Computer System Safety in Relation to Maritime Systems

Dr Mark Nicholson

Department of Computer Science,
University of York

Goal

- Describe challenges associated with assuring safety of marine computer based systems
- Background
  - ECDIS has significant potential for safety gains
  - Weaknesses in current approach reduces some of these benefits
    - Chasing "case by case" error approach is inefficient
- Elements
  1. Technology change: Complex computer based systems
  2. Organisational change: "open network"
  3. System safety for complex computer based systems
  4. Lifecycle for system safety

Complex Technical Systems

- Elements of ECDIS tech change
  - Digitised data generated
  - Sent out, loaded & Integrated into positioning, autopilot, e-nav
    - Automated cartography by provider software, user settings & data encoding
  - Maintenance
    - Data: errors and failures sent back and updates generated
    - Software: not robust
- Complex safety critical software in platforms
  - ARP-4754a, ISO-26262
  - DO-178B/C, UK MoD Statement of Best Practice 2009
  - OPENCOSS safety and compliance cases

Open Networks

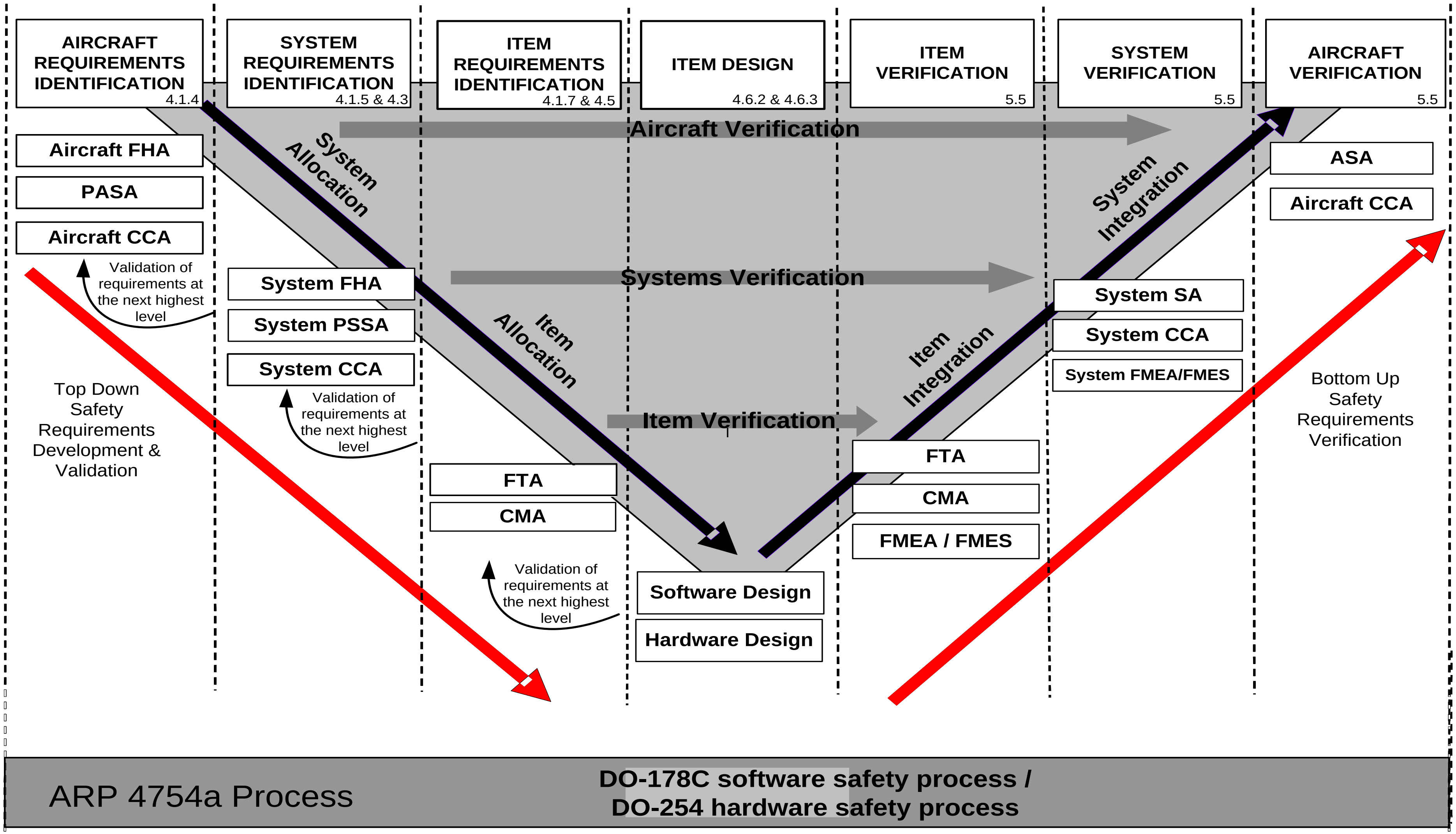
- Historically closed network (known people & provenance)
  - Hydrographic surveyors etc send in data
  - Compiled into chart by cartographers
  - Errors / new data received from operators
  - Cartographers update charts and send out
- Open network (lots more organisations involved)
  - Multiple software configurations and display providers
  - Multiple operators (and tweaking of what they see)
  - Multiple pathways for feedback
    - Not clear who to send it to and in what form
- **Interface and configuration control issues**

System Safety Engineering

Managing *unintentional* harm *caused* by complex / integrated (often computer based) systems

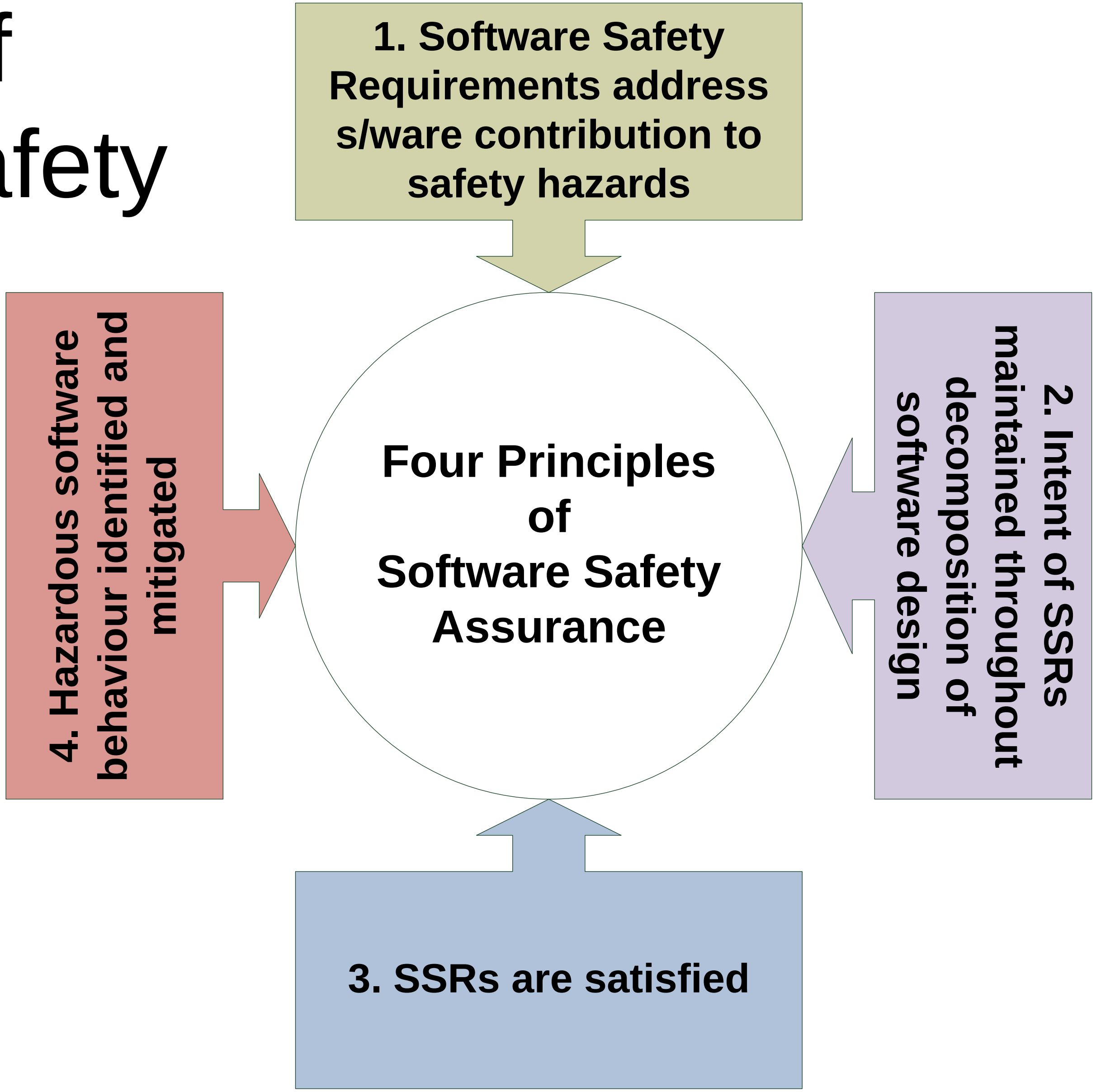
1. Understand system of interest
   - Including environmental / human / organisational context
2. Identify and evaluate safety risks associated with system
   - Usage pathways, applied experience, predictive analysis
3. Develop means of controlling risks
   - Evaluating cost / benefit trade-offs
   - Driving design and operational activities
4. Verify effectiveness of controls
   - Through analysis, testing, in-service feedback, etc.
5. Provide evidence of acceptable safety
   - For certification / customer / public acceptance
6. Maintain safety throughout system life

Example Process



Principles of Software Safety

- Computer safety addresses
  - Random failures from computing hardware
  - Systematic logic issues from software



Conclusion

- Why system safety is hard?
  - Scale & Complexity
  - Difficulty of validating & verifying safety features of functionality
- Software is a focus because it is often
  - Main determinant of function
  - Most complex part of design
  - Has significant authority over actions of vessel
    - operators do not have “headsworth” to overcome errors
- Issues
  - Advisory only system: mission creep
  - Not aerospace: true not railway, automotive, medical either but…
  - Cost is main driver: true but “ryan air SMS”, costa concordia claims etc