RESMED

Integrating lightweight software processes into a regulated environment

Adrian Barnes

Principal Engineer, ResMed Ltd

Agenda

- Software development at ResMed

- Oil & Water
  - Agile Lightweight Process
  - Formal Process
  - Will they Mix?

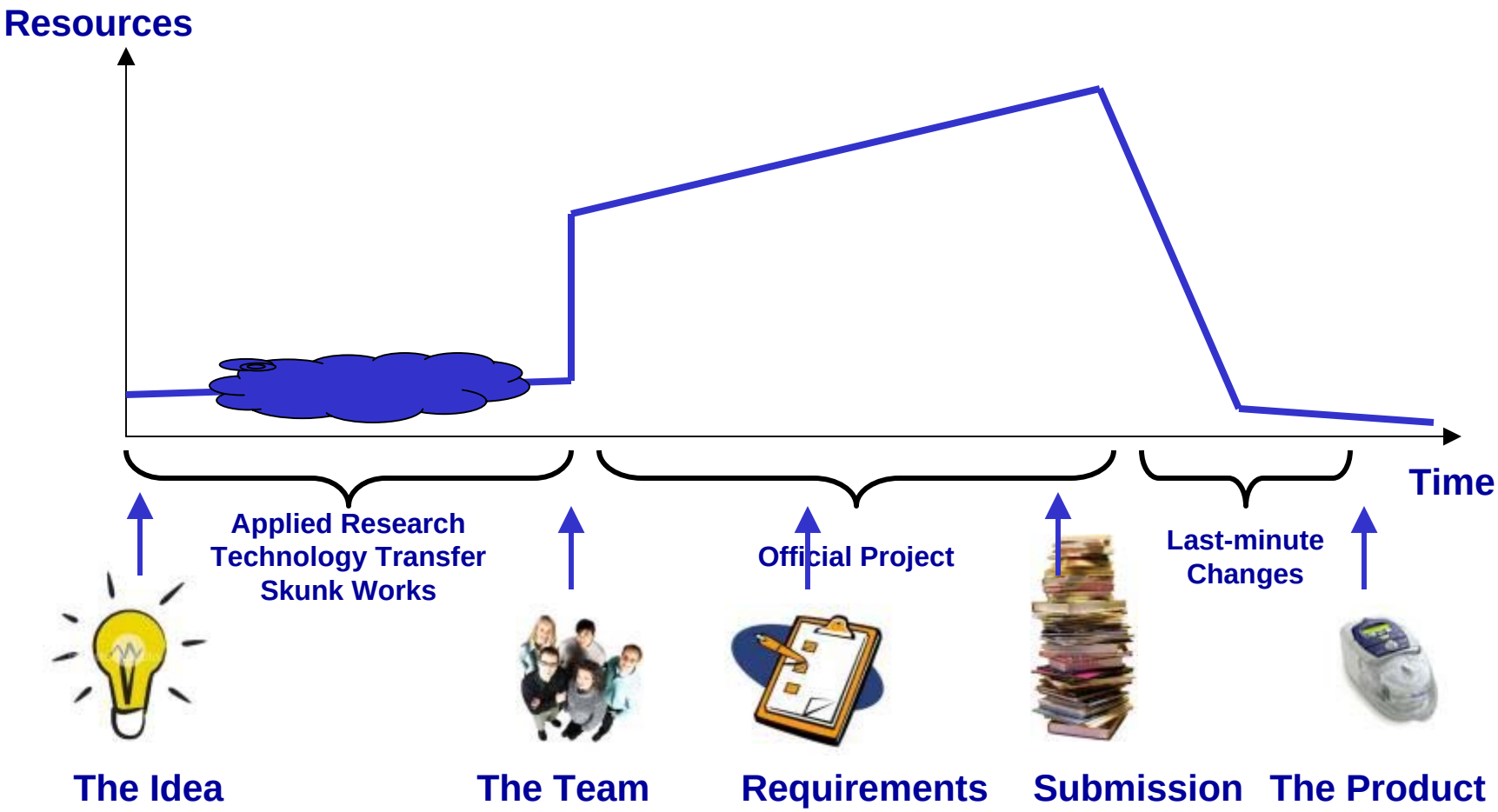
- Mixing processes
  - Problems in trying to mix methodologies

- Bridging the gap
  - A strategy for co-existing methodologies

About ResMed

- Buisiness
  - Medical Devices (Respiratory)
  - Global market
  - Regulated (by FDA and other bodies)
- Revenue
  - 2005 $A 550 million +
- Global software development team

A ‘typical’ medical device project



ResMed

Oil & Water

- 10 years ago things were simple(r)
  - All we had was water
    - Standards based on processes (unashamedly waterfall-like).
    - Software engineers only knew waterfall-like lifecycles
    - Regulators expected process artifacts (waterfall-like)
    - We didn't know of anything better

- Now
  - We have oil and water
    - Standards based on processes (waterfall, iterative, incremental)
    - Software engineers aware of alternative methodologies
    - Regulators expect process artifacts (waterfall-like)
    - We are aware that the 'old way' is not always the best way

“Oil”
Lightweight Processes

Agile Manifesto:

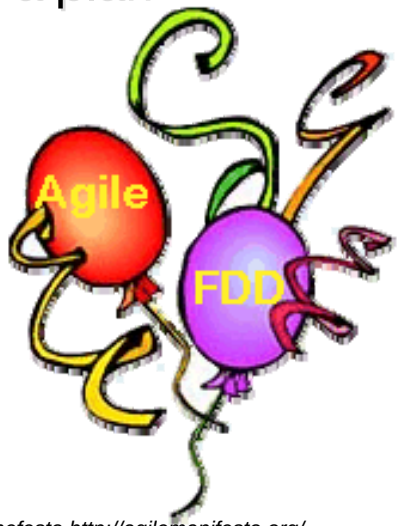
…….we have come to value

Individuals and Interactions over processes and tools

Working Software over comprehensive documentation

…Responding to Change over following a plan

- When I say Agile I mean Agile…..any methodology that shares these values.



Agile Manefesto http://agilemanifesto.org/

“Water”
Formal (Heavy) Processes

CMM(I) – processes based on two assumptions

- **A system is best managed by disaggregating it into defined work products that are converted from an input to an output state to achieve a specific design goal**
- **A mature Software Organization is one in which all activities are planned and then controlled to achieve specific design goals**

Regulators and auditors generally value:

Processes and Tools over individuals and interactions

Documentation over working software

Following a plan over responding to change

Can Agile and Formal processes co-exist?

Horses for Courses

AGILE

- People based
- Adaptive
- Unpredictable or rapidly changing requirements.
- Low criticality
- Senior developers

FORMAL

- Process based
- Predictive
- Stable and known requirements
- High criticality
- Junior developers

Boehm, B.; R. Turner (2004). *Balancing Agility and Discipline: A Guide for the Perplexed*. Boston, MA: Addison-Wesley. ISBN 0321186125.

Do we want Light & Heavy processes to coexist?

- Instead of trying to make XP work rationally with the firm's existing processes, each side is pointing fingers at the other.

  No one seems to be trying to apply XP within the SW-CMM context rationally and profitably as the sages have suggested ...

  XP adherents feel they don't have time for the formality ...

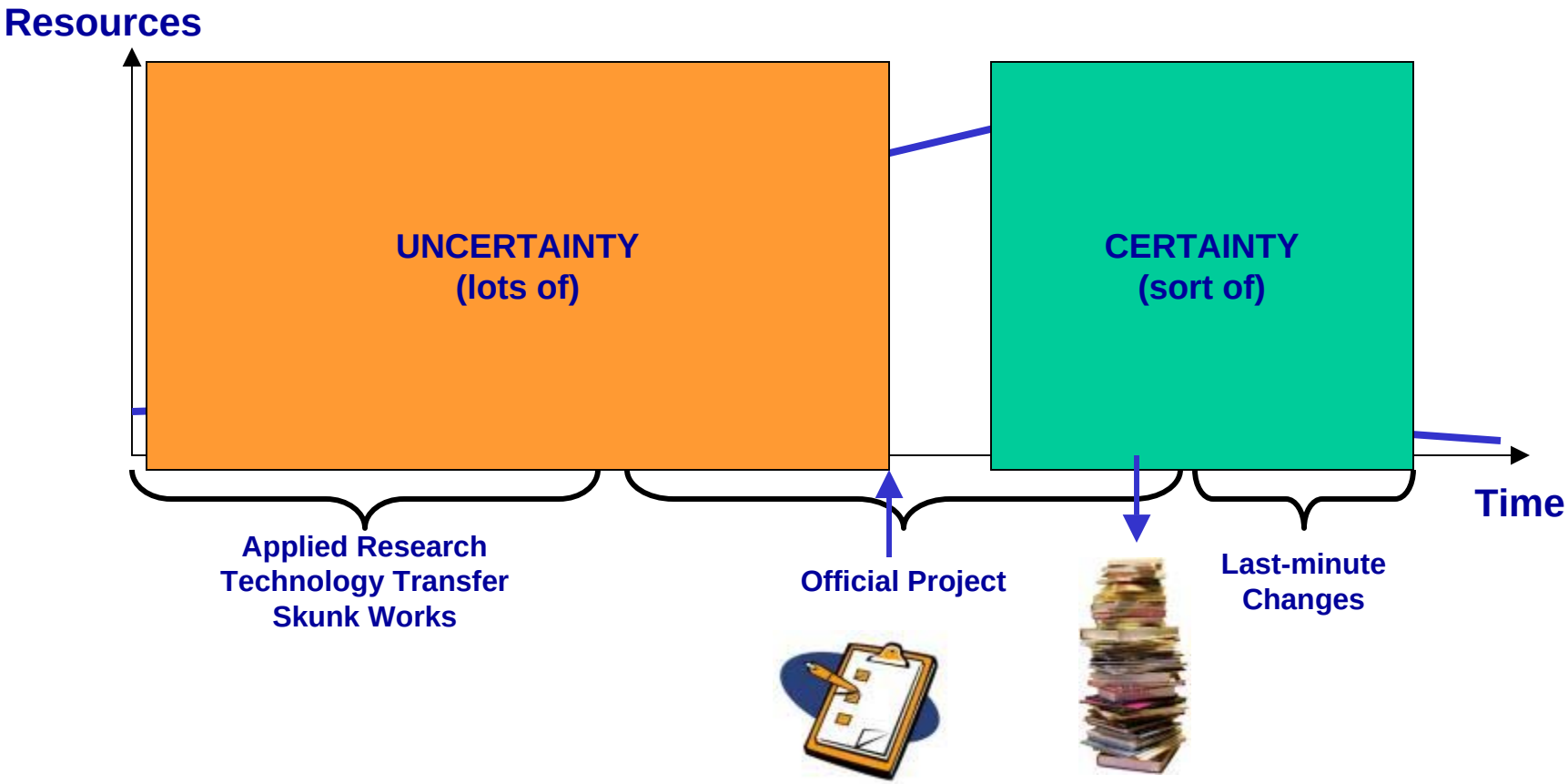
  Process proponents argue ... quality will suffer and customer satisfaction will be sacrificed.

Reifer, Don. "XP and the CMM." *IEEE Software* May/June 2003: 14-15.

Can we manage with just one or the other?

- Agile and Formal processes have different strengths
- If different methodologies are better for different problems
  - What happens if one project has characteristics that needs both Agile and Formal processes?
- If a project is best addressed with an Agile lifecycle
  - What do we show the regulators?
- If a project is best addressed with a planned incremental lifecycle
  - How do we manage project uncertainty?

An innovation project
One process or two?



RESMED

Mixing up methodologies

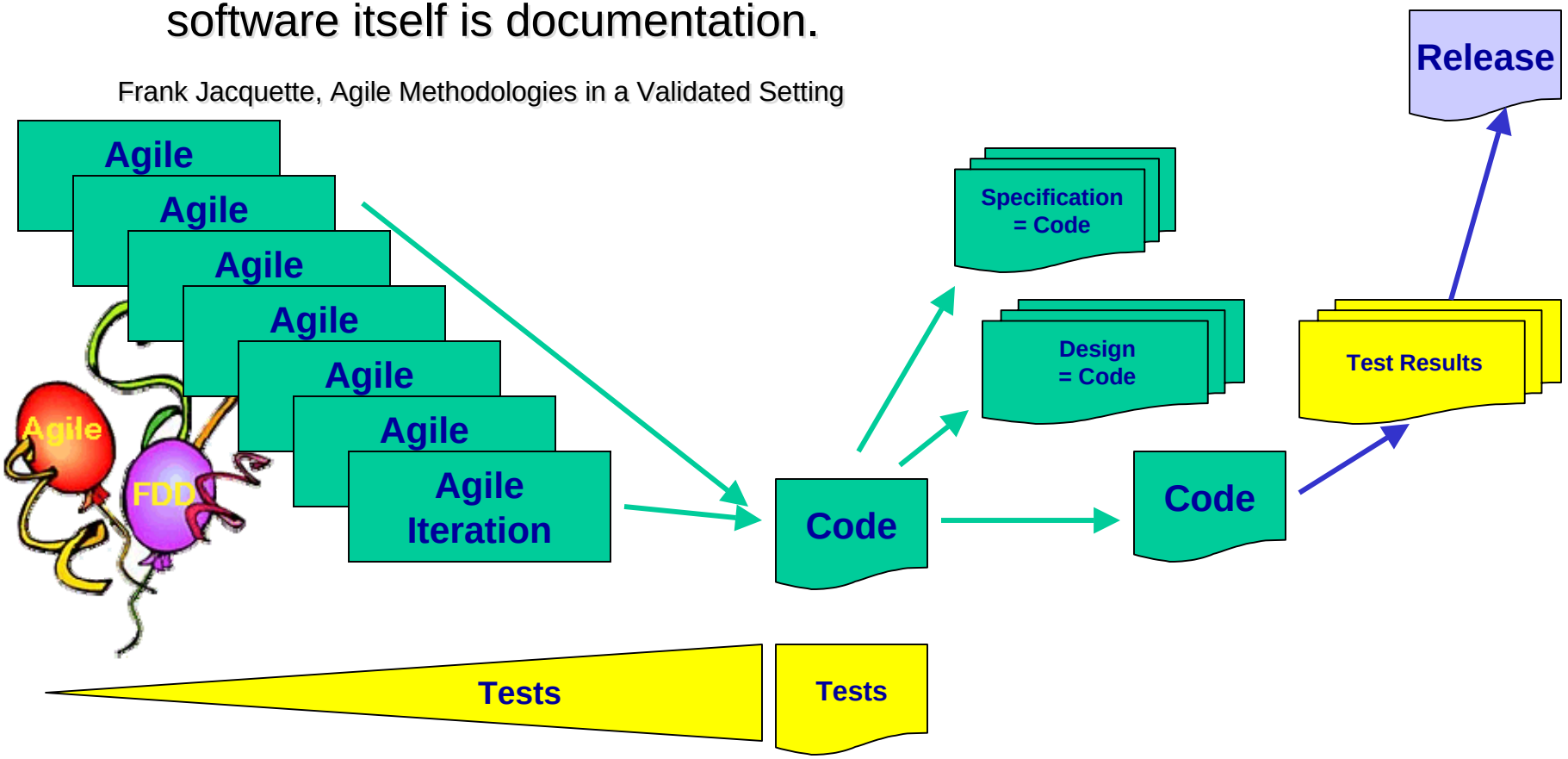
- Agile + Formal mixed = best of both worlds???
- Three approaches to mixing Agile processes into a formal process environment.
  - Redefine the formal environment
  - Convert to formal process late in the project
  - Merge Agile into the formal process
- Each approach either
  - Devalues Agile Process
  - Devalues Formal Process
  - Or both!

Mixing by redefinition

You want specs and documents?

- Writing code is part of the specification process, and the software itself is documentation.

Frank Jacquette, Agile Methodologies in a Validated Setting



http://www.jacquette.com/articles/agilevalidation.shtml

ResMed

Mixing by redefinition

You want specs and documents?

VERDICT: **NOT CONVINCED**

Pure Semantics

Ad hoc product functionality

Does Frank design avionics software?

Late conversion

Just a mere formality

Project Plans

Requirements

Assumed Requirements

Agile

Agile

Agile

Agile

Agile

Agile

Agile Iteration

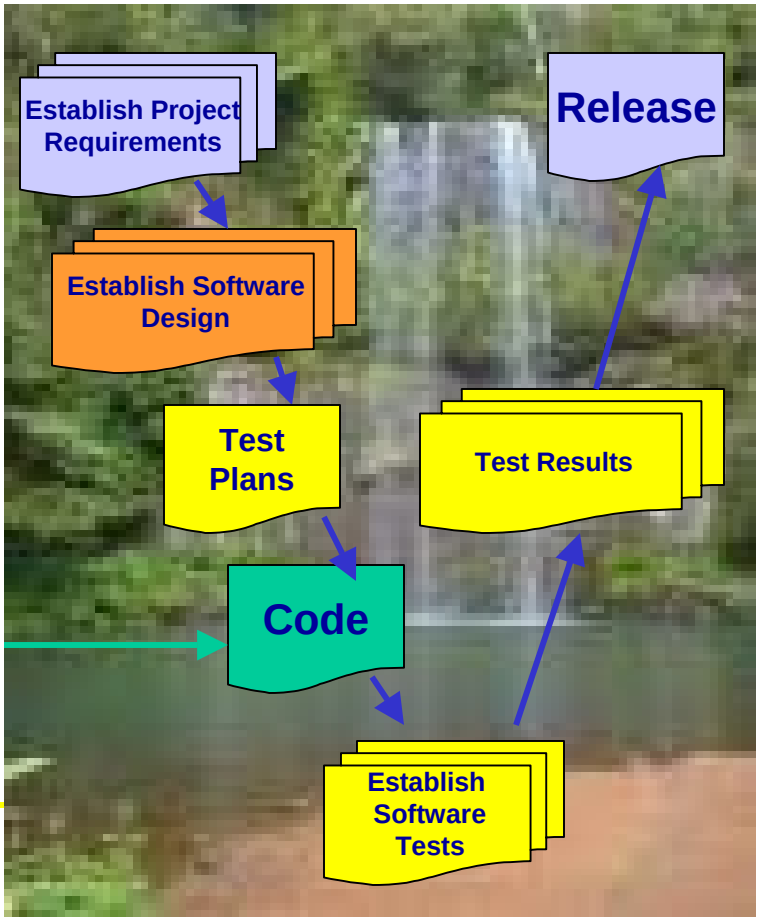
Agile

FDD

Code

Tests

Automated Tests



ResMed

Late conversion

Just a mere formality

VERDICT: **NOT CONVINCED**

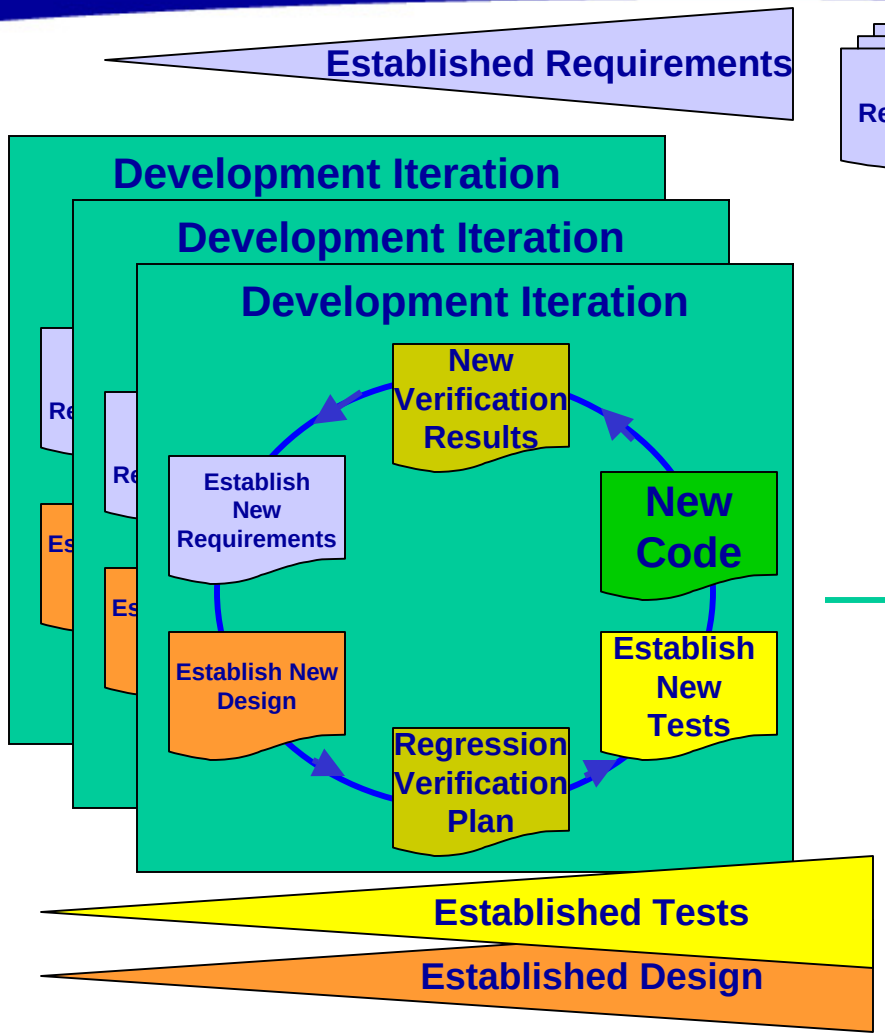
Testing in Quality at last minute

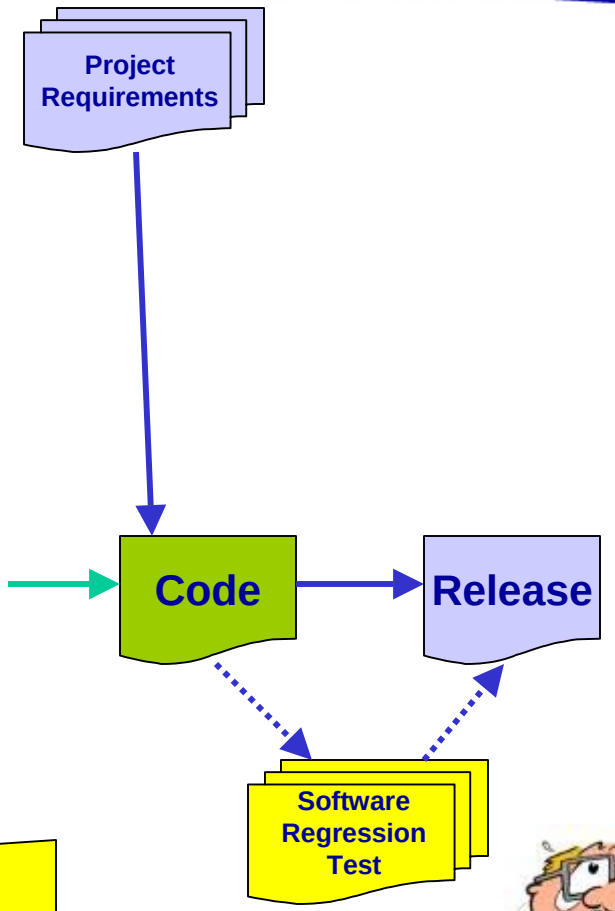
Just using formal lifecycle to create documents

A Waterfall for show rather than value

Make Agile Heavyweight!

You want documents not code





RESMED

Make Agile Heavyweight!

You want documents not code

VERDICT: **NOT CONVINCED**

Benefits of Agile are lost

Confidence in final product lost

More document rework than development

Anything but Lean!

If methods don't mix
Why not just use Agile?

- …. they never -- ever -- disclose the risks and the downsides. ……. The truth is that these practices come at a price, and for a lot of organizations, the price gets high very quickly. Agile development will never go far if its proponents keep ignoring these organizations and make condescending comments to its members.

  Posting: Joseph Ottinger on June 09, 2006
  www.theserverside.com

- Risks of using Agile processes are unknown
  - The risk reduction using formal processes is known
  - Fear of the unknown isn't a valid reason to avoid Agile completely

What is the risk in just using Agile?

Potential product risks mitigated by formal methodologies

- Functional Risk
  - Code does not actually do what is claimed
- Integrity Risk
  - Code may directly cause hazards, or may interfere with the correct operation of other parts of the software
- Compliance Risk
  - Code may not comply with legal requirements for its development
- Quality Risk
  - Code may not have the required quality attributes

Agile quality
Some expected attributes

Agile Quality attributes

- Changeability
  - Can the software be easily changed without compromising functionality?
- Testability
  - Can the design be easily tested?
- Acceptability
  - Does the prototype address the needs of differing user groups?
- Understandability
  - Does it adequately address human factors and usability?

Software Engineering — Product Quality — Part 1: Quality Model. ISO, Geneva, Switzerland, 2001. ISO/IEC 9126-1:2001(E)

Agile quality?
More challenging attributes

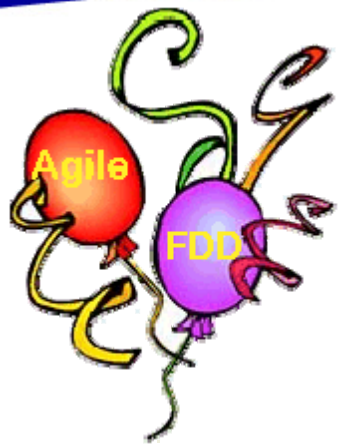
- Completeness - Does it do everything it claims to do?
- Accuracy -Does it actually do what it claims to do?
- Interoperability - Does it work with other system interfaces?
- Dependability - Is the implementation robust and fault tolerant?
- Maturity - Does it demonstrate a reduction in latent defects over time?
- Capability - Does it function correctly at limits of throughput and capacity?
- Consistency - Are all artifacts consistent with each other?
- Install-ability - Can it be efficiently be installed/manufactured without loss of quality?

Co-existing Processes

I propose that:

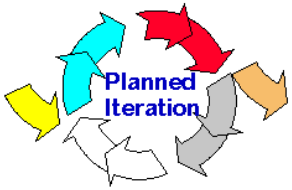
- Two contrasting processes can successfully co-exist provided that
  - The strengths of each process are respected
  - The project needs change significantly during the lifecycle, and
  - The interface between processes is managed

Separate project phases
Separate processes



High Project Risk Adaptive Phase

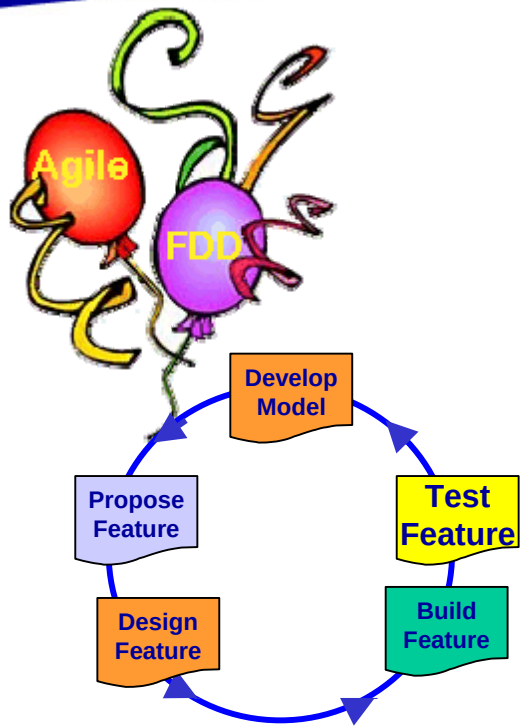
- Focus is on reducing project risk
  - Architecture, Algorithm, Features, Interfaces, Capability
- Deliverables are not just code – but also drafts of defining documents
- Ends when project risk has been lowered to an acceptable level

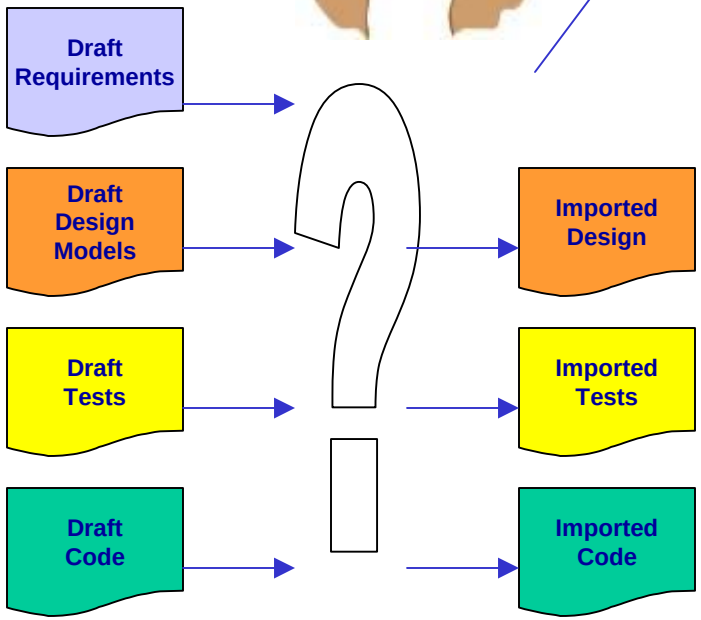


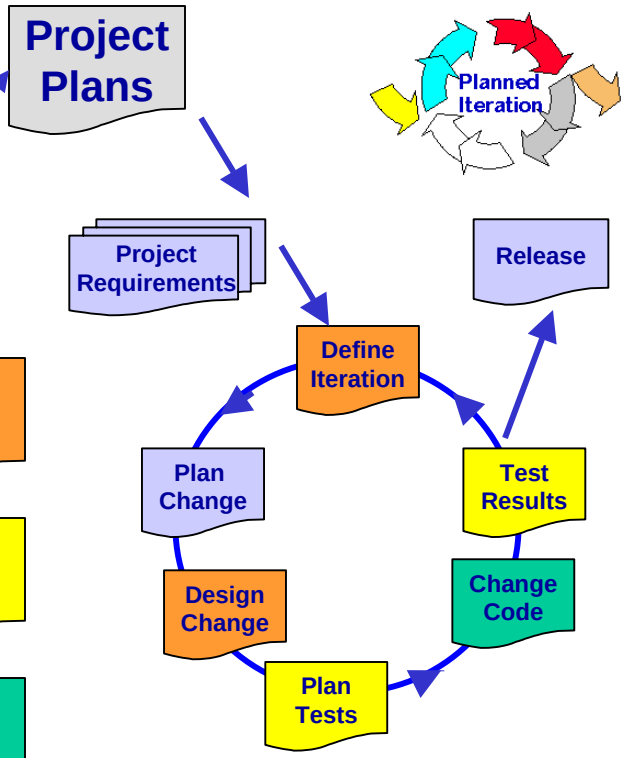
Low Project Risk Predictive Phase

- Focus is on reducing product risk
- Few Unknowns
- Planned iterations
- Deliverables are approved artifacts for 'customer'
- Ends when product risk has been lowered to an acceptable level

Managing the gap between two phases







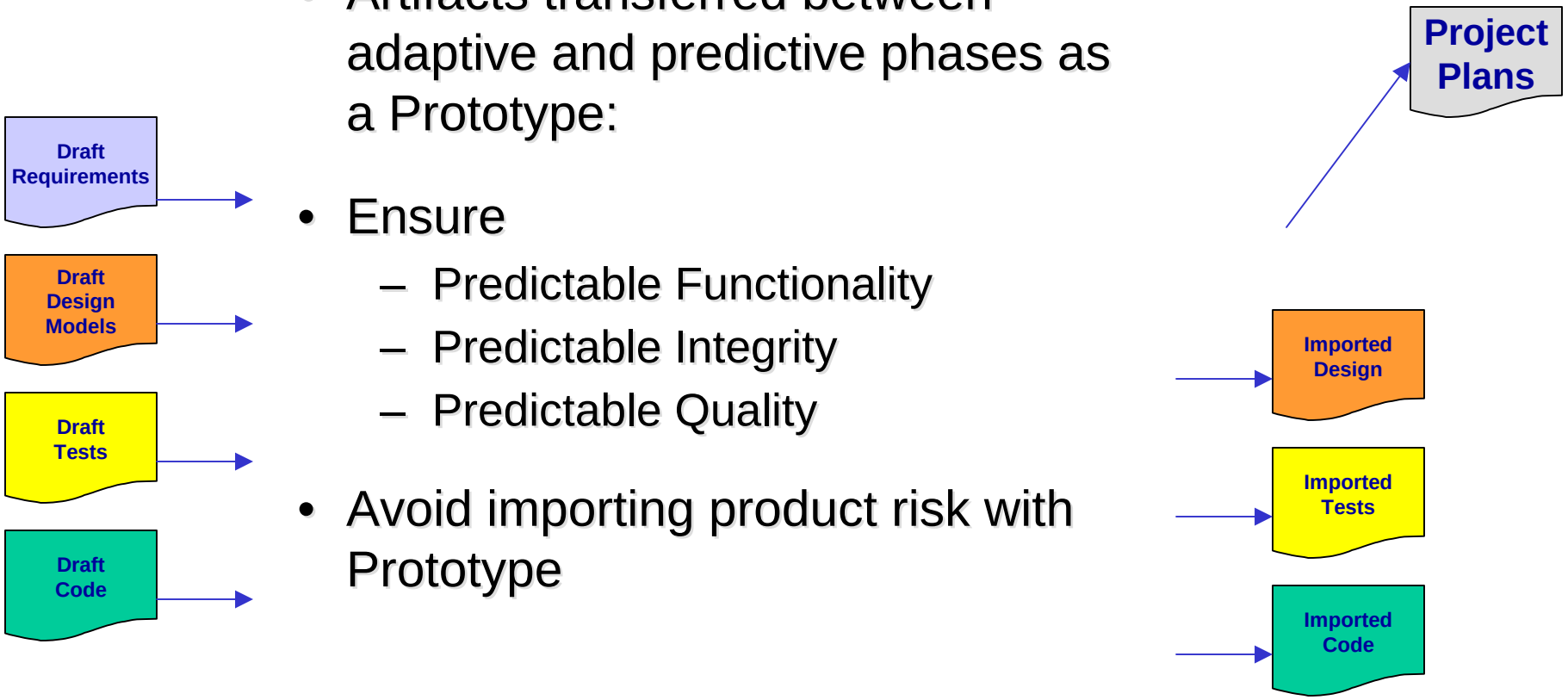
High Project Risk Adaptive Phase

Low Project Risk Predictive Phase

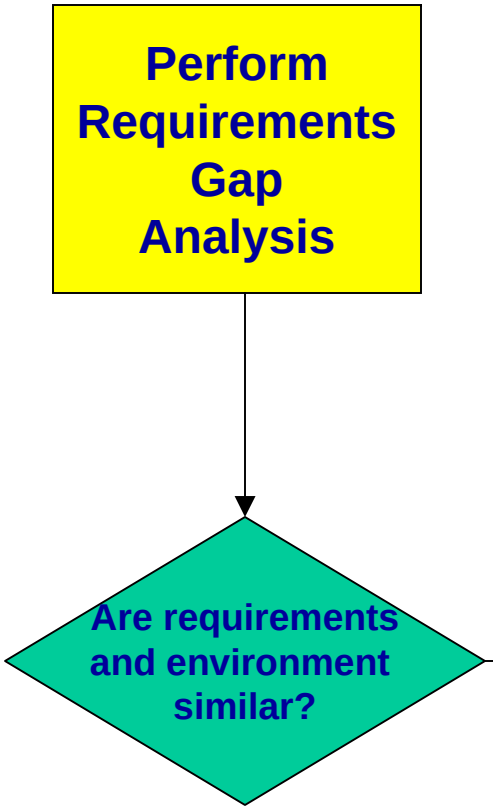
ResMed

Bridging the gap

- Artifacts transferred between adaptive and predictive phases as a Prototype:
- Ensure
  - Predictable Functionality
  - Predictable Integrity
  - Predictable Quality
- Avoid importing product risk with Prototype



Bridging the gap



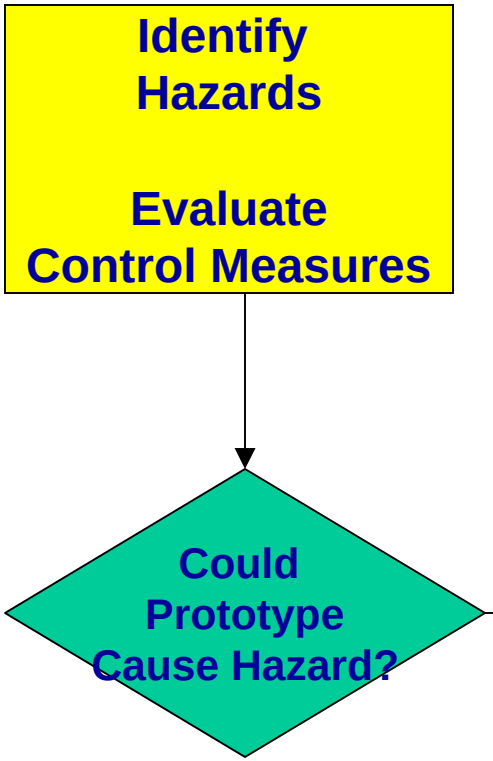
Functional Risk:

- Do we know what the prototype software is supposed to do?
- Are the project requirements similar to prototype requirements?
- Is the environment used for the prototype similar to the project target?

NO?

– Stay in first phase
or if so bad just

Bridging the gap



Integrity Risk:

- Can the prototype itself cause hazards
  physical harm, corporate harm, financial harm
- Could the prototype cause other parts of our system to cause hazards…
- Will we be relying on the prototype to mitigate hazards?

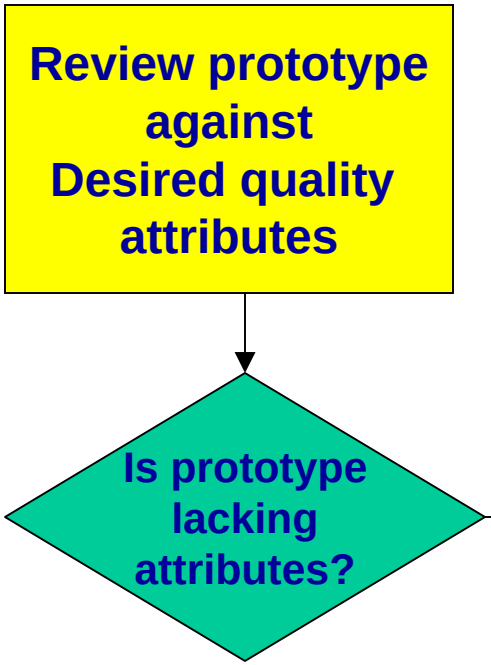
YES?

  – Find those parts of the prototype and throw the code away!

Bridging the gap

Quality Risk:

- Does the prototype lack quality attributes that could result in product risk…



YES?

- Develop a plan to improve deficient attributes
- Develop a plan to verify that the remedial actions have been successful.

Bridging the gap

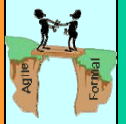
Example Quality Attributes Review Criteria

Attribute	Question
Completeness	Does the prototype fully address its assumed requirements?
Accuracy	Does the prototype correctly address each assumed requirement?
Interoperability	Is the prototype compatible with other system interfaces?
Dependability	Is the prototype adequately robust and fault tolerant?
Maturity	Does the prototype demonstrate a reduction in latent defects over time?
Capability	Will prototype function correctly at the limits of throughput and capacity?
Consistency	Are all the draft artifacts consistent with each other?
Install-ability	Can the prototype be installed/manufactured without loss of quality?

One project – 2 methodologies

Resources

AGILE PROCESSES



FORMAL PROCESSES

Time

Applied Research
Technology Transfer
Skunk Works

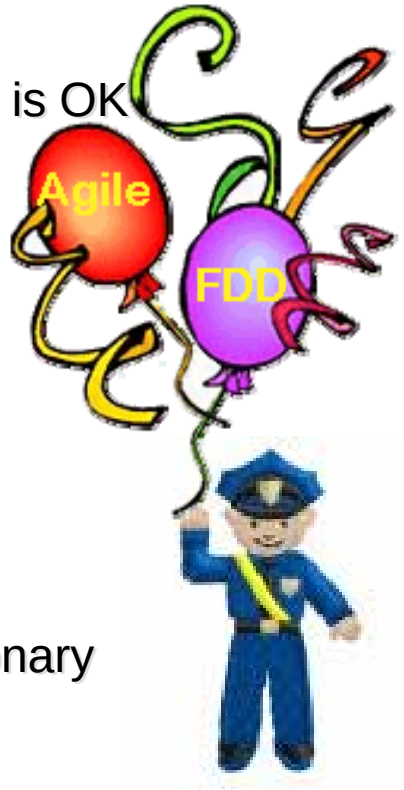
Official Project

Last-minute Changes

ResMed

Summary

- Lightweight processes and formal regulatory environments do not mix!
  - A brave man would try to convince the FDA that Agile is OK
- Agile and Formal processes do not mix!
  - But they can co-exist
- A strategy
  - Use Agile processes to reduce Project risk
  - Transfer a prototype with draft documents
  - Review prototype for Product Risk
    - Functional, Integrity, Quality (and Compliance)
  - Introduce accepted drafts into formal iterative/evolutionary lifecycle as appropriate
  - Use formal processes to reduce Product Risk



Questions

My email

abarnes@resmed.com.au