

From Space to Software: Reliability Lessons 40 Years After Challenger



Robert Barron
SRE Architect, Office of CIO

IBM

flyingbarron on social media
https://flyingbarron.com

brobert@il.ibm.com

flyingbarron

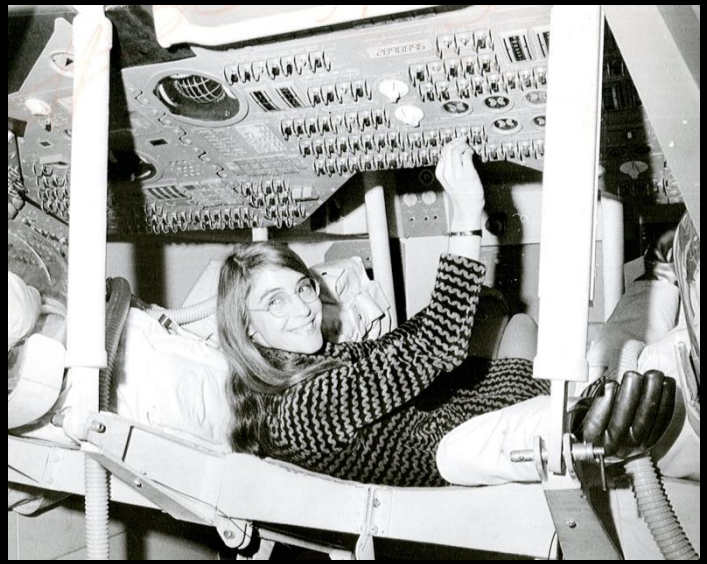
SRECON EUROPE MIDDLE EAST AFRICA

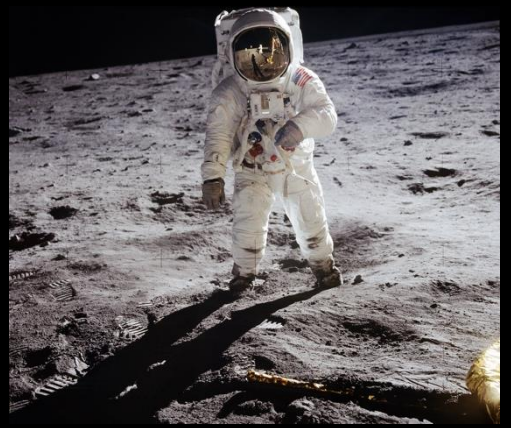
IBM

Margaret Hamilton (1936-2026)

- Coined the term Software Engineering
- NASA's Exceptional Space Act Award:
  "*Her concepts … became the foundation for ultra-reliable software design*"





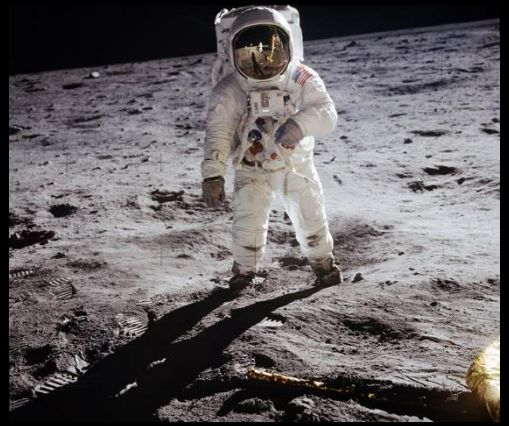


Margaret Hamilton (1936-2026)

- Coined the term Software Engineering
- NASA's Exceptional Space Act Award:
  "*Her concepts … became the foundation for ultra-reliable software design*"





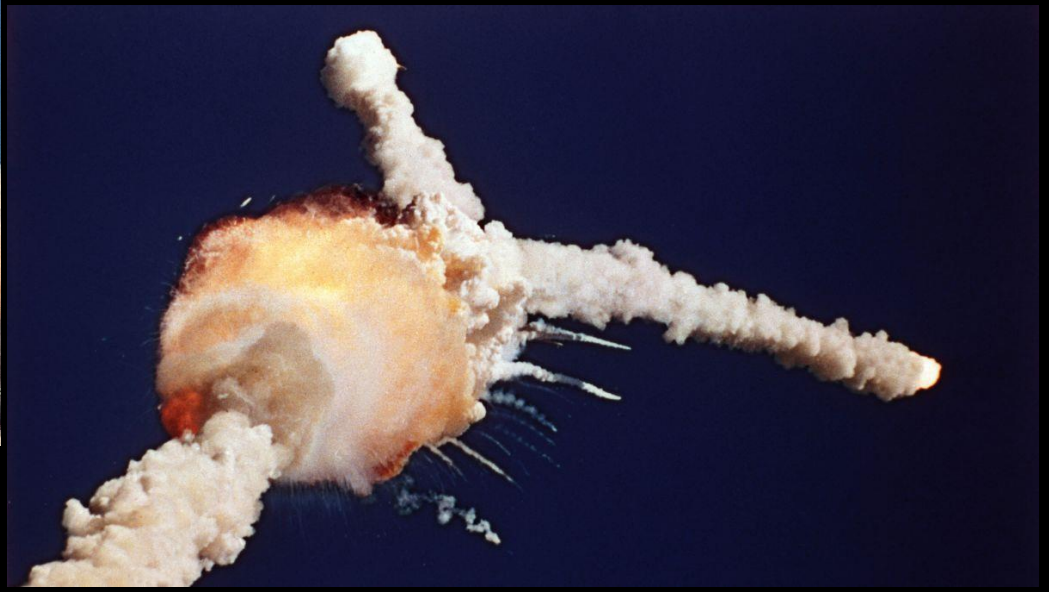


25th Space Shuttle launch – 28th January 1986



73 seconds after launch





The Promise of the Shuttle

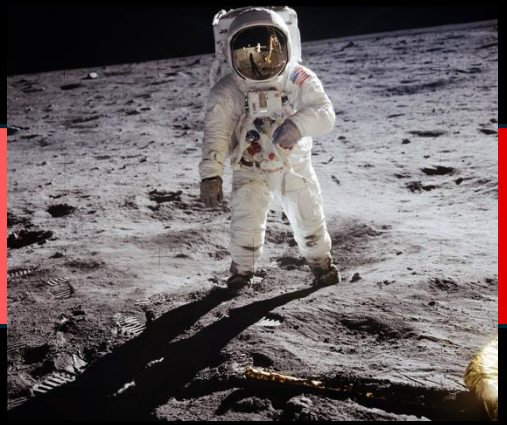






~30 years







20 years

1982 – It's Operational!

First Four Flights
2 pilots
No production payload
Telemetry
Ejection Seats





Pressure to achieve promise



The Daily W

December 31, 1953

50,000,000 sky travellers

Only 50 years after the first aeroplane took flight, 1953 saw over 50 million people flying. Re... foll... im...

1953



Hollywood Sun-Tattler

HOME EDITION ★★★ Home Delivered 90c a Week Call 925-5511

VOL. 52 —NO. 86 HOLLYWOOD, FLORIDA FRIDAY, APRIL 10, 1981 72 PAGES IN 4 SECTIONS

LAUNCH DELAYED

Space Shuttle Countdown Halted Minutes To Lift Off

Flight Reset For Sunday

BY AL ROSSITER JR.
UPI Science Editor
KENNEDY SPACE CENTER
CAPE CANAVERAL, FLORIDA

"The crew is in fine shape," said George Abbey, director of flight crew operations. "They had a good rest last night. We're looking for a real good flight."

They are the first Americans to venture into space since 1875, and the first to fly a spaceship not preceded by an unmanned test flight.

SUCCESSFUL COMPLETION of the 54 ½-hour orbital flight will open a new era of lower cost space

1981



Challenger mission STS-51-L

- First “regular person” to be launched into space – Christa McAulife, a high school social sciences teacher.

- Scheduled for
  - ~~July 1985.~~
  - ~~November 1985.~~
  - ~~January 22 1986.~~
  - ~~January 25 1986.~~
  - January 28 1986.

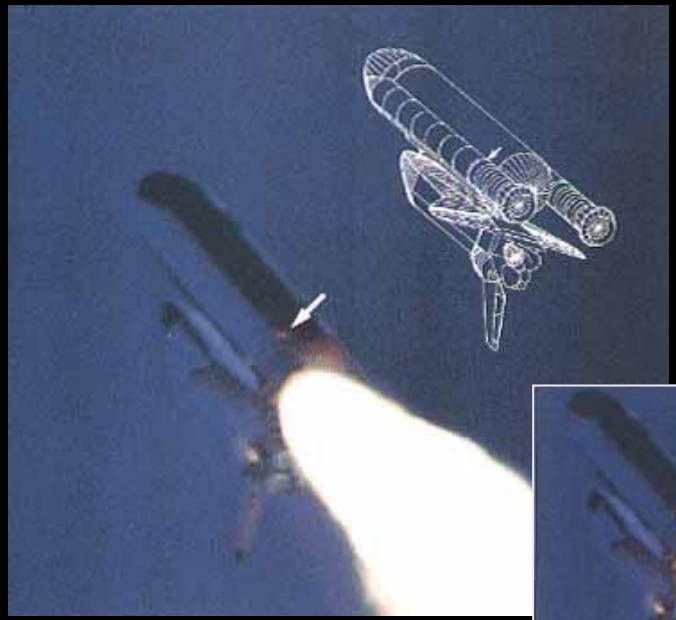
  Coldest launch ever!

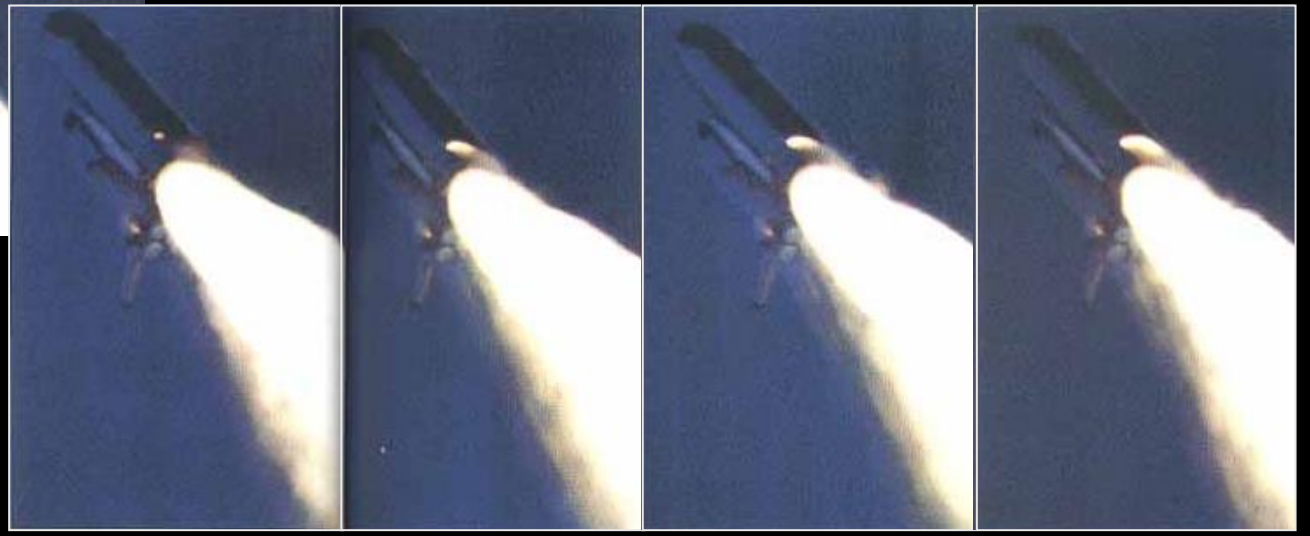


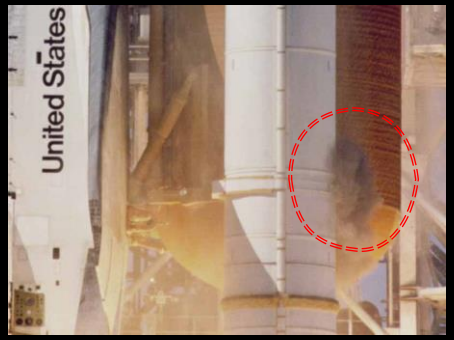
T+72 seconds
“Challenger, go at throttle up.”


“Challenger, go at throttle up.”
T+73 seconds

A small flame

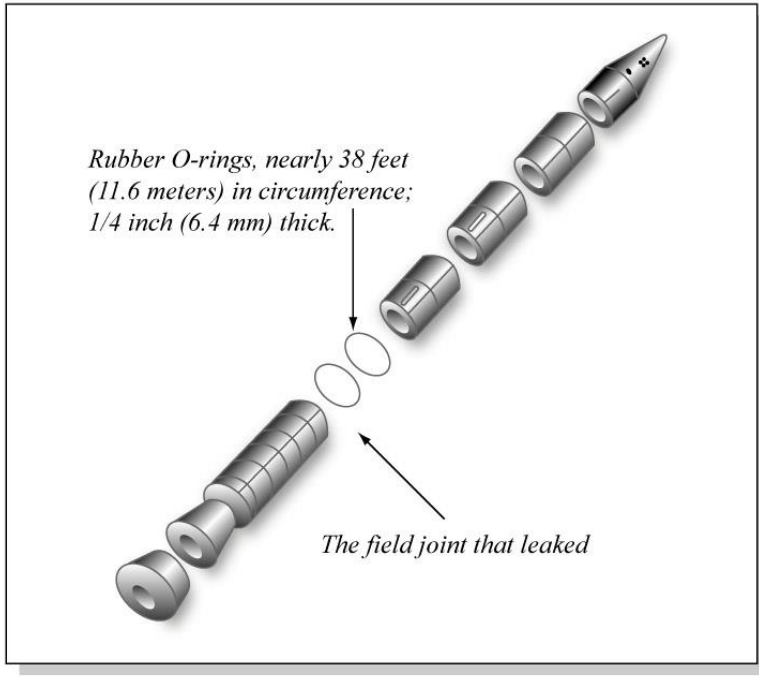




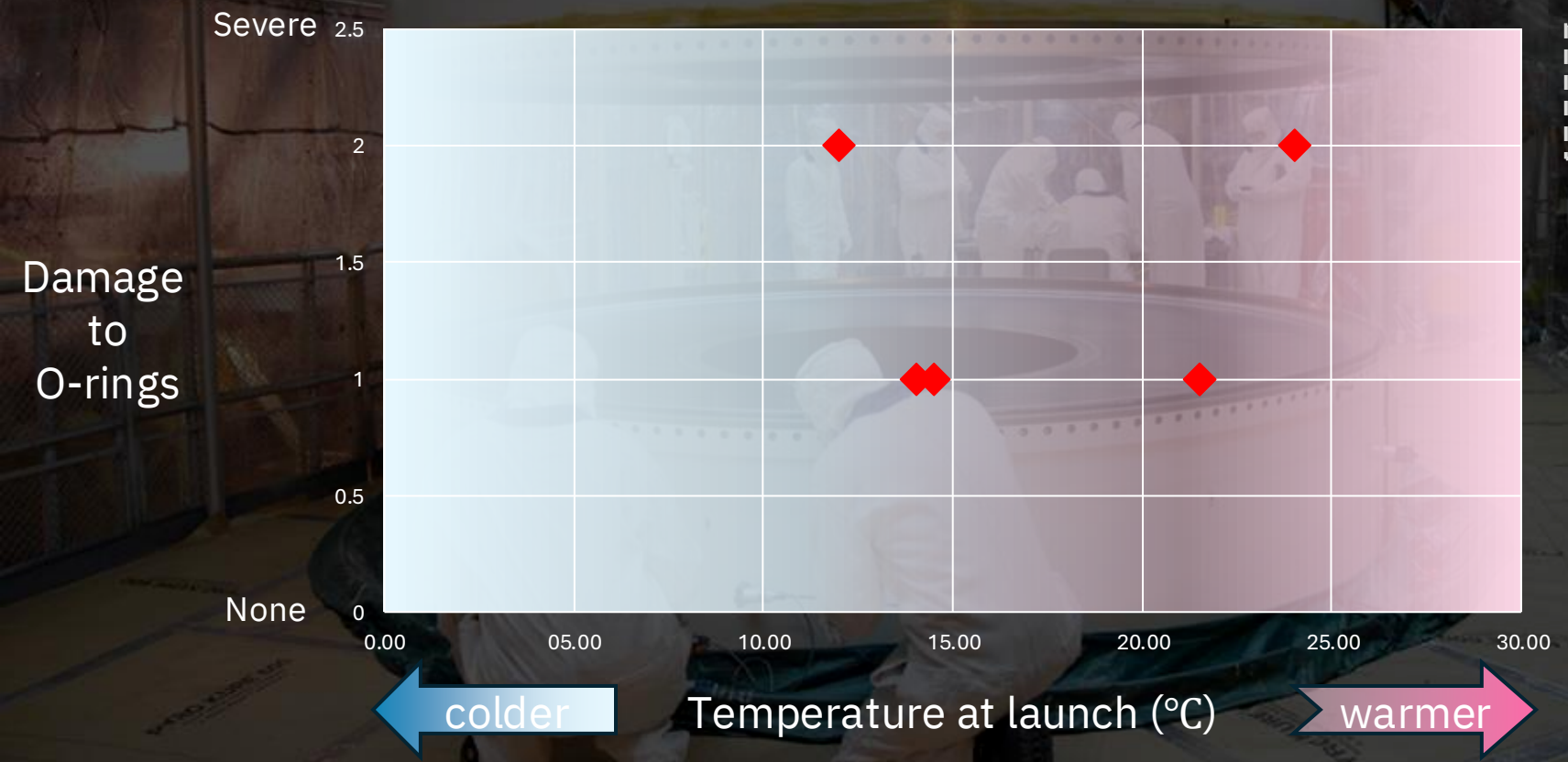


O-ring installation

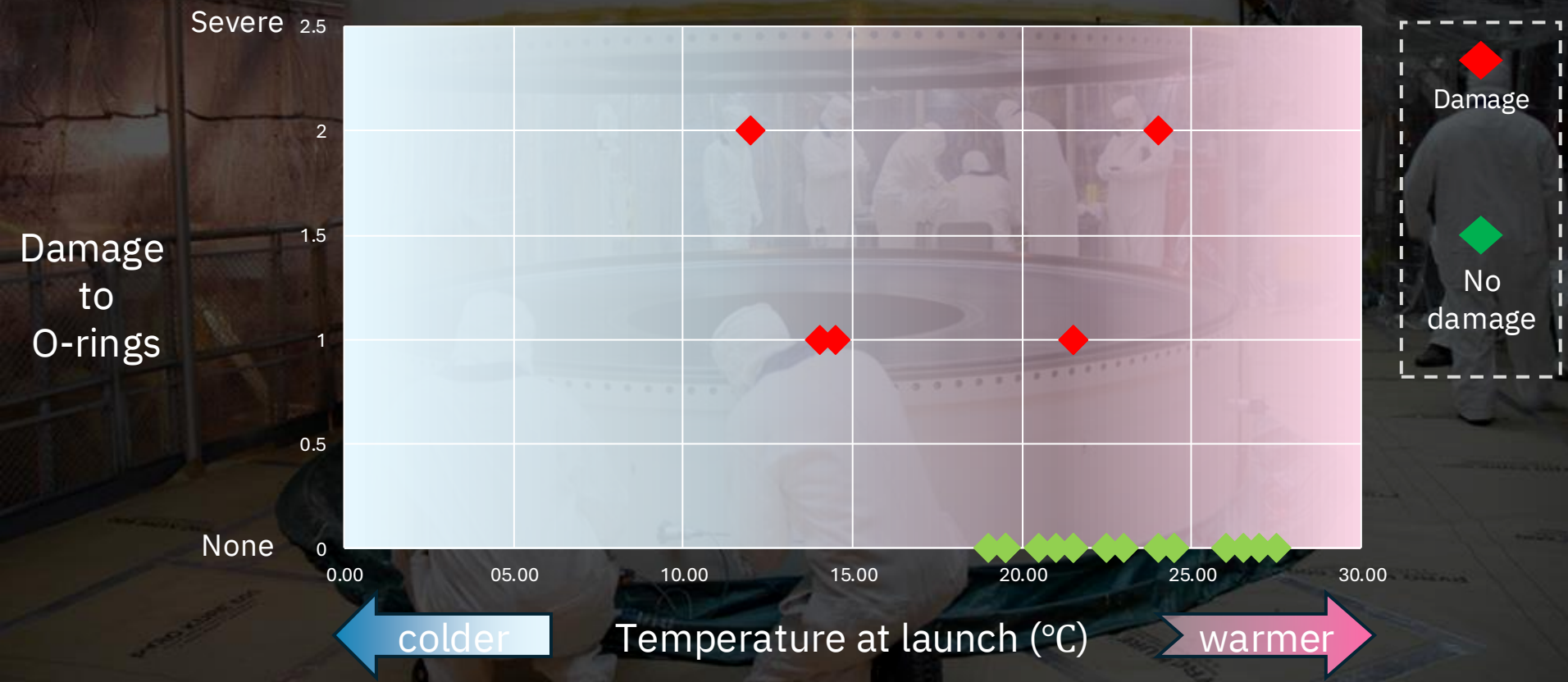




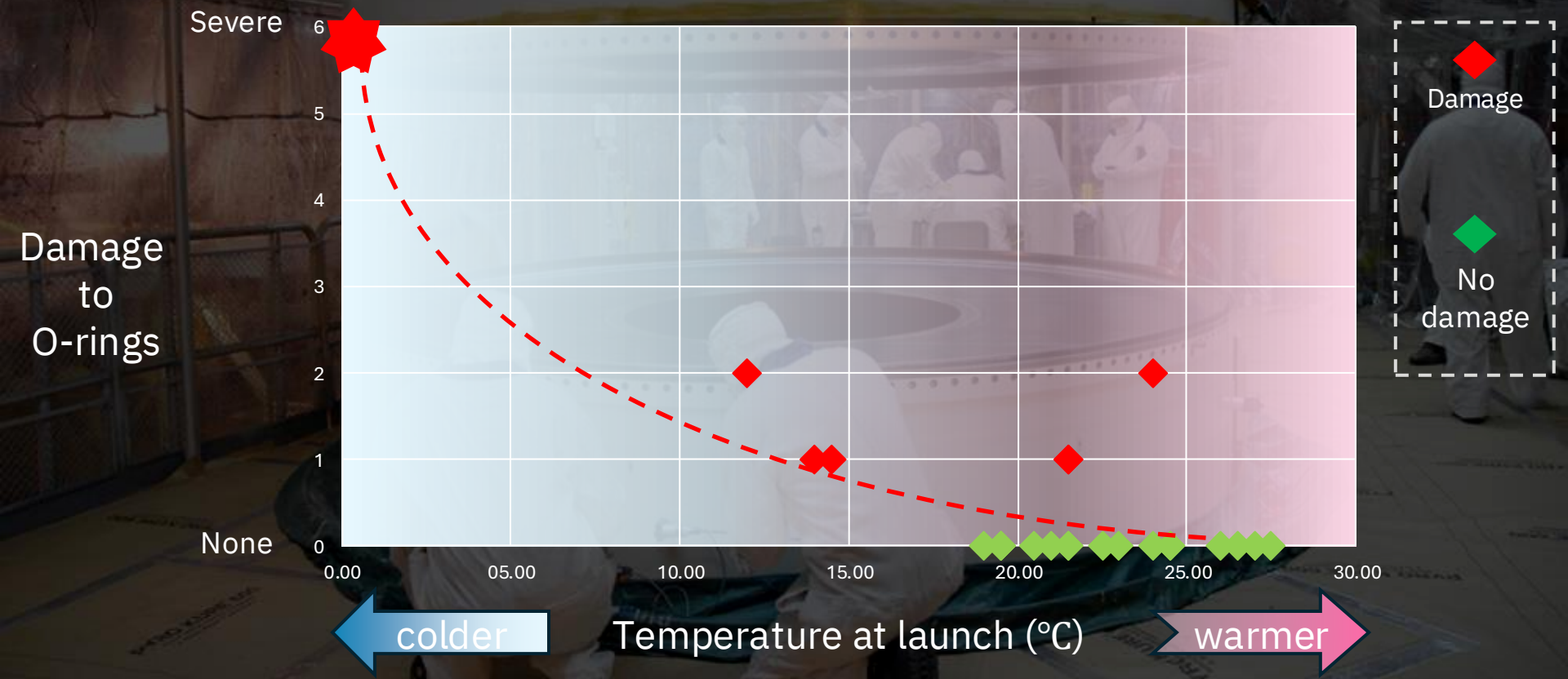
O-ring damage on flights vs temperature



O-ring condition on all flights vs temperature



Challenger launches on the coldest day



Normalization of Deviance

- Small departures from acceptable risk gradually become accepted because nothing bad happened.
- Near misses get reclassified from warnings to business as usual.

-- Sociologist Diane Vaughan,

The Challenger Launch Decision

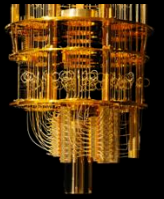
Normalization of Deviance

Challenger	AI Software Teams
O-ring erosion	AI-generated code nobody fully understands
Launch pressure	Ship faster because of AI
Previous successful launches	“AI got it right last week”
Engineers raise concerns	“I don't trust this implementation”
Managers ask for proof	“But the tests are green”

The promise of computing progress







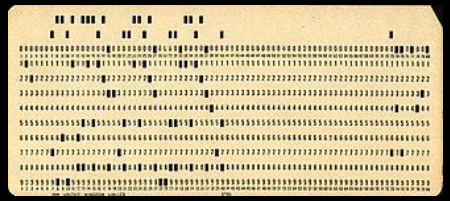


20’s 30’s 40’s 50’s 60’s 70’s 80’s 90’s 00’s 10’s 20’s









brobert@il.ibm.com

flyingbarron

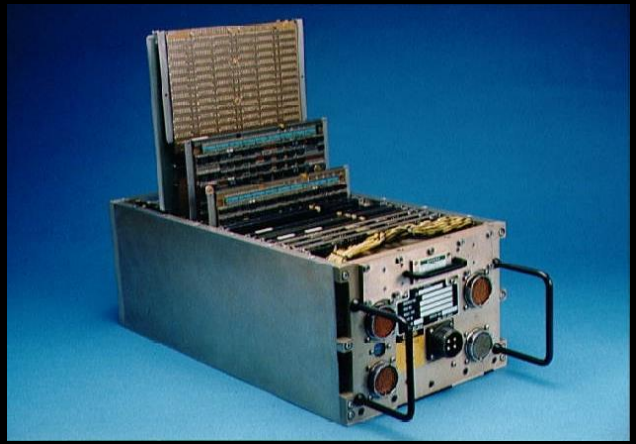
SRE CON EUROPE MIDDLE EAST AFRICA

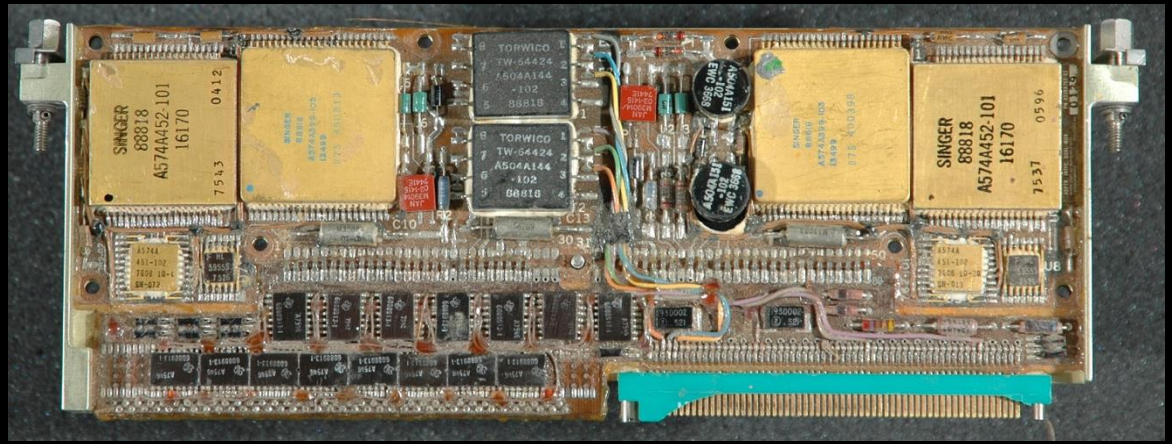
IBM

IBM’s Shuttle Computers

- 5 x AP-101 computers working in parallel with redundancy
- 420,000 lines of code. Fewer than 1 defect per 10,000 lines

It wasn’t enough to prevent disaster





Reliability is part of culture, not technology

- Not created by developers, owned by SRE, or enforced by dashboards.
- **Expressed through what managers do when tradeoffs appear.**
- If concerns stop escalating, it is not because there are no concerns.
- **“What happens when someone raises a signal you don't want to hear?”**

You’re also flying a shuttle

The Shuttle in 1986	Your Org Today
Normalization of deviance O-ring erosion survival is reclassified as acceptable	**Test suite erosion** AI ships fast. Tests are auto-generated without depth. You set an agent to check the results of an agent. Each safe release normalizes the next shortcut.
Ownership ambiguity Morton-Thiokol built it. NASA approved. Nobody owned the risk.	**“The AI wrote it”** The developer accepted. The team merged. The manager set velocity target. Blame dissolves into the toolchain.
Schedule pressure overrides signal Cost of delay outweighed likelihood of risk.	**Velocity as the metric** Success is measured on speed (or maybe volume).
Optimistic reporting The 'within limits' language shaped what felt normal.	**Confident-sounding output** It looks right. It compiles. It passes shallow review.

Safety

The definition of safety should be changed from

‘avoiding that something goes wrong’
to
‘ensuring that everything goes right’

-- Professor Erik Hollnagel,
From Safety-I to Safety-II: A White Paper

The Challenger Lesson

- Your Challenger probably won't involve rockets.
- It will involve software
- How do leaders make good decisions when the technology becomes more capable than the organization?



Summary

- NASA proved that you can build ***nearly*** flawless software.
- Challenger proved that flawless software ***cannot compensate*** for flawed organizational decision-making.
- AI is the newest chapter in engineering.
- **Management is the oldest dependency.**

Summary

- NASA proved that you can build ***nearly*** flawless software.
- Challenger proved that flawless software ***cannot compensate*** for flawed organizational decision-making.
- AI is the newest chapter in engineering.
- **Management is the oldest dependency.**

Summary

- NASA proved that you can build ***nearly*** flawless software.
- Challenger proved that flawless software ***cannot compensate*** for flawed organizational decision-making.
- AI is the newest chapter in engineering.
- ~~**Management**~~ **Leadership is the oldest dependency.**

@flyingbarron
https://flyingbarron.com



brobert@il.ibm.com