




**DSDM**  
CONFERENCE



***Agility requires Discipline;  
Formality is not Discipline***

**Andrew Craddock**

12<sup>th</sup> October 2004

andrew.craddock@projectagility.co.uk www.dsdm.org

---

---

---

---

---

---

---

---

**Configuration Management...**

...is the discipline of controlling the evolution of complex systems.

...came into existence in the U.S. defence industry where it was used to control manufacturing processes.

...exerts control by managing change within a defined process thereby increasing predictability of the outcome.

12<sup>th</sup> October 2004

andrew.craddock@projectagility.co.uk www.dsdm.org

---

---

---

---

---

---

---

---

**Software Configuration Management...**

...is intended to control the development and modification of software systems and associated products throughout their lifecycle.

...assumes a traditional command and control management style in a defined approach.

...is often seen as an unnecessary, inflexible bureaucratic process designed to discourage change.

12<sup>th</sup> October 2004

andrew.craddock@projectagility.co.uk www.dsdm.org

---

---

---

---

---

---

---

---



### Traditional approach

**n** Waterfall based:

**n** Origin: Engineering

**n** Adopted in simpler times

- § technology better understood (mainframes)
- § requirements better understood (computers replacing people...)
- § change low

12<sup>th</sup> October 2004

andrew.craddock@projectagility.co.uk www.dsdm.org

---

---

---

---

---

---

---

---

---

---

### Simplicity...

**n** The further from certainty; the greater the likelihood of change.

**n** When was the last simple project of any size?  
- 1970s?

**n** CM exerts control by working to stabilise requirements and technology (theoretically moving projects closer to the origin on the graph)

**n** But is this really the right thing to do?

Modified Stacey graph - Ken Schwaber 2002

12<sup>th</sup> October 2004

andrew.craddock@projectagility.co.uk www.dsdm.org

---

---

---

---

---

---

---

---

---

---

### Innovation drives business

**n** U.S. Companies, New Product Time to Market

- § 1990 avg. = 35.5 months
- § 1995 avg. = 23 months
- § 2000 avg. = 11 months
- § 2005 ?

**n** Cars, Concept to Production

- § 1990 avg. = 6 years
- § 2001 avg. = < 2 years
- § Renault's goal = 9 days!!!

**n** Software ?

Source, Tom Wujec & Sandra Muscat, *Return on Imagination*

12<sup>th</sup> October 2004

andrew.craddock@projectagility.co.uk www.dsdm.org

---

---

---

---

---

---

---

---

---

---



## IT systems and the dynamic business

- n Business survive by being able to respond to change; they gain competitive advantage through innovation (i.e through pro-active change).
- n Computer systems are increasingly being used to support processes at the leading edge of businesses.
- n Computer systems need to be able to change fast in order to support the changing business.
- n Processes supporting such change need to be dynamic and responsive.

They need to be **AGILE**

12<sup>th</sup> October 2004  
andrew.craddock@projectagility.co.uk [www.dsdm.org](http://www.dsdm.org)

---

---

---

---

---

---

---

---

## What is agility

- n Agility is about *creating and responding to change*.
- n Agility is about *flexibility and discipline*.
- n Agility is about *innovation*.

*"Agility is dynamic, context-specific, aggressively change-embracing, and growth-oriented.*

*It is not about improving efficiency, cutting costs, or battering down the business hatches to ride out fearsome competitive storms."*

Agile Competitors and Virtual Organizations, by Goldman, Nagel, and Preiss, 1995.

12<sup>th</sup> October 2004  
andrew.craddock@projectagility.co.uk [www.dsdm.org](http://www.dsdm.org)

---

---

---

---

---

---

---

---

## Manifesto for Agile Software Development

[www.agilemanifesto.org](http://www.agilemanifesto.org)

We are discovering better ways of developing software by doing it and helping others to do it. Through this work we have come to value:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

© 2001 Kent Beck, Mike Beedle, Eric van Breda, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Melor, Ken Schwaber, Jeff Sutherland, Dave Thomas

12<sup>th</sup> October 2004  
andrew.craddock@projectagility.co.uk [www.dsdm.org](http://www.dsdm.org)

---

---

---

---

---

---

---

---



**Why the bad press about CM?**

<p><b>n Bureaucratic</b></p> <p><b>Bureaucracy:</b> def: Management or administration marked by hierarchical authority among numerous offices and by fixed procedures.</p>	<p><b>CMII Principles (excerpt)</b></p> <ul style="list-style-type: none"> <li>Knowing the ability to communicate is not size limited if using documents, forms, records and data properly.</li> <li>Knowing the highest level of integrity is achieved when documents are validated by actual users.</li> <li>Knowing that documentation should always lead and physical items must conform.</li> <li>Knowing lifecycles begin and end with documented requirements.</li> <li>Knowing the primary product of engineering is documentation.</li> <li>Knowing the purpose of a prototype is to validate the documentation.</li> </ul>
<p><b>n Document Driven</b></p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p>Clearly at odds with the Manifesto for Agile Software Development</p> </div>	

12<sup>th</sup> October 2004 10

andrew.craddock@projectagility.co.uk www.dsdm.org

---

---

---

---

---

---

---

---

---

---

---

---

**Why the bad press about CM?**

<p><b>n Bureaucratic</b></p> <p><b>Bureaucracy:</b> Management or administration marked by hierarchical authority among numerous offices and by fixed procedures.</p>	<p><b>CMII Principles (excerpt)</b></p> <ul style="list-style-type: none"> <li>Knowing the ability to communicate is not size limited if using <del>documents, forms, records and data</del> <b>prototypes</b> properly.</li> <li>Knowing the highest level of integrity is achieved when <del>documents</del> <b>prototypes</b> are validated by actual users.</li> <li>Knowing that <del>documentation</del> <b>prototypes</b> should always lead and physical items must conform.</li> <li>Knowing lifecycles begin and end with <del>documented requirements</del> <b>working software</b>.</li> <li>Knowing the primary product of engineering is <del>documentation</del> <b>working software</b>.</li> <li>Knowing the purpose of a prototype is to validate the <del>documentation</del> <b>requirements</b>.</li> </ul>
<p><b>n Document Driven</b></p> <p><b>Most Agile proponents would probably be comfortable with this substitution.</b></p> <p><b>It is difficult to see how traditional configuration managers would...</b></p>	

12<sup>th</sup> October 2004 11

andrew.craddock@projectagility.co.uk www.dsdm.org

---

---

---

---

---

---

---

---

---

---

---

---

**Because Agilists believe**

**n Documentation** is not the same as **Understanding**

**n Formality** is not the same as **Discipline**

**n Process** is not the same as **Skill**

and they often observe

**n Documentation** *substituted for* Understanding

**n Formality** *substituted for* Discipline

**n Process** *substituted for* Skill

**They believe it is this, and not poor configuration management, that is the primary reason for chaos and failure of IT projects.**

12<sup>th</sup> October 2004 12

andrew.craddock@projectagility.co.uk www.dsdm.org

---

---

---

---

---

---

---

---

---

---

---

---



**Is CM important in an agile regime ?**

**Yes !**

It is absolutely vital

but it has a different focus

12<sup>th</sup> October 2004  
andrew.craddock@projectagility.co.uk www.dsdm.org

---

---

---

---

---

---

---

---

**Agile SCM must facilitate...**

...a shared understanding of exactly what makes up all elements of your solution baselines.  
Including software, models, test scripts, test data, documentation etc.

...a shared confidence that in the event of a problem you can safely return to any previously baselined state. Let's revert to yesterday's baseline; we know that was OK

...a shared knowledge of what has changed but **not** place a *restriction on or control of it.*

12<sup>th</sup> October 2004  
andrew.craddock@projectagility.co.uk www.dsdm.org

---

---

---

---

---

---

---

---

**DSDM CM Strategy**

**Options**

- n Baselining every prototype before demonstration.
- n Baselining daily, which enables flexibility but can be very onerous.
- n Baselining software items once they have been unit-tested.
- n Baselining at the end of a timebox, which is the absolute minimum.

12<sup>th</sup> October 2004  
andrew.craddock@projectagility.co.uk www.dsdm.org

---

---

---

---

---

---

---

---



### Change Control and CM

**n** There is a need to ensure that the scope of the work is strictly controlled whilst ensuring that change at the detailed level is unconstrained.  
Fitness for business purpose is the essential criterion for the acceptance of deliverables (DSDM Principle no 4)

**n** The detail of the requirements will be incrementally elicited and refined.

**n** All team members must be aware of the latest state of the requirements and the functionality that meets those requirements.

**n** Hence the Prioritised Requirements List should be considered as a Configuration Item related to the software solution.

12<sup>th</sup> October 2004

andrew.craddock@projectagility.co.uk www.dsdm.org

---

---

---

---

---

---

---

---

### Managing Change in DSDM

**n** Change to Scope  
§ Managed by Project Manager.  
§ Decisions made formally & collectively by  
- Executive Sponsor,  
- Visionary,  
- Technical Coordinator.

**n** Change to Detail  
§ managed by Team Leader.  
§ Decisions made informally and collectively by  
- Ambassador User  
- Developer.

**n** Configuration Management  
§ Responsibility of the Technical Coordinator throughout  
§ Enacted by the development team

12<sup>th</sup> October 2004

andrew.craddock@projectagility.co.uk www.dsdm.org

---

---

---

---

---

---

---

---

### People and Discipline

**n** **Not Command and Control**  
- DSDM team members must be empowered to make decisions\* without bureaucratic overheads

**n** **Not Formal**  
- Change is inevitable as iterative and incremental development facilitates convergence on an accurate business solution\*  
- The CM methods and tools must not hinder development

**n** **Necessary**  
- All changes during development of an increment are reversible\*

**n** **Disciplined**  
- Any release/prototype must be reproducible, including build information, test data and scripts, expected results, etc.

\* Reflecting principles 2, 5 and 6 of DSDM

12<sup>th</sup> October 2004

andrew.craddock@projectagility.co.uk www.dsdm.org

---

---

---

---

---

---

---

---



## Agile Configuration Management

must align with the Manifesto for Agile Software Development and demonstrably value:

**Individuals and interactions** over processes and tools

**Working software** over comprehensive documentation

**Customer collaboration** over contract negotiation

**Responding to change** over following a plan

12<sup>th</sup> October 2004

andrew.craddock@projectagility.co.uk

www.dsdm.org

---

---

---

---

---

---

---

---

## Thankyou

Yes

No

Thoughts...

Maybe...

Yes but...

Questions...

What if...

Coffee...

12<sup>th</sup> October 2004

andrew.craddock@projectagility.co.uk

www.dsdm.org

---

---

---

---

---

---

---

---