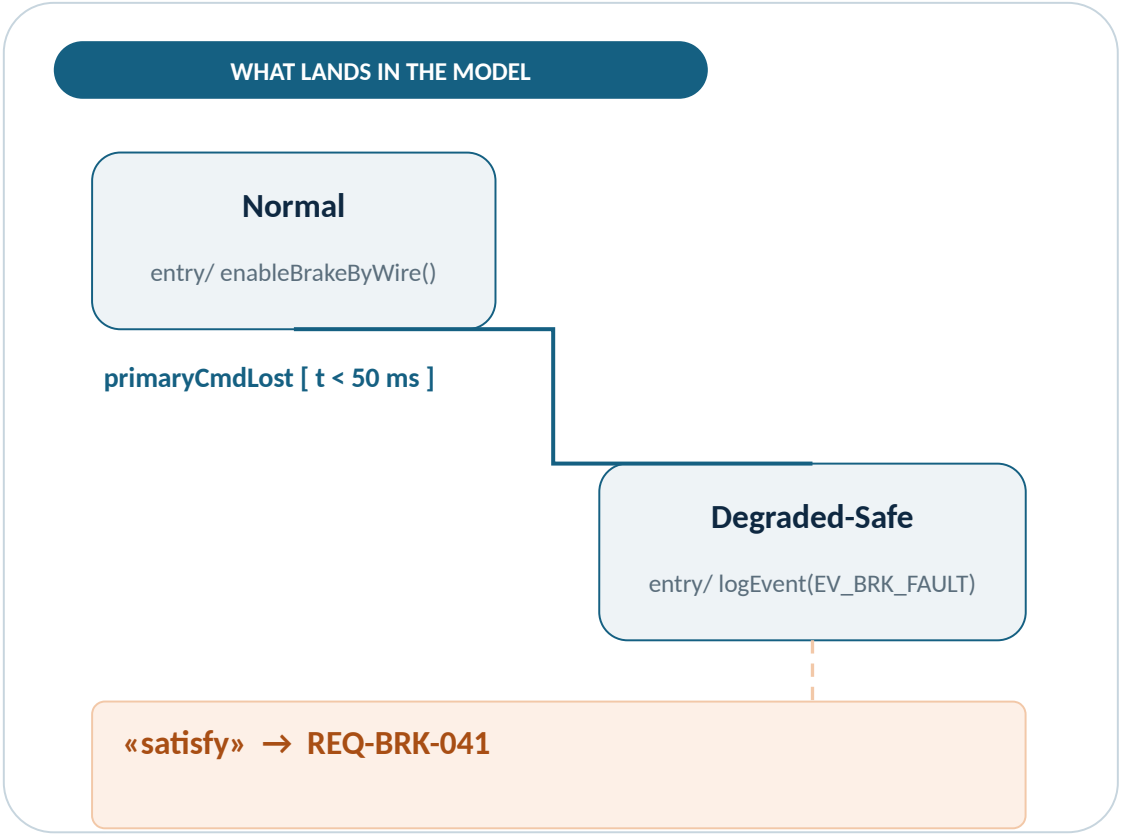
PROMPT

“Add a fault-tolerant state to the brake controller: if the primary command is lost, move to a degraded-safe state within 50 ms and log the event. Link it to REQ-BRK-041.”

— **Same request as before.** The difference is where the answer lands.

— **Not a paragraph** — a state, a transition, a guard, an action, and a satisfy-link to a requirement.

— **Reviewable in seconds** by anyone who reads state machines.



WORLD OF INDUSTRY TECHNOLOGY & SCIENCE

STEP 3

Level 3 — the one you are already using

THE FAMILIAR ONE

- You start typing a function. It finishes it.
- You describe what you want in a comment. It writes the body.
- You glance at it, it looks right, you press Tab.

Copilot, Cursor, and everything like them.

1. **You read it. You did not write it.**

   Those are two different kinds of knowing — and only one of them survives being asked about it six months later.

2. **Accepting takes one keystroke.**

   Reviewing it properly takes minutes. Whenever those two costs are that far apart, you can guess which one wins on a Friday afternoon.

3. **It compounds quietly.**

   One accepted suggestion is nothing. A team, a sprint, a release — and the review debt is already in the repository.

This is where the comprehension gap actually opens. Levels 4 and 5 do not create it — they just make it impossible to ignore.

WoTS

WORLD OF INDUSTRY TECHNOLOGY & SCIENCE

STEPS 4 AND 5

Levels 4 and 5 — and then it just... works

PROMPT

“Build me a CAN bus monitor that warns when arbitration delay exceeds 3 ms.”

2 400 lines of C.
It compiles. The tests are green.

No one has read it.

Not the author — there isn’t one. Not the reviewer — 2 400 lines is a day of work to review properly, and it took four minutes to write.

There is no design.

No architecture document, no interface spec, no state machine. The design exists only as a side-effect of the code.

You cannot ask it why.

You can ask it to explain, and it will produce a fluent explanation. You have no way to check that the explanation matches the code.

WORLD OF INDUSTRY TECHNOLOGY & SCIENCE

THE QUESTION

It compiles. It runs. The tests are green. Now explain it to the assessor.

“We used AI to generate it and we reviewed it” is not an argument. It has no volume, no depth and no denominator.

Marc Thomas takes the verification side of this at 14:00. I want to talk about what you hand him.

WoTS

WORLD OF
INDUSTRY
TECHNOLOGY
& SCIENCE

THE CORE OF IT

Two verification surfaces, same system

	2 400 lines of C	23 states, 41 transitions
Readable in a review?	hours, if anyone really does it	**minutes**
Traceable to a requirement?	by comment, maintained by hand	**structurally, by construction**
Reviewable by a domain expert?	only if they write C++	**yes — they read behaviour**
Executable before hardware exists?	no	**yes — simulate the model**
Claim in a safety case?	“we read it”	**“it satisfies R1–R41 by construction”**

Same behaviour. One of these you can defend in a room full of people who did not write it.

WORLD OF INDUSTRY TECHNOLOGY & SCIENCE

THE CLAIM

A model is a human-readable, machine-checkable statement of intent.

Code is what the machine does. The model is what we agreed it should do.

When AI writes the code, that difference stops being academic.

READABLE BY A HUMAN

CHECKABLE BY A TOOL

TRACEABLE BOTH WAYS

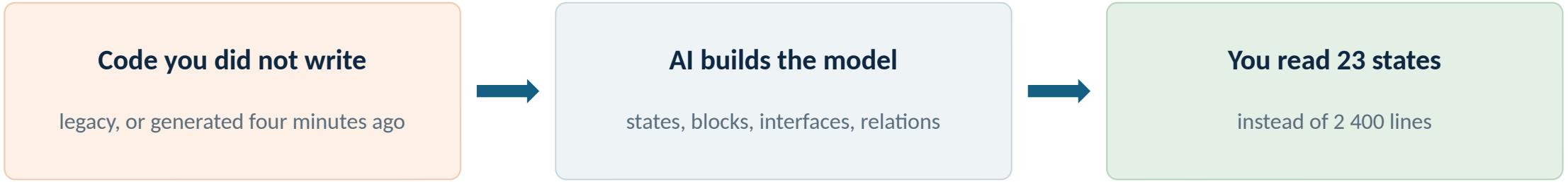
WoTS

WORLD OF INDUSTRY TECHNOLOGY & SCIENCE

THE USEFUL REVERSAL

Modeling is how the AI shows its work

Most of this talk runs requirements → model → code. The direction that surprised me runs the other way.



— **Two uses, same mechanism.** Understanding the code you inherited, and understanding the code the assistant just produced — these turn out to be the same problem.

— **Ask for the model, not the explanation.** An explanation in prose is another fluent artefact you cannot verify. A model you can simulate, check for consistency, and hold against the requirements.

— **This is where the sceptics come round.** Not “the AI writes code for me”, but “the AI finally documented the thing we have been afraid to touch since 2014”.

WORLD OF INDUSTRY TECHNOLOGY & SCIENCE

TRACEABILITY

The chain that survives the audit

Every arrow is a human decision. The AI can maintain the links — it cannot own them.

Six months later, in a review, you do not defend 2 400 lines. You defend one transition, and you show the chain it hangs from.

WORLD OF INDUSTRY TECHNOLOGY & SCIENCE

TIPS & TRICKS

The model makes the AI better, too

1

The model is external memory

The assistant does not need your 40-state machine in context. It needs the three states that matter. Slice the model and pass the slice — do not paste the whole thing and hope.

Long contexts degrade in the middle. A model lets you avoid needing one.

2

The model is a constraint

A blank .cpp file gives the assistant infinite freedom. A state machine that must stay consistent with the 40 states already there gives it almost none. Fewer degrees of freedom, fewer inventions.

The strongest anti-hallucination measure I know is a well-formed model.

3

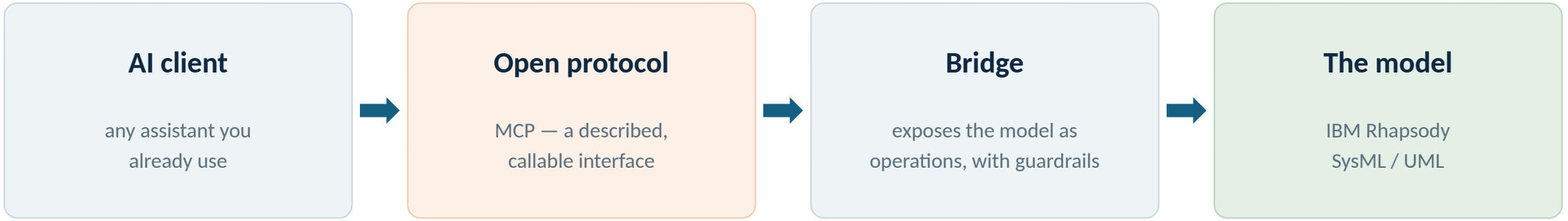
The model is the source of truth

When the code and the model disagree, the model wins and the code is regenerated. The moment you let that slip, you are back to reading code to find out what the system does.

This is a discipline decision, not a tooling one.

THE PATTERN

In practice: how the assistant reaches the model



read element · **create element** · **create trace link** · **run automation** · **query the model**

— **Open protocol, not a closed integration.** The model outlives whichever assistant is fashionable this year — swap the client, keep the model, keep the trace.

— **I use Rhapsody and SAM because that is my daily work.** The pattern is the point: a model, a described interface, a client. Build it on whatever stack you already own.

— **Neil Langmead goes much deeper into MCP at 13:30.** If this plumbing interests you, stay in the room.

WORLD OF INDUSTRY TECHNOLOGY & SCIENCE

HONEST REPORT

What actually works today

✓ Build a first model from a requirements document

Not the final model. A structured starting point, in minutes rather than days.

✓ Requirements coverage analysis

Which requirements does the model not satisfy? This alone has found gaps I missed.

✓ Reverse-engineer legacy code into a model

The subsystem everybody avoids becomes something you can read and discuss.

✓ Document an existing model

The task everybody postpones, done well enough to be worth correcting.

✓ Impact analysis before a change

What does this touch? Traversing relations is exactly machine work.

✓ Teach the tool through worked examples

Faster onboarding than a 400-page manual. The learning curve was always the barrier.

THE OTHER HALF

What does not work — yet

Context

It cannot see the whole model. Large models must be sliced, and slicing well is a real skill that nobody teaches.

Determinism

Same prompt, different answer. Fine for a proposal you are going to review. Not acceptable as a build step.

Confident nonsense

It will produce a perfectly well-formed state machine for a system it has completely misunderstood — and it will look right.

None of these is a reason not to use it.

All three are reasons to keep the model in the middle and the engineer at the end. Which is, conveniently, where they were already.

WoTS
WORLD OF
INDUSTRY
TECHNOLOGY
& SCIENCE

WHAT YOU WOULD HAVE SEEN

The demo lives in the Development paviljoen

1

A raw requirement

One sentence from a workshop. Unmeasurable, like they all are.

2

The AI structures it

And asks about the ambiguity instead of quietly resolving it.

3

It lands in the model

A Rhapsody element with a «satisfy» link, not a paragraph of text.

4

The trace pays off

Open it from the other end. Change something. Ask what it touches.

Come and watch it run — Development paviljoen, straight after this session.

Thirty minutes is not enough to do it live and do it honestly. I would rather show you properly, with time for your questions.

youtu.be/_50kUVFoPTc

WORLD OF INDUSTRY TECHNOLOGY & SCIENCE

What this does to the job

WAS		IS BECOMING
Writing the model	→	**Deciding whether the model is right**
Typing requirements	→	**Owning the intent behind them**
Reading code	→	**Reading structure**
Knowing the tool	→	**Knowing the domain**

The part of this job that was typing is going. The part that was judgement is growing. I have not met an engineer who got into this for the typing.

CLOSE

Three things to take away

1

The fear is old. The differences are real.

Probabilistic, no oracle, produces our artefacts. Argue about those three — not about whether it is coming.

2

Text alone does not survive an audit.

The model is the smallest thing a human can read that still says what the system must do.

3

Open protocols beat closed integrations.

Your model has to outlive whichever assistant is fashionable this year. Keep the model; swap the client.

STAYING IN THIS ROOM?

13:30 Neil Langmead — MCP, DSM analysis and architectural coupling. **14:00 Marc Thomas** — verifying what the agent produced.
And the live demo: **Development paviljoen, straight after this session.**

WoTS

WORLD OF
INDUSTRY
TECHNOLOGY
& SCIENCE

Thank you

Walter van der Heiden

VDH Informatica BV · IBM Champion 2021–2026

walter@vdh-informatica.com

rhapsody.blog · autosar.blog

The live demo runs in the Development paviljoen — come and watch it there, straight after this session.

Recording: youtu.be/_50kUVFoPTc

WORLD OF INDUSTRY TECHNOLOGY & SCIENCE