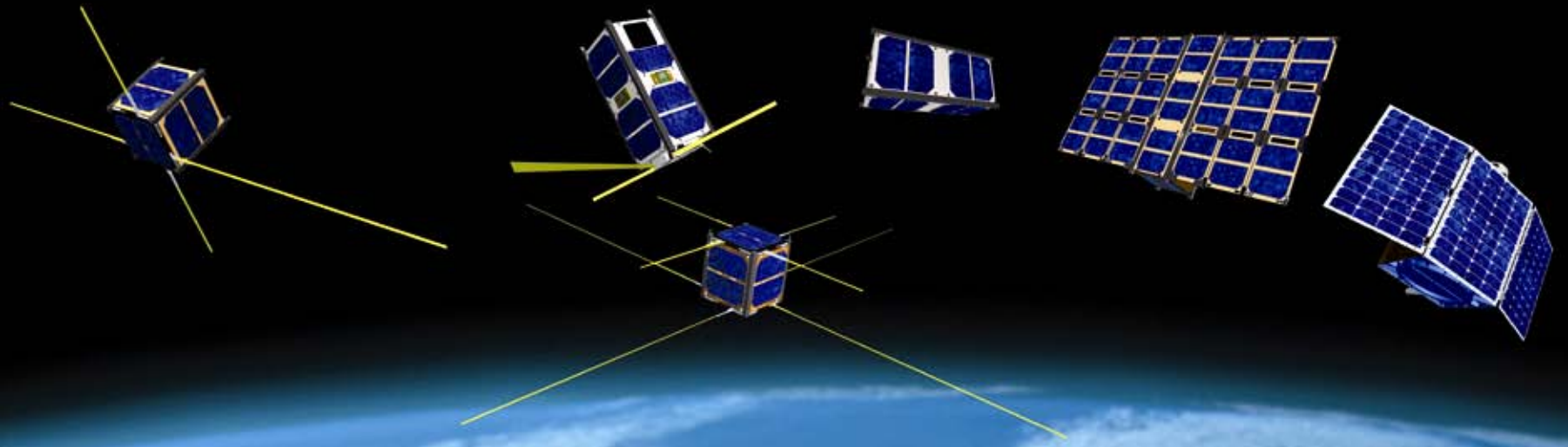




ISIS – Innovative Solutions In Space

Don't Worry, We'll Fix it in Software

Maxime Castéra





Summary

- The software Myths
- The software project cycle
- The flight software
- The embedded stack
- Sw / Hw interaction phases
- Subsystems VS Systems testing
- Pitfalls to be avoided
- Conclusion

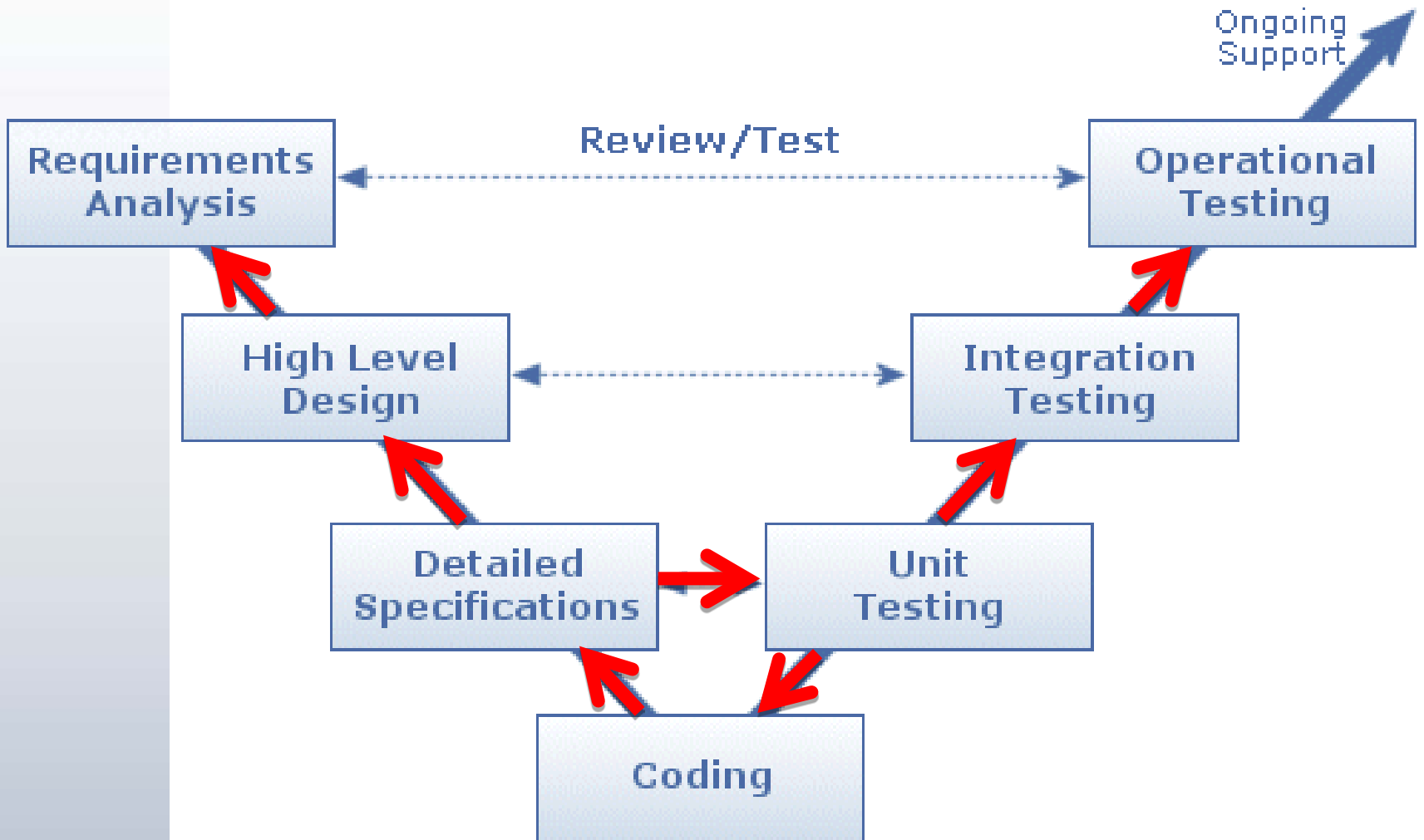


Software Myths

- Flexibility of the software
- Software effort estimation
- Re-usability
- Maturity / Testing of the software
- Bug fixing
- Fixing everything 'later' in software



The software project cycle



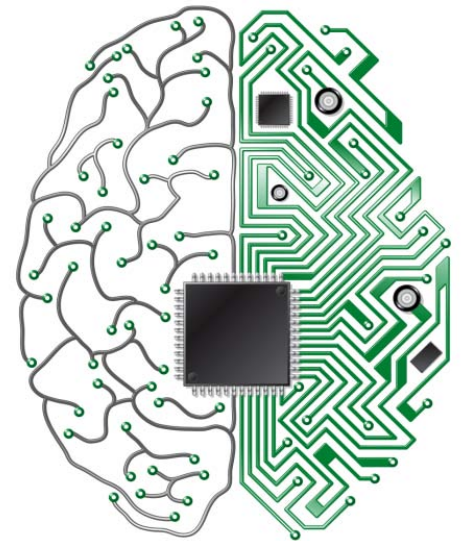


The software project cycle

- Flow-down software and hardware requirements from the mission requirements
=> Not the opposite.
- Involve the software team early on in the mission definition.
- Plan testing early enough.
- Document every step.

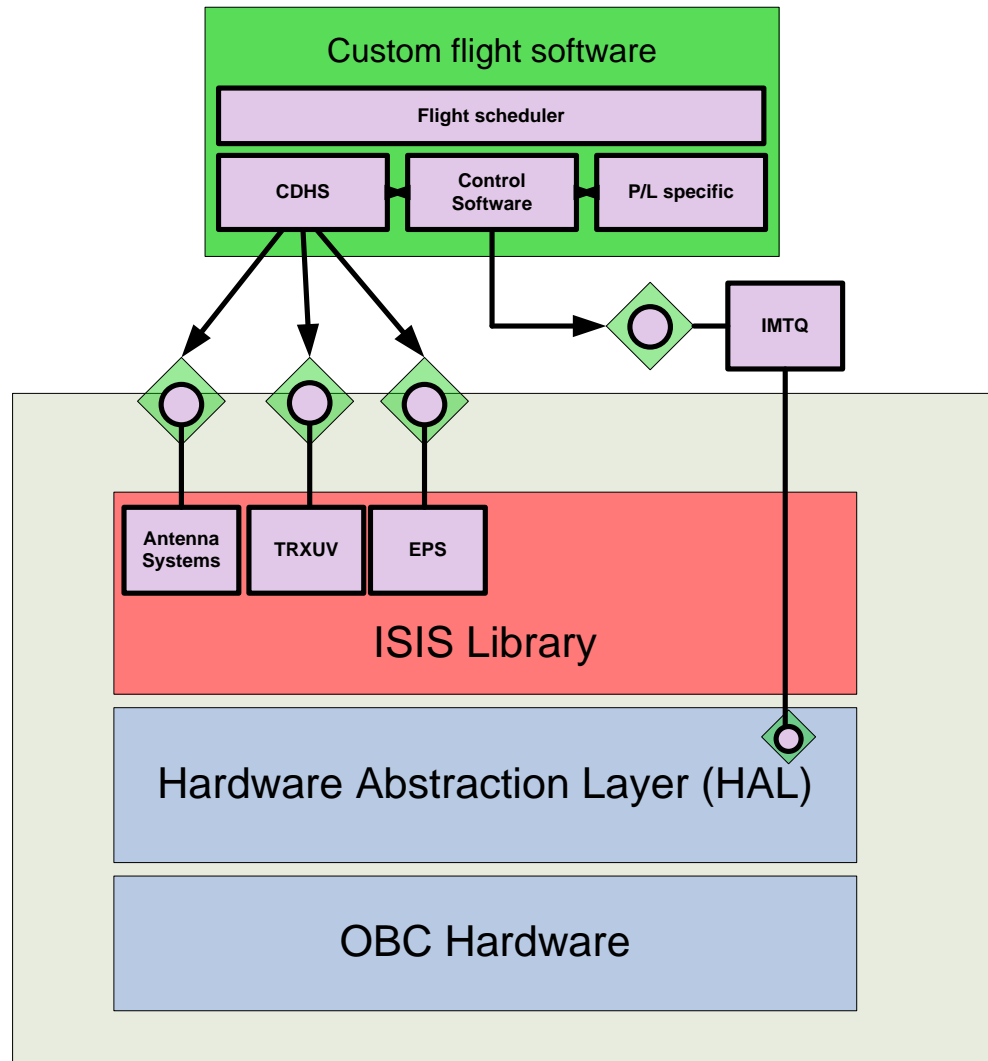
The flight software

- On-board Computer
 - Definition of the databus
 - Overall satellite operational mode
 - Flight scheduling
 - Command and Data Handling
- ADCS Computer
 - Sensors reading
 - Actuators commanding
 - Attitude determination algorithms
- Local intelligence of the subsystems
 - Housekeeping data collection
 - Command handling





The embedded stack





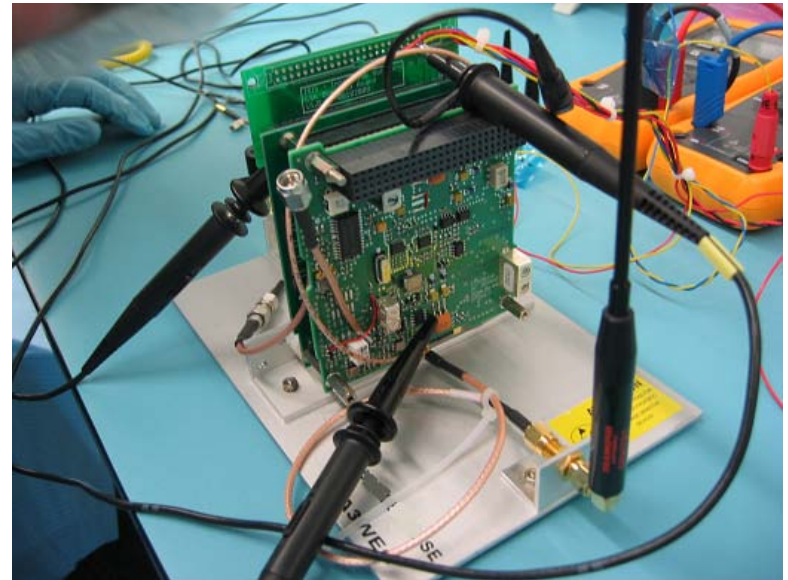
Sw / Hw interaction phases

- **Stubbing phase**
 - When hardware not available
 - I/F being defined
- **Development board phase**
 - When hardware not finalized or fully defined
 - I/F still open
- **Breadboard phase**
 - When hardware characterized and under-test
 - I/F frozen
- **EM phase**
 - When hardware on the table



Subsystems VS Systems testing

- Subsystems testing
 - Unit testing on embedded systems
 - Regression testing
- System testing
 - Flat sat setup
 - Hardware stubbing
 - Full stack testing
- E2E testing
 - Gaining uptime
 - Full chain testing





Pitfalls to avoid

- Involving software people too late.
- Involving software people too early.
- Underestimating the need for mission specific knowledge.
- Cutting corners on software testing.
- Excessively re-using old software.
- Changing databus philosophy late in the project.
- Assuming that writing flight software is the same as regular software development.
- Forgetting that your code will be in space.



Conclusion

- Software can't fix everything
- Proper interfaces are everything
- Involvement of the team is critical
- Educate the software team
- Let the software team educate you

- **An untested software is nothing else than a project risk**

