

# Some pointers to when a large project is going wrong

Paul Caseley OBE  
Dstl Fellow  
FIET FBCS FSaRS MACS

Cyber Security and Safety Group  
Cyber and Information Systems Division

© Crown copyright (2024), Dstl (DSTL/PUB161535). This information is licensed under the Open Government Licence v3.0. To view this licence, visit <https://www.nationalarchives.gov.uk/doc/open-government-licence/>. Where we have identified any third party copyright information you will need to obtain permission from the copyright holders concerned. Any enquiries regarding this publication should be sent to: [centralenquiries@dstl.gov.uk](mailto:centralenquiries@dstl.gov.uk).



## Disclaimer

*This presentation is an overview and is released for informational purposes only. The contents of this presentation should not be interpreted as representing the views of the UK MOD, nor should it be assumed that they reflect any current or future UK MOD policy. The information contained in this presentation cannot supersede any statutory or contractual requirements or liabilities and is offered without prejudice or commitment.*

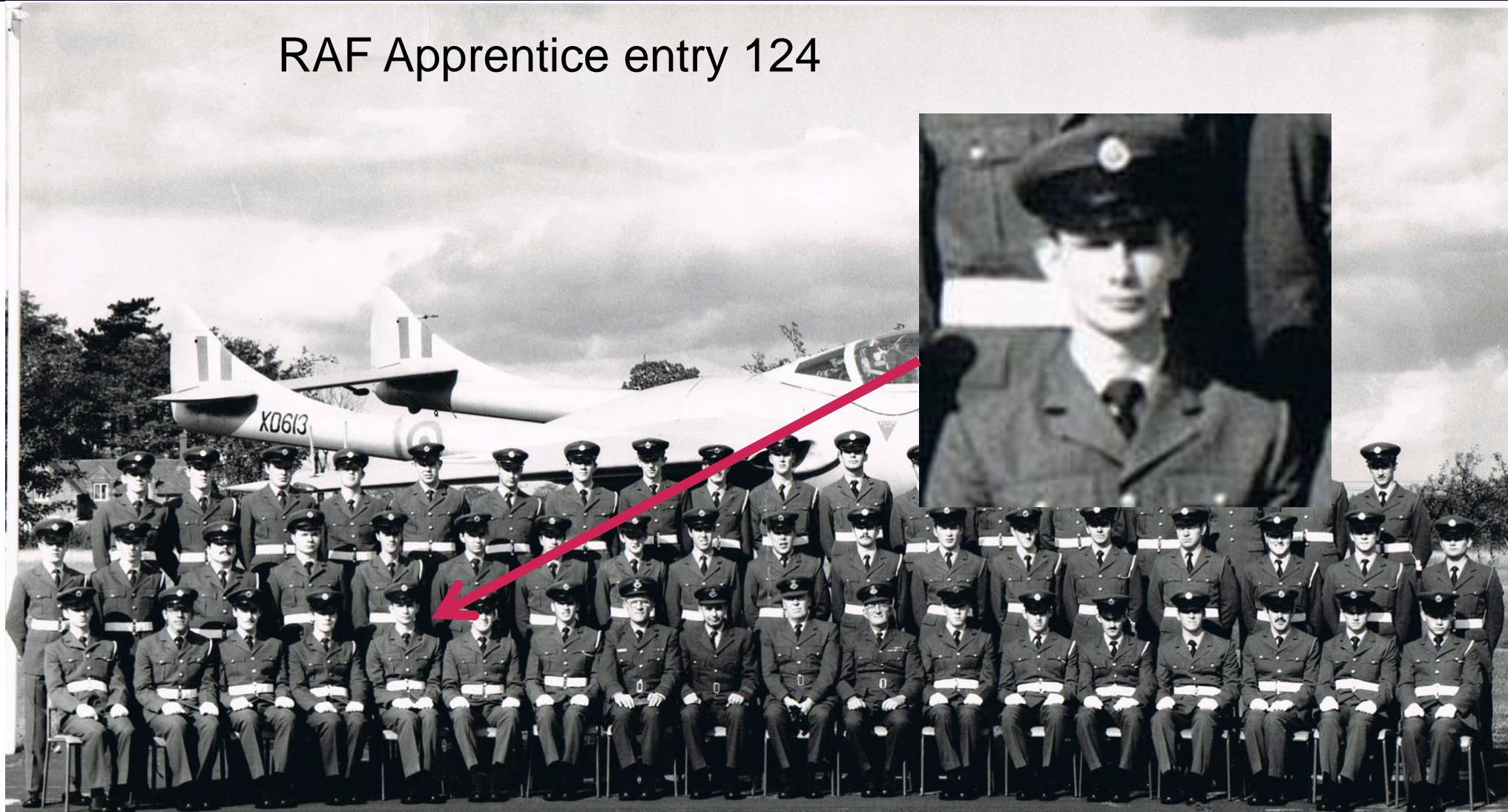


Danger driver asleep!  
Yellow jackets are  
insufficient protection  
against 80T tank





# RAF Apprentice entry 124

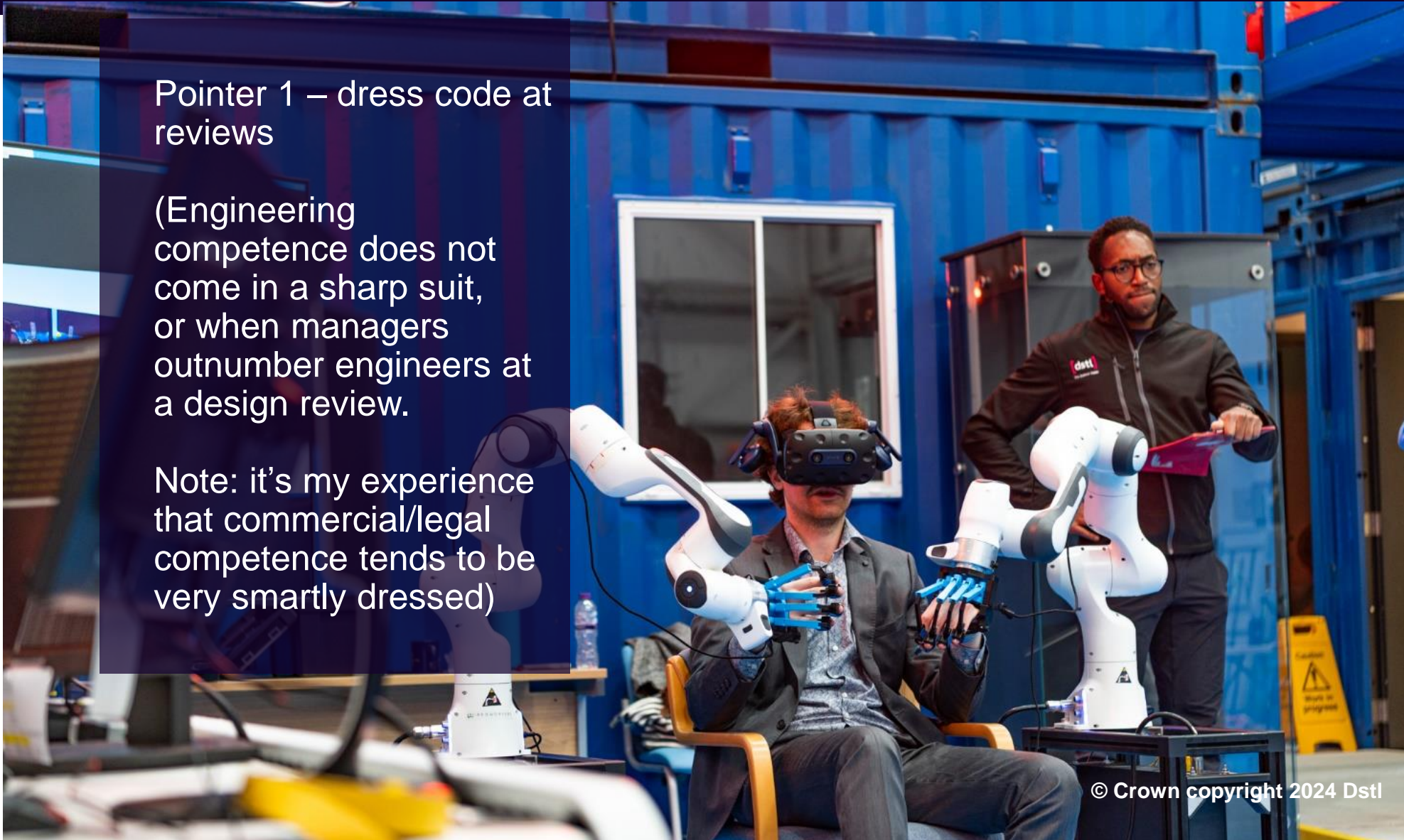




Pointer 1 – dress code at reviews

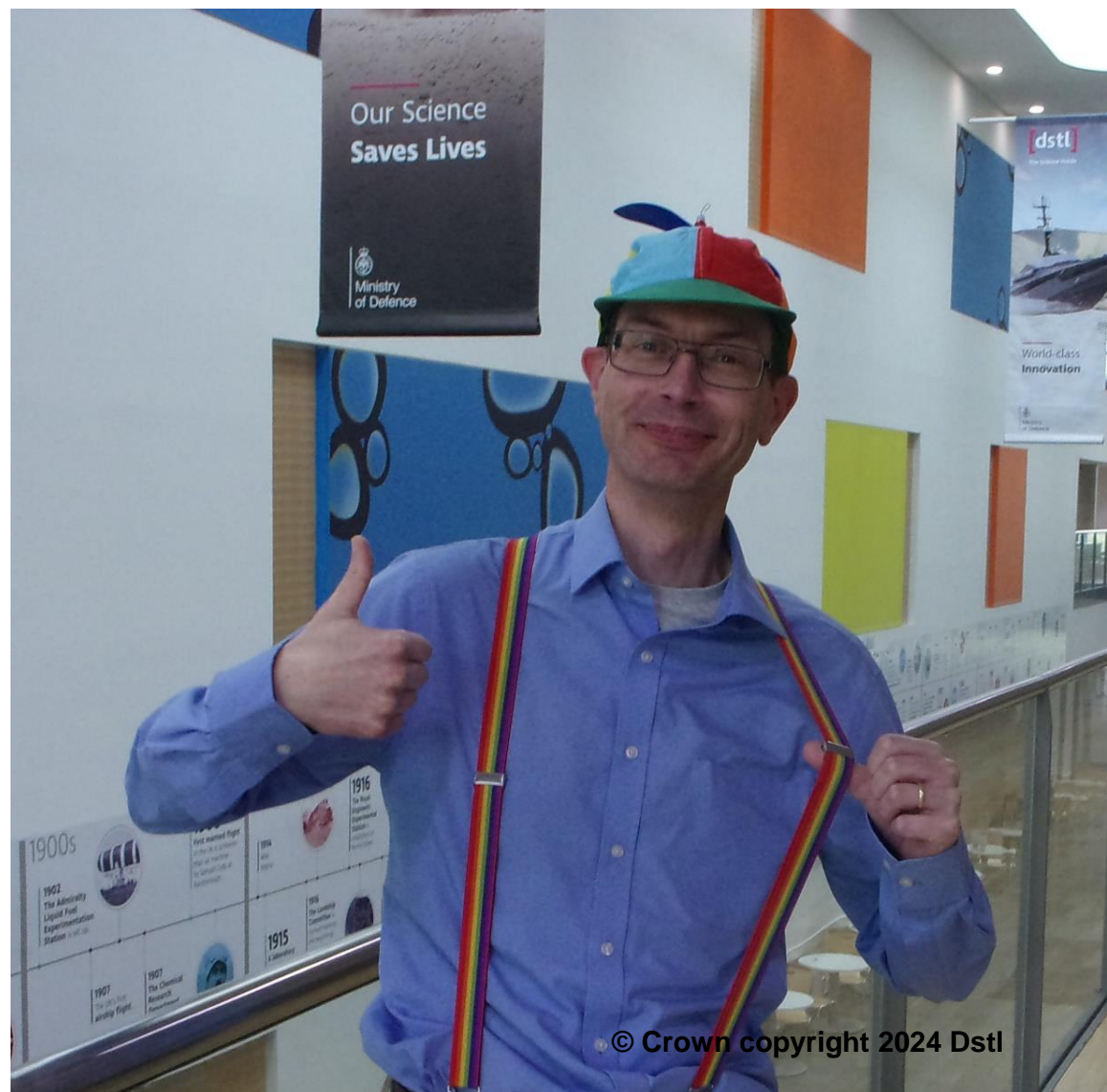
(Engineering competence does not come in a sharp suit, or when managers outnumber engineers at a design review.

Note: it's my experience that commercial/legal competence tends to be very smartly dressed)



Competence  
over style!

Braces a bit too much?





If you go through 3 versions of “systems” before first delivery of the system/software then you are probably in trouble!

■ Versions:

- 1.1
- 1.2
- 1.3
- 1.3.a
- 2.0 .....

Pointer 2 – What version?

(Re-baseline continuously so we can deliver less but claim a payment milestone.

Or, is this the third version numbering system you’ve used and you’ve still not released first field version – is your project in trouble?)

■ Versions systems:

- 1.1
- 2
- A-1
- A-2
- Sys1-0 ?????

# A useful paperweight? Software lags behind the Hardware

Yes that is a 8 inch floppy disc

What's a floppy disc?

World Class Physicist. Now with grey hair!

Pointer 3 – paperweight  
(Buy all the hardware now! - NOT!  
Or lets buy all the hardware at the beginning of the project  
we can always upgrade at the end)

For software development you don't need hardware to prototype!



When something is validated it is usually associated with a significant delivery payment milestone. Canny managers can manipulate a definition to say, as far as they are concerned, its validated!

Pointer 4 – The reinvention of English

(or subtly, or not so subtly, redefining internationally accepted key definitions to deliver less)

Contract speak, engineering should be simple and clear



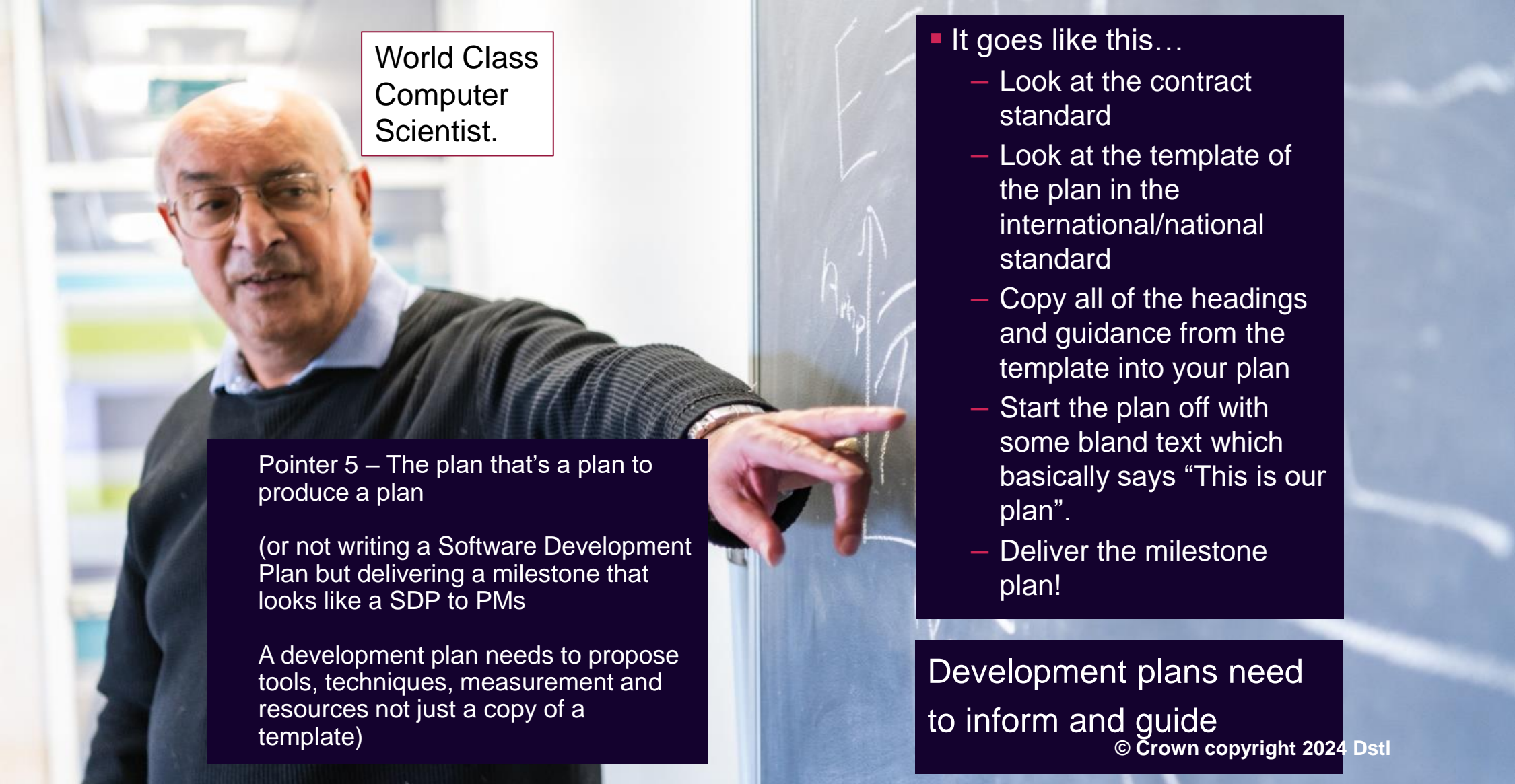


**Validation** – The process of evaluating a xxxxxxxx or, where appropriate, by zzzz yyyyyyyy xxxxxxxxxxxxxxxx using approved procedures, directives, or checklists. xxxxxxxx models xxxxxxxx determining xxx x xxxxxx xxxxxx xxx an accurate abstraction xxxxxxx x xxxxxxx xxxxxx xxx from the context of xxx xxxxxx xxxxxx xxx xxxxxx xxx xxx xxx x xxx x xxxxxxx xxx x xx simulation. As an example, a XXX model that models the YYY is validated xxx xxxxxx xxxxxx xxxxxx xxxxxx xxx xxx xxx function.

Keep Definitions simple and standard

Or more simply: Validation: The process of determining the degree to which a model or a simulation is an accurate representation of the real world from the perspective of the intended uses of the model or the simulation (NASA-STD-7009)

## Delivering a quick payment milestone... or is it?

A man with glasses and a dark sweater is pointing at a whiteboard. The whiteboard has a diagram with arrows and some text. The man is looking towards the camera.

World Class  
Computer  
Scientist.

Pointer 5 – The plan that's a plan to produce a plan

(or not writing a Software Development Plan but delivering a milestone that looks like a SDP to PMs)

A development plan needs to propose tools, techniques, measurement and resources not just a copy of a template)

### ▪ It goes like this...

- Look at the contract standard
- Look at the template of the plan in the international/national standard
- Copy all of the headings and guidance from the template into your plan
- Start the plan off with some bland text which basically says “This is our plan”.
- Deliver the milestone plan!

Development plans need to inform and guide

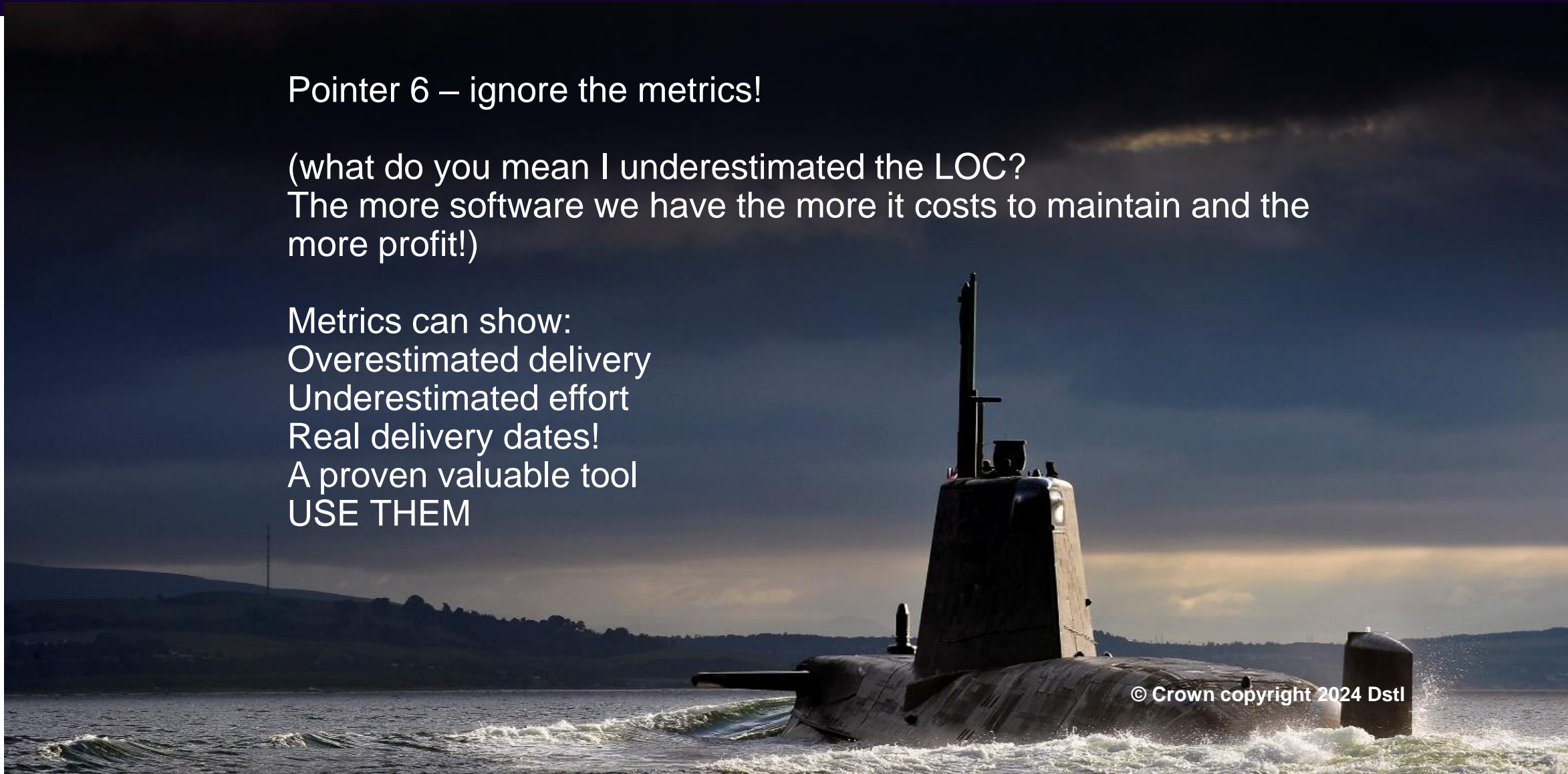


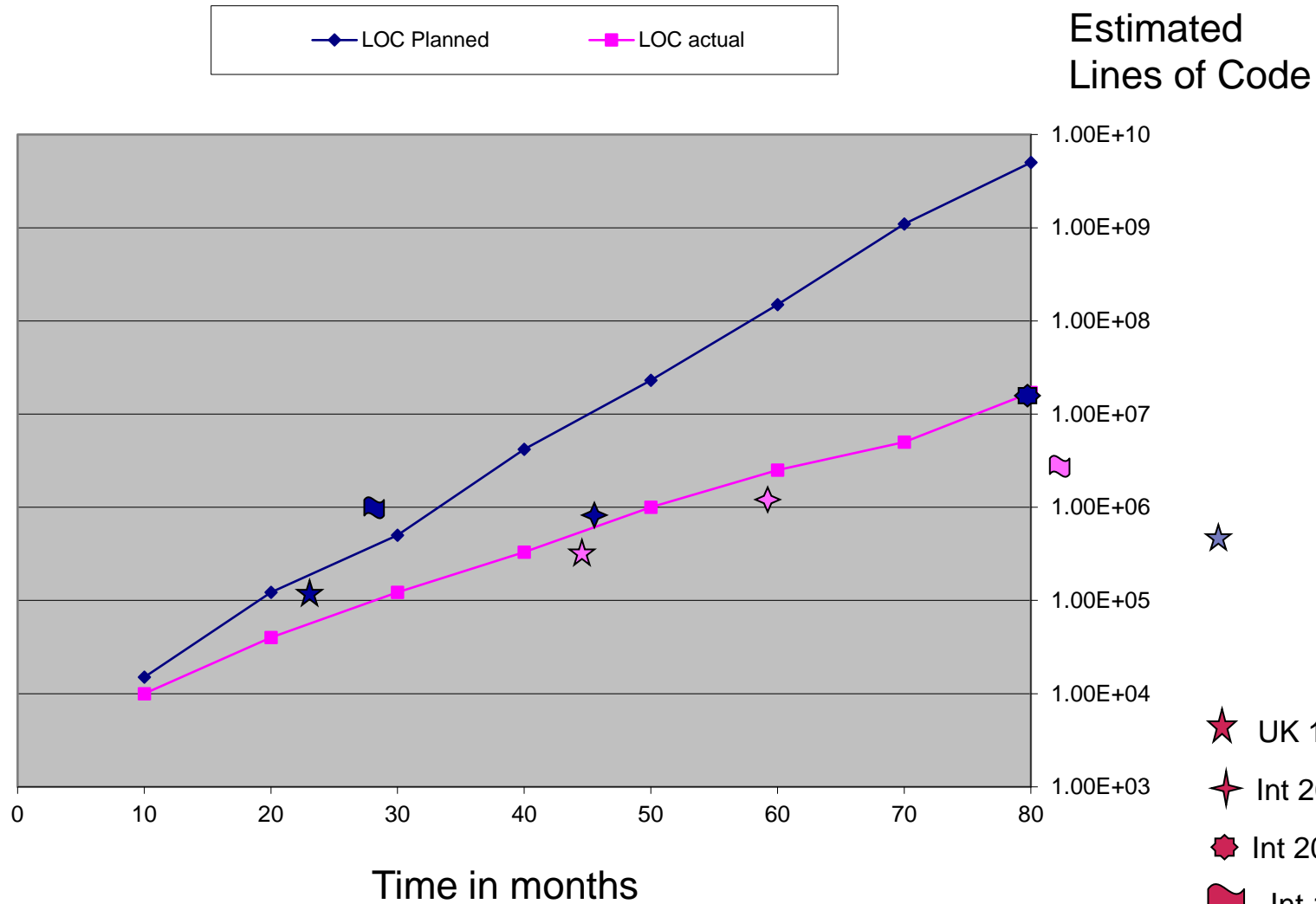
Development measurement is useful, but only if you use it!

Pointer 6 – ignore the metrics!

(what do you mean I underestimated the LOC?  
The more software we have the more it costs to maintain and the more profit!)

Metrics can show:  
Overestimated delivery  
Underestimated effort  
Real delivery dates!  
A proven valuable tool  
**USE THEM**







Pointer 7 – Using someone else's processes  
(We're using the same processes as our super-duper team  
in division XX, so our products will be as good.

SQEP is better than processes, but good processes  
enhance SQEP)

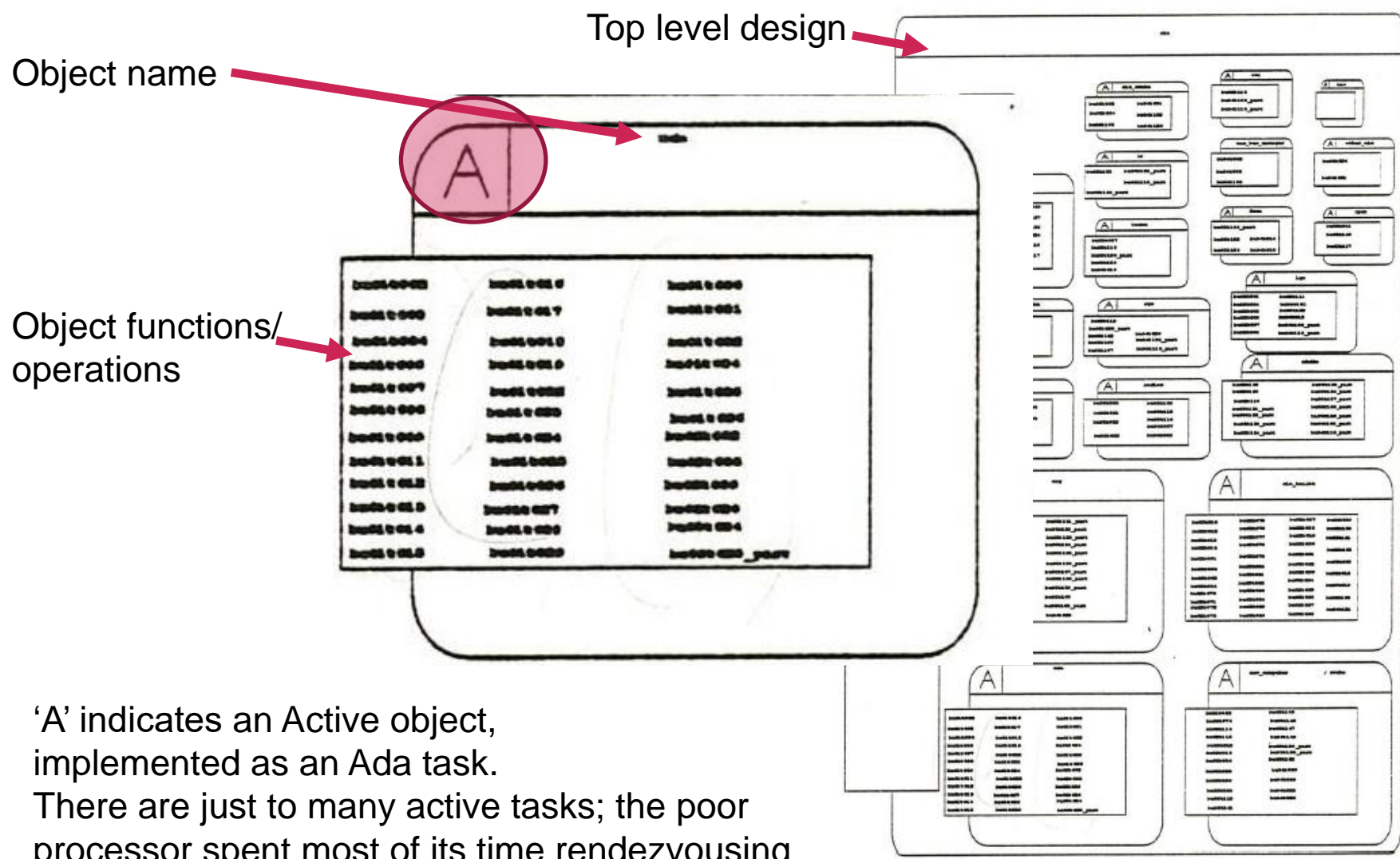
You can't convert  
the team to new  
tools and processes  
without training and  
practice



Pointer 8 – Overusing  
the new thing, its not  
fashion, its engineering

(Whether it's a new  
design method, new  
language, or new  
features in a language.  
Over use, because  
everyone else is doing  
it can be just plain  
wrong)





- Communicate clearly the science and technology
- No hard and fast rules but..
  - Competent teams
  - Control the project
  - Prototype the engineering
  - Clear definitions
  - Plans that inform and guide
  - Measure the development and use the measurements
  - Beware of the new thing, it may lead to technical debt
  - Processes and tools are only part of the solution, SQEP matters

But is  
there more?





We take safety and safety culture seriously

Pointer 9 – Cyber Security is not my problem!

Software engineers are probably the biggest problem in Cyber Security

Bad actors can exploit this problem

A Secure by Design culture is needed by everyone, particularly software engineers

Secure by Design applies to all elements of the software development supply chain

- Large projects are challenging and sometimes the warning pointers just fail to get the attention of decision makers
- 'AJAX report highlighted safety issues not acted on and subsequently:
  - “there have been 167 incidents of pain/harm/injury”
- Dstl supported and advised the project including safety
  - Dstl believes it is “*the science inside UK defence and security*”.
- Some our AJAX customers thought:
  - “With Dstl it’s really difficult to stop them being overly picky.”
  - “*they could be a bit annoying because they could be a bit dogmatic and bloody-minded*”
- The KC’s view is that, on key S&T advice:
  - the advice .. was largely clear and practical
  - prescient about potential safety problems
  - scientific advisers should be diligent and raise points of detail, in particular when they concern safety
- Co-operation is challenging and many of the recommendations related to improving MOD internal communication/co-operation

Report of the Armoured Cavalry Programme  
(Ajax) Lessons Learned Review



Clive Sheldon KC  
19 May 2023



# [dstl] The Science Inside

Discover more

