SOFTWARE SYSTEMS

Changing the way the world does software

D-RisQ SOFTWARE SYSTEMS

Where Innovation Soars BLUEBEAR

Certifying Autonomous Systems

Nick Tudor

njt@drisq.com

A D-RisQ research project supported by the Technology Strategy Board:

“Developing novel concepts in autonomous service robotics”

In collaboration with Blue Bear Systems Research

D-RISQ SOFTWARE SYSTEMS

Where Innovation Soars BLUEBEAR

The Approach

- Unmanned systems are simply systems with software
- Certification requirements change per domain
  - Context for safety case is different
  - Essentially all want to show that system and software is 'safe'/'secure' [in context]
- We will use a maritime example - COLREGS
  - Only 3 dimensions
  - Cheaper demonstration than using a UAV!
- Aim to use/meet:
  - DO178C
  - DO333 FM Supplement
  - DO330 Tool Qualification

} Aim to considerably reduce cost/time

D-RisQ SOFTWARE SYSTEMS

Where Innovation Soars BLUEBEAR

Autonomous vs Automatic

- Automatic: systems that exist already
  - Eg Flight control, undercarriage, fuel control
- Autonomous: replaces the “human decision making algorithm”
  - Eg Decides to take avoiding action or to re-plan

D-RISQ SOFTWARE SYSTEMS

Where Innovation Soars BLUEBEAR

What DAL?: Detect, Sense, Avoid

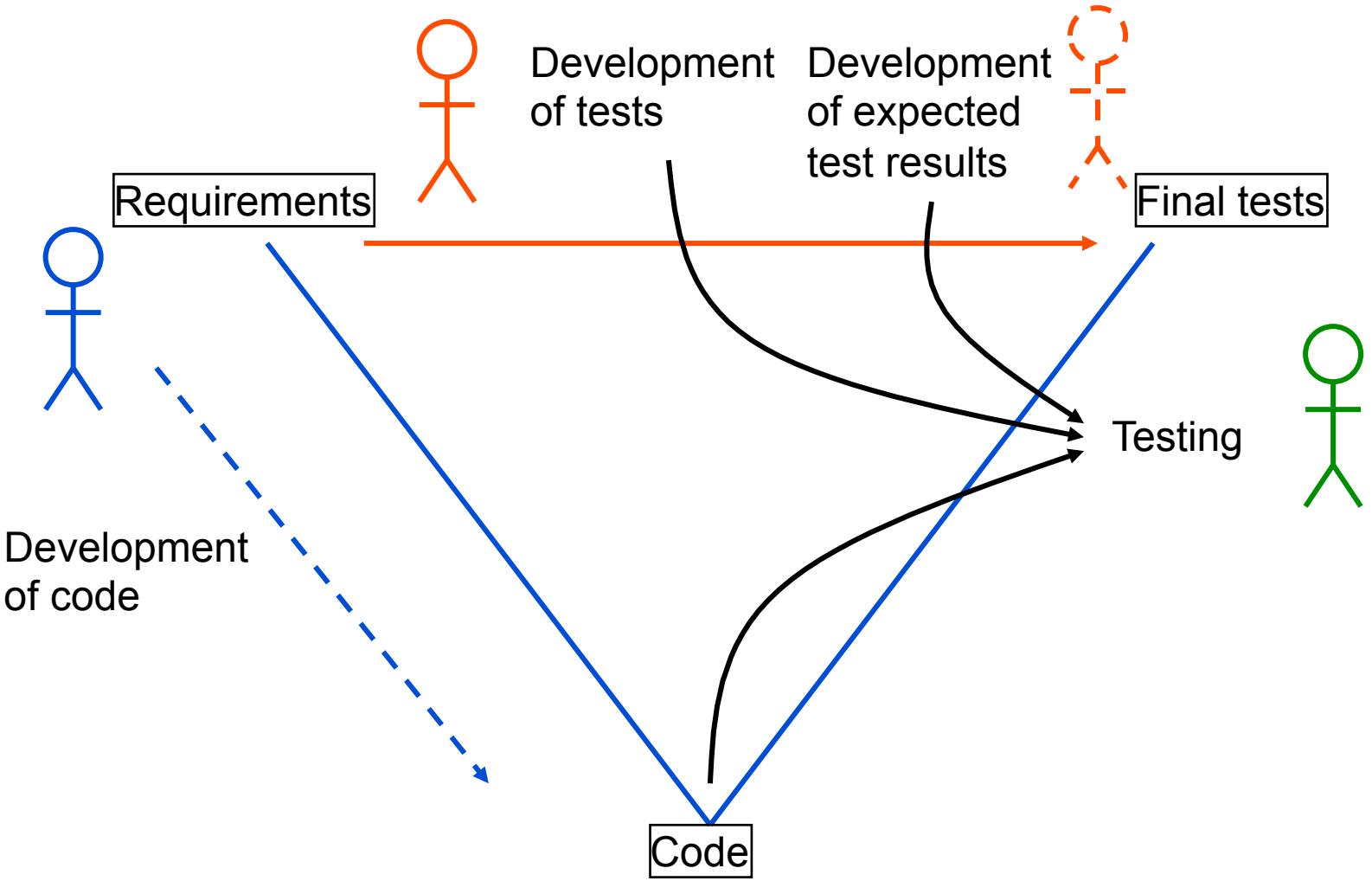
- Detect:
  - This is partly an architectural, partly integrity of each system and partly spectrum argument
  - DAL could therefore be anything from DAL A-D
- Sense:
  - Replacement of the decision making made by an operator (pilot, driver, helmsman,…); ie is 'autonomy'
  - Is part of the architectural argument
  - Is assumed to be DAL A
- Avoid
  - This is the automatic reaction to an instruction to move given by the 'sense'; ie is the control system
  - Is therefore assumed to be DAL A

D-RISQ
SOFTWARE SYSTEMS

Where Innovation Soars
BLUEBEAR

DEMONSTRATING CORRECT BEHAVIOUR

Testing and Standards – Human Error Entrapment

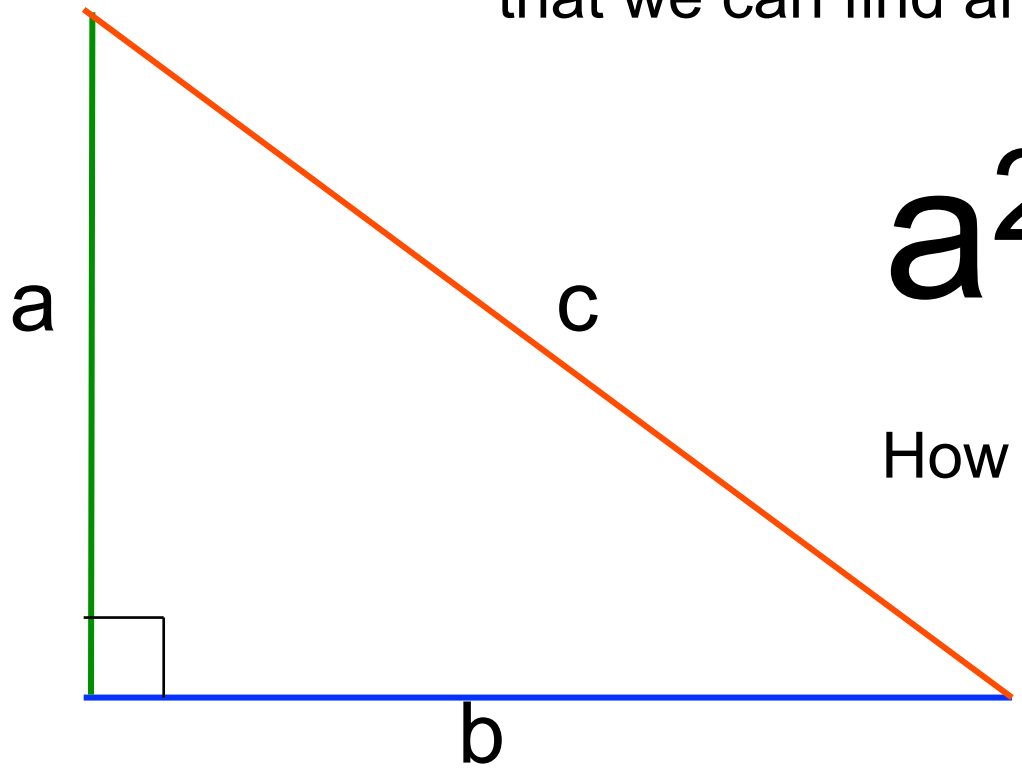


D-RISQ SOFTWARE SYSTEMS

Where Innovation Soars BLUEBEAR

“For All” vs Testing - 1

Can we show that for any a and any b, that we can find any c?



$$a^2+b^2=c^2$$

How much testing is required?

D-RISQ SOFTWARE SYSTEMS

Where Innovation Soars BLUEBEAR

“For All” vs Testing - 2

I believe that that no three positive integers a, b, and c can satisfy the equation $a^n + b^n = c^n$ for any integer value of n greater than two.
For example:

$$a^3+b^3=c^3$$

How much testing is required?

This is known as “Fermat’s Last Theorem” which he postulated in the year 1637 and it took until 1995 for an English mathematician called Andrew Wiles to prove this to be true

This is showing absence of behaviour and is a major difference between a test based approach and a proof based approach

D-RISQ
SOFTWARE SYSTEMS

Where Innovation Soars
BLUEBEAR

CERTIFICATION REQUIREMENTS (AKA ‘THE CHALLENGES’)

D-RisQ SOFTWARE SYSTEMS

Where Innovation Soars BLUEBEAR

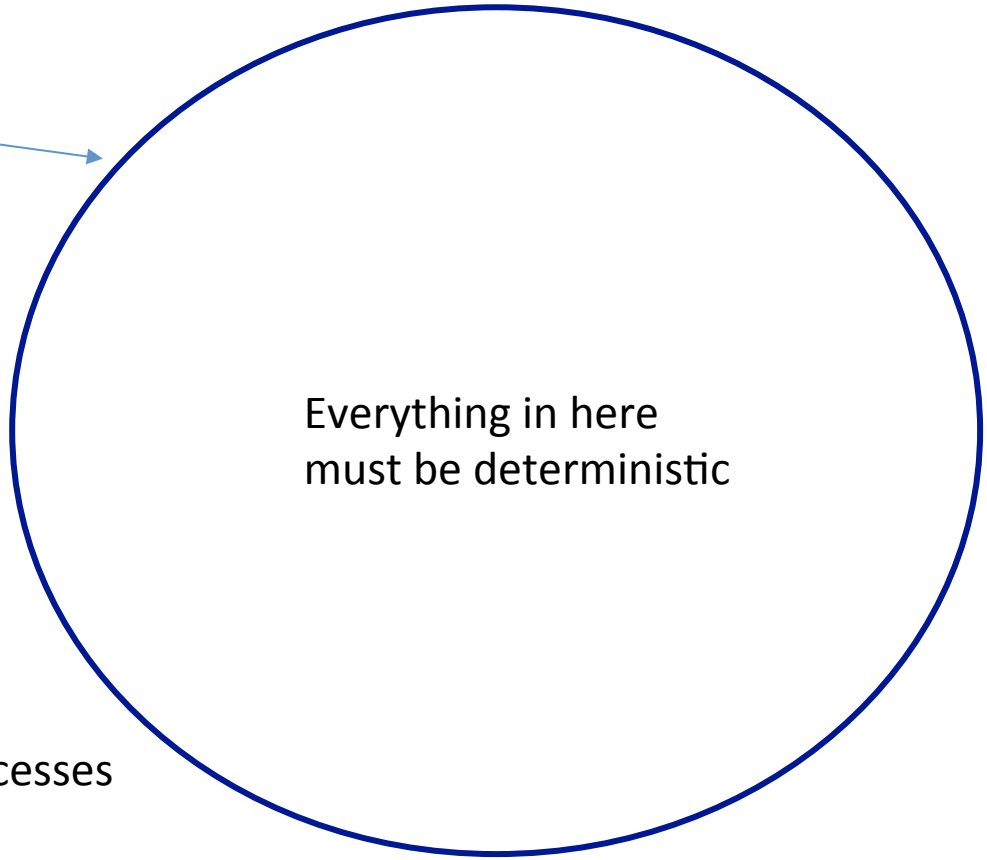
The Environment vs the 'System'

- Any system can be 'deterministic' – including autonomous systems
  - Probabilistic arguments are not
  - Knowledge of the environment is not necessarily deterministic
- Not possible to envision ALL circumstances that the system will have to encounter
  - Eg asynchrony
- Have a set of requirements/behaviours
  - Must do — Routine to achieve using 'conventional' techniques
  - Must NEVER do — Not achievable using 'conventional' techniques
  - Failure behaviour — Very expensive using 'conventional' techniques

Deterministic Behaviour

Behaviour Boundary

Defined by:
User requirements
Regulations
Physics
Environment



- Determinism demonstrated by:
  - Probing the boundary and
  - By design/development processes
- Aiming to ensure:
  - Nothing can penetrate or leak from the boundary
  - And everything inside is done correctly

System Safety Boundary

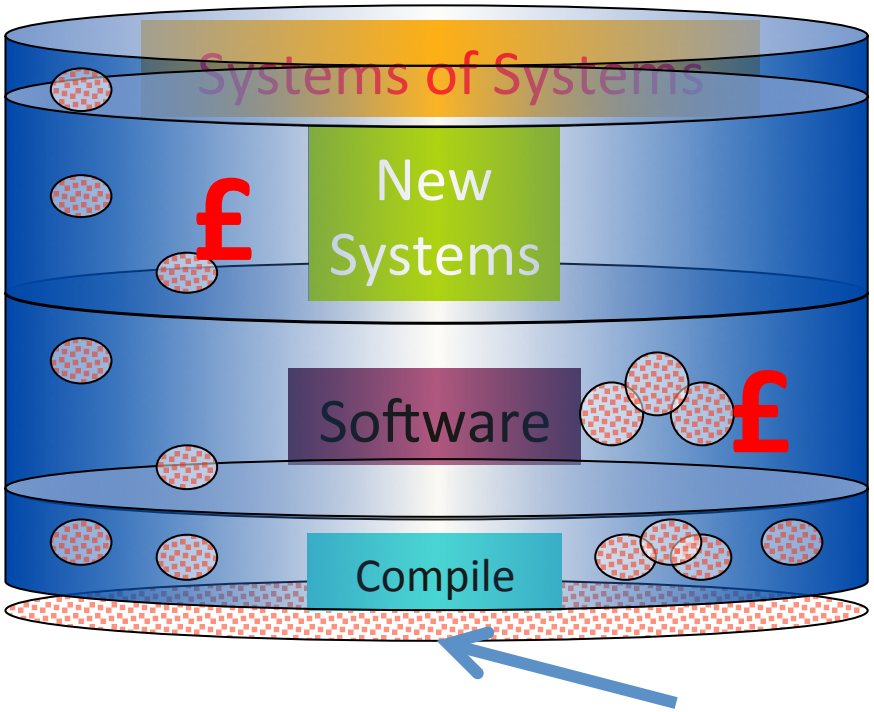
Air Domain Docs

Platform Level £ — JAR CS 23/25

System Level £ — ARP4754/4761

Software Level £ — DO178C

Executable Object Code £ — DO178C



Hardware defined and developed in parallel with software and re-introduced post compilation

Hardware with software is verified to meet (partially) software standard and (partially) system standard – mitigation of hazards

D-RISQ SOFTWARE SYSTEMS

Where Innovation Soars BLUEBEAR

The Key Issues DO333

- Have to use some parts of DO178C
  - Over-ridden by DO333 as required
  - And some reversion to eg test where it is the best tool for the job
- Tool Qualification done iaw DO330
  - Verification tools
  - Apply to systems domain as well as software

D-RisQ SOFTWARE SYSTEMS

Where Innovation Soars BLUEBEAR

The Main Formal Challenges - 1

- Within DO333 the formal method should be correctly defined and justified
  - These are new Objectives:
- All notations used for *formal analysis* should be verified to have precise, unambiguous, mathematically defined syntax and semantics, that is, they are *formal notations*
- The soundness of each *formal analysis* method should be justified. A sound method never asserts that a *property* is true when it may not be true
- All assumptions related to each *formal analysis* should be described and justified, for example, those assumptions associated with the target computer or about the data range limits

D-RISQ SOFTWARE SYSTEMS

Where Innovation Soars BLUEBEAR

The Main Formal Challenges - 2

- All assumptions related to each *formal analysis* should be described and justified, for example, those assumptions associated with the target computer or about the data range limits
- If a requirement has been translated to a formal notation as the basis for using a formal analysis, then review or analysis should be used to demonstrate that the formal statement is a conservative representation of the informal requirement

D-RISQ SOFTWARE SYSTEMS

Where Innovation Soars BLUEBEAR

Divide and Conquer

- Start with Systems Architecture using Modelworks
  - Tackling the 'Sense' aspects first
- Have a set of requirements/behaviours
  - Must do
  - Must NEVER do
  - Failure behaviour
- Derive a DAL for the system that is appropriate for the hazards
  - And for each component of the system
- Then tackle the control system software using CLawZ

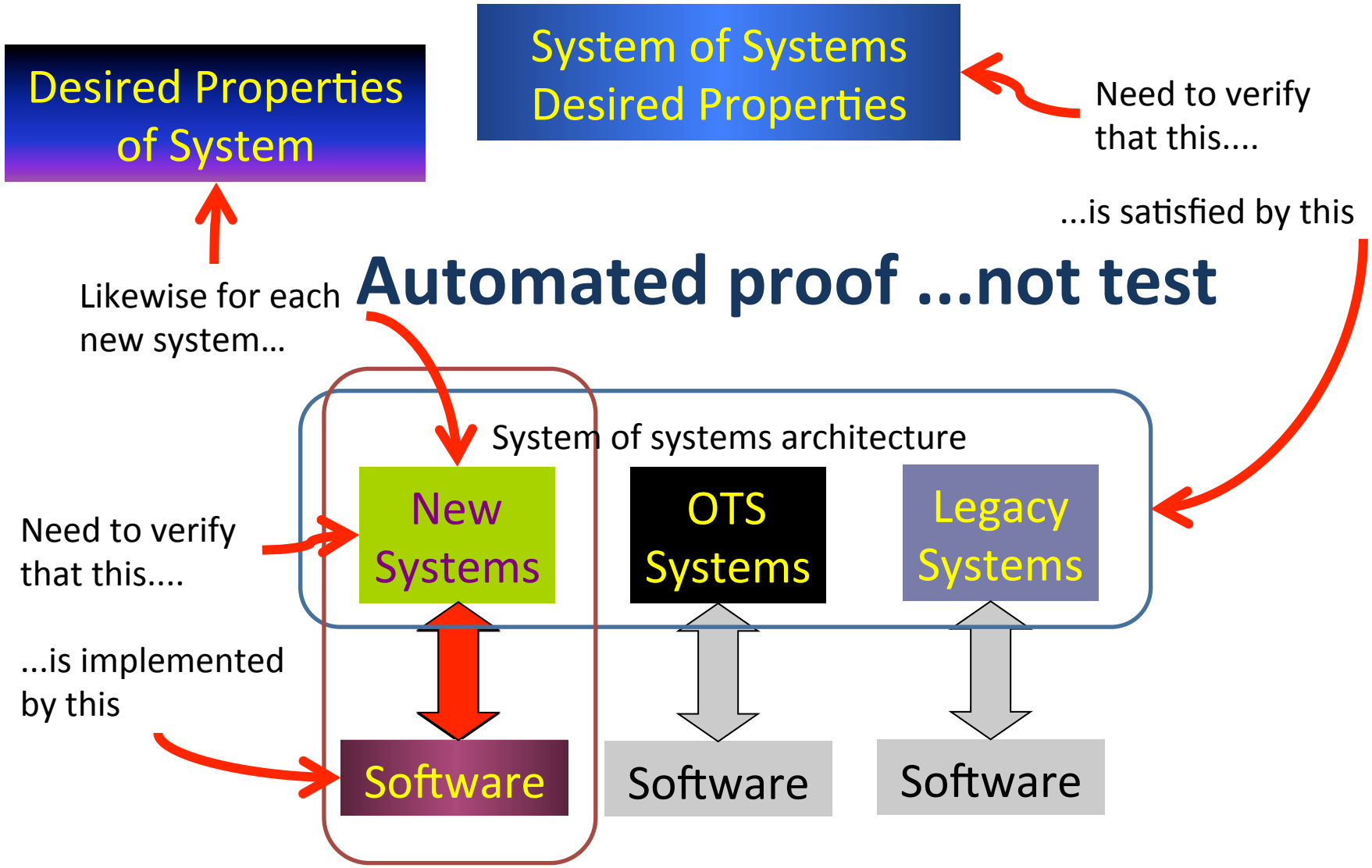
D-RISQ SOFTWARE SYSTEMS

Where Innovation Soars BLUEBEAR

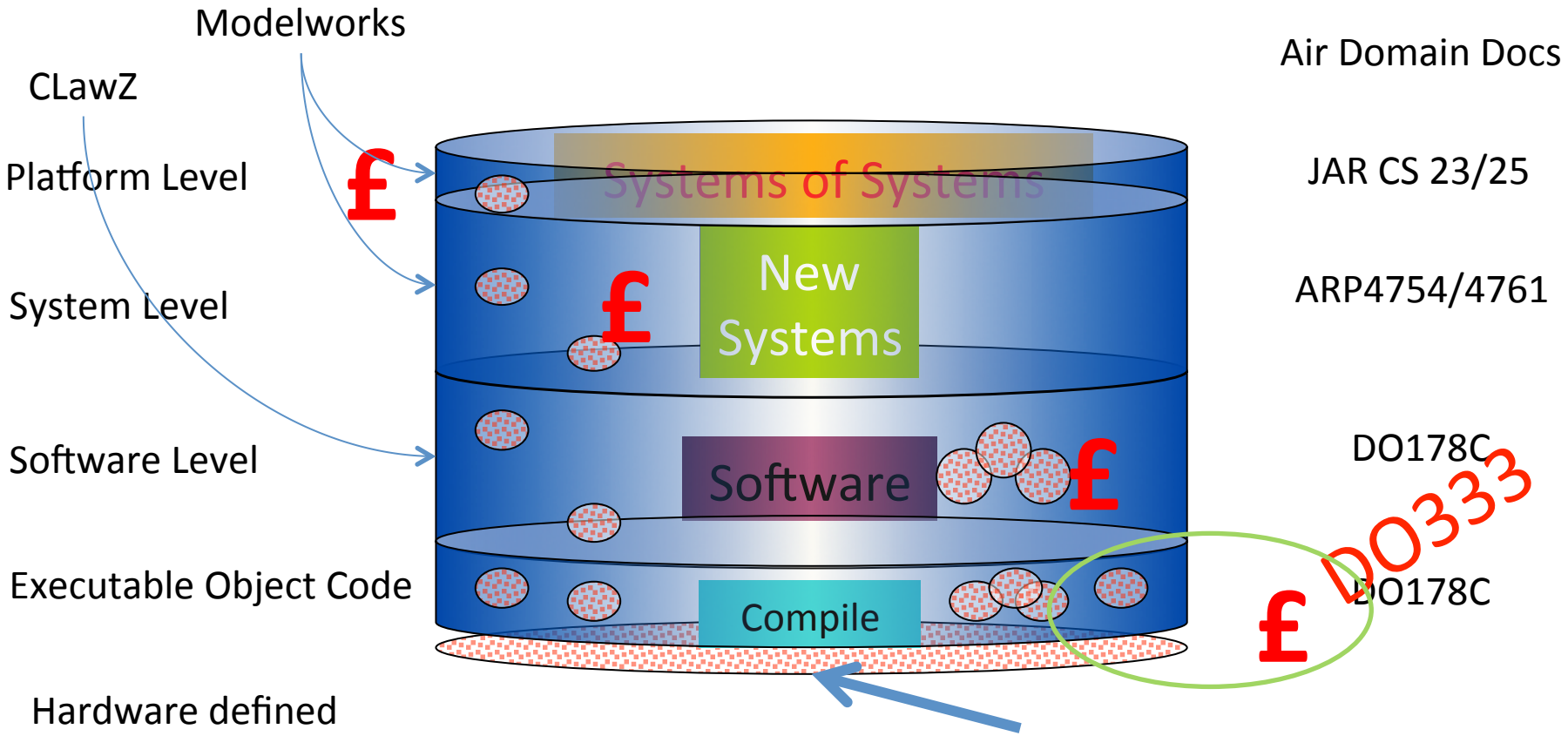
Dealing with Autonomy

- Using ‘Entry Level’ autonomy
  - Enables principles to be established before going to further levels of autonomous behaviour
- Approach is to show that requirements for behaviour of the autonomous system have been designed
  - Some ‘Must do’ and ‘Must NEVER do’ will have been passed down from System Safety processes

The Systems and Software 'Chain'



System Safety Boundary



Hardware defined and developed in parallel with software and re-introduced post compilation

Hardware with software is verified to meet (partially) software standard and (partially) system standard – mitigation of hazards

D-RisQ SOFTWARE SYSTEMS

Where Innovation Soars BLUEBEAR

Testing and Formal Methods

- Testing seeks to find errors in the development
  - The corollary is that development seeks to find errors in the testing….?
- Testing is either validation (doing the right thing) or verification (doing the thing right)
  - Often the use of tests can be about both at the same time
- Formal Methods can examine more behaviours than testing
- What happens if we can use formal methods to bolster the results of test?
  - In other words, use test to back up the analysis
  - If so, can we significantly reduce the test effort, effectively becoming validation of the analysis?

SOFTWARE SYSTEMS

Changing the way the world does software