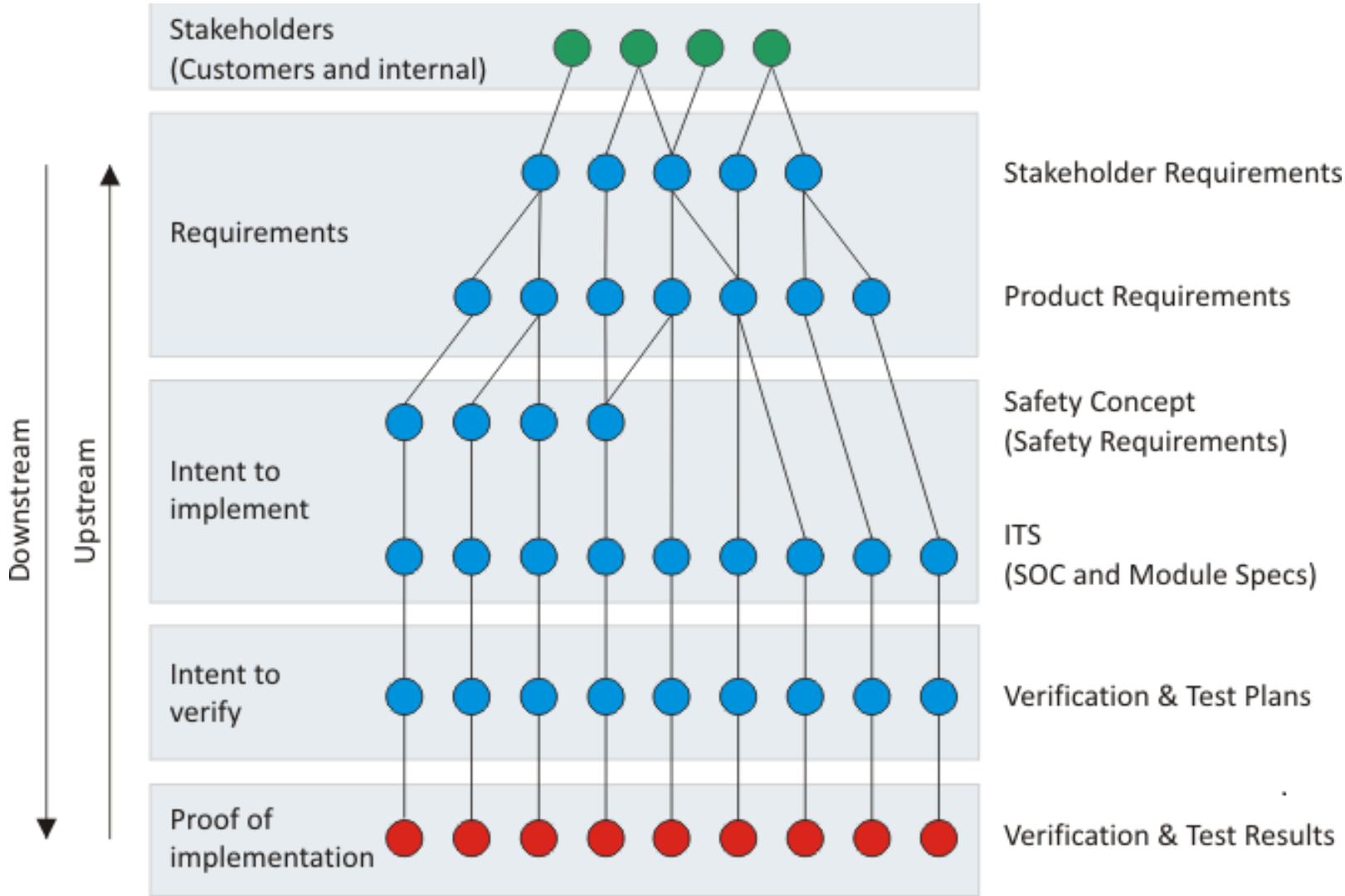
Requirements Engineering – the future in verification

Darren Galpin - System IP verification manager

Serrie-justine Chapman - Requirements Traceability Manager

What is requirements Engineering?



Copyright © Infineon Technologies 2011. All rights reserved.

The Drive for Requirements Engineering and where it‘s driving us …

Tipping Point

- New Safety Standards such as the ISO26262 (SIL)

- Quality and completeness
- Identification of requirements & dependencies
- Need to add linkage to legacy documents
- Are the requirements feasible?
- Over & under engineering
- Identify issues in flow

Copyright © Infineon Technologies 2011. All rights reserved.

UML

- Current standards drive for formal methods such as UML to document top level Safety Requirements
- Possible that this will pass into testbench documentation in the future
- Better to have more live than dead documents, as it is easy and quick to update and maintain for documentation purposes rather than leave to the end.
- Could allow round robin to assist with maintenance and handovers – change UML, see what needs to be changed in testbench, or load in testbench and visualise.
- Would also help identify dead verification code, from code which is not executed to code which is not covering a requirement.
- Test plans could become UML use cases.

Copyright © Infineon Technologies 2011. All rights reserved.

UML – The Problems and Solutions

- UML is designed for object oriented languages and software systems, so does not map cleanly to aspect oriented languages or hardware.
- Theme/UML is an extension to UML pioneered by Trinity College Dublin to handle aspects, specifically AspectJ.
- This fits better, but is still not enough for complex verification environments written in e.
- Constraints can be handled by OCL – Object Constraint Language, a UML extension.
- Extend UML by using Modelling and Analysis of Real Time and Embedded systems (MARTE) to handle time concepts.
- Extend MARTE and OCL to use Theme/UML cross-cutting aspects to allow extensions and over-rides.
- Prototype developed supporting this by Trinity College Dublin, no commercial support yet.

Copyright © Infineon Technologies 2011. All rights reserved.

How to ‘prove’ the requirement is implemented

Options :

- Link to HDL
  - + Proves that some code is in place
  - - Does not prove that it necessarily covers the function in question
  - + Allows quick support of any change requests
  - – Would need somehow still to ensure that the HDL is being checked and is working – 100% coverage but is there a checker?

- Link to tests
  - + Ensures we have a test/PSL/Coverage point intent to implement a testplan
  - – does not prove implementation, would need further linkage

- Link to pass fail criteria
  - + Proves a check/test/psl/proof has been written to test the function
  - + Prove that that has been exercised and passing

Copyright © Infineon Technologies 2011. All rights reserved.

Problem?

- Link testplan to the proof of implementation
- For random and PSL
  - Is a check there & where
  - Did we test the check and is it passing
- Currently we only look to 100% of coverage. What if the code is missing?
- Audit trail
- Link crossways – pre-silicon, post silicon. Common format independent of methodology

Copyright © Infineon Technologies 2011. All rights reserved.

Summary

- Improvements are necessary to fulfil standards
- UML can improve documentation in all areas
- Closer documentation to verification ensures likelihood of keeping it current
- Improvement of interaction between methodologies

Copyright © Infineon Technologies 2011. All rights reserved.

ENERGY EFFICIENCY
MOBILITY
SECURITY

Innovative semiconductor solutions for energy efficiency, mobility and security.