Software Development Practices for Climate Models:

What we've learned

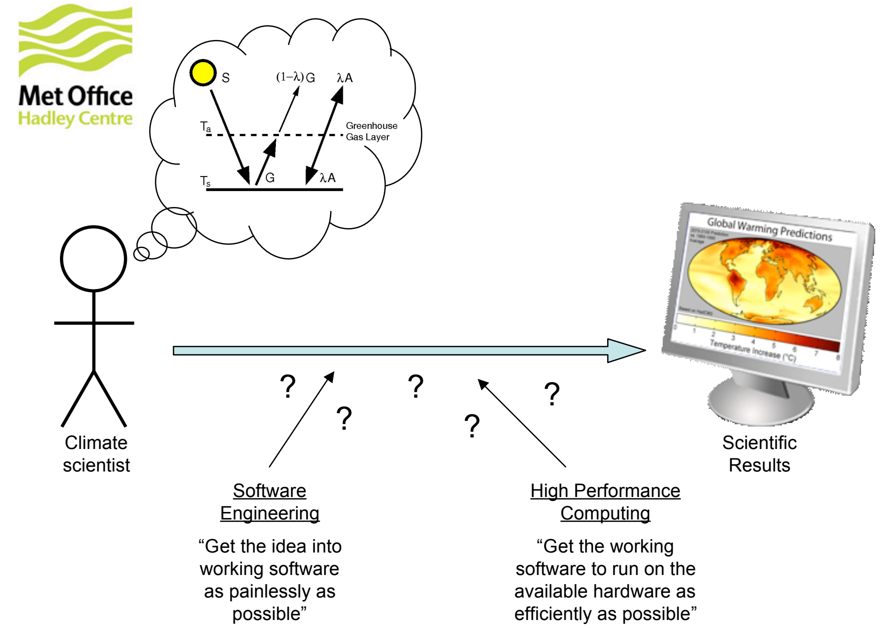
Steve Easterbrook

University of Toronto & Met Office Hadley Centre

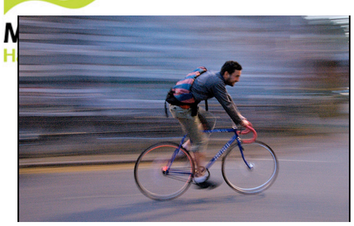
Contents

This presentation covers the following areas

- Current UM Development Practices
- Key issues:
  - Code Management & Coordination
  - Validation and Verification
  - Collaborations with other labs
- Risks and Opportunities
- Future Work



Assessing Software Quality

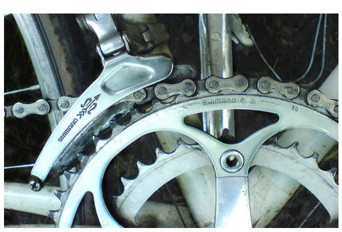


Quality in Use

(Does it suit its intended purpose?)

External Quality Attributes

(Does it pass all the tests?)



Internal Quality Attributes

(Is it well-designed?)

Process Quality

(Are we following best engineering practice?)

Hadley study: initial questions

- Correctness
  - How do scientists assess “correctness” of the code in their models?
- Reproducibility
  - How do they ensure experiments can be reproduced?
- Shared Understanding
  - How do they develop and maintain a shared understanding of large complex codes?
- Prioritization
  - How do they prioritize their work?
- Debugging
  - How do they detect (and/or prevent) errors in the software?

Philosophical Status of Climate Models

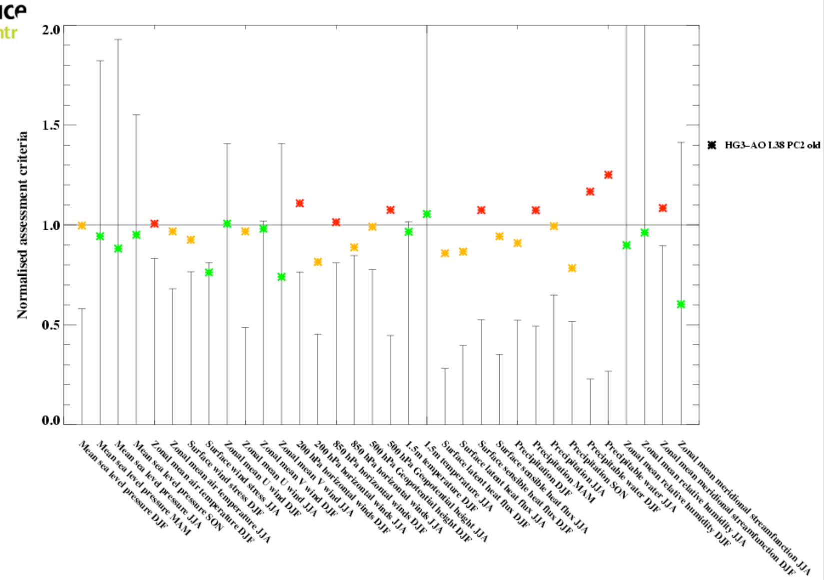
- Climate is a complex system
- Sources of Uncertainty:
  - Measurement Error
  - Variability in the physical processes
  - Model imperfections
- Imperfection of models is routinely accepted
  - Many different types of model
  - Many choices of resolution, timescale, science
  - Scientists continually select their abstractions

Quality = Fitness for Purpose

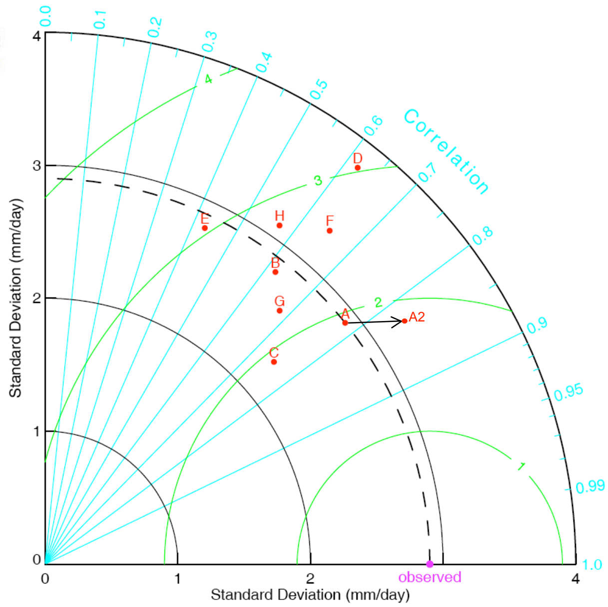
- Purpose of Earth System Models:
  - “To test our understanding”
  - “To quantify uncertainty”

All models are wrong,
but some are useful
- George Box

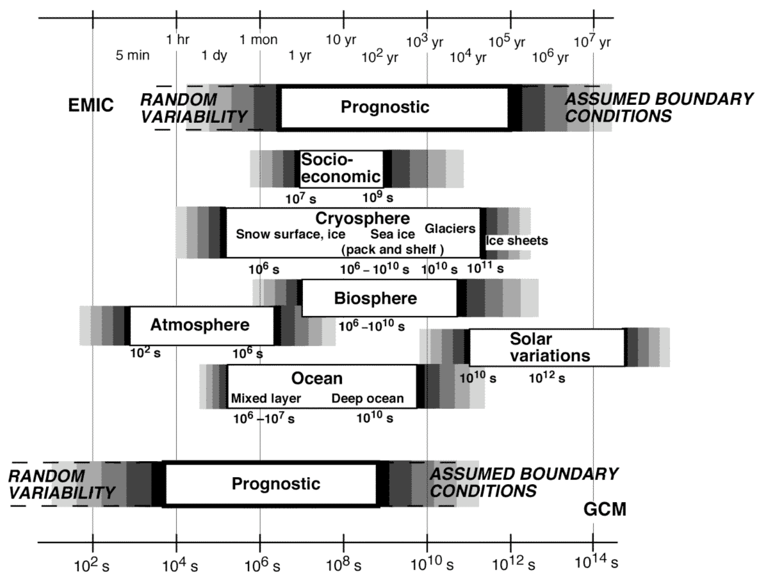
Summarizing skill gain



Taylor Diagrams

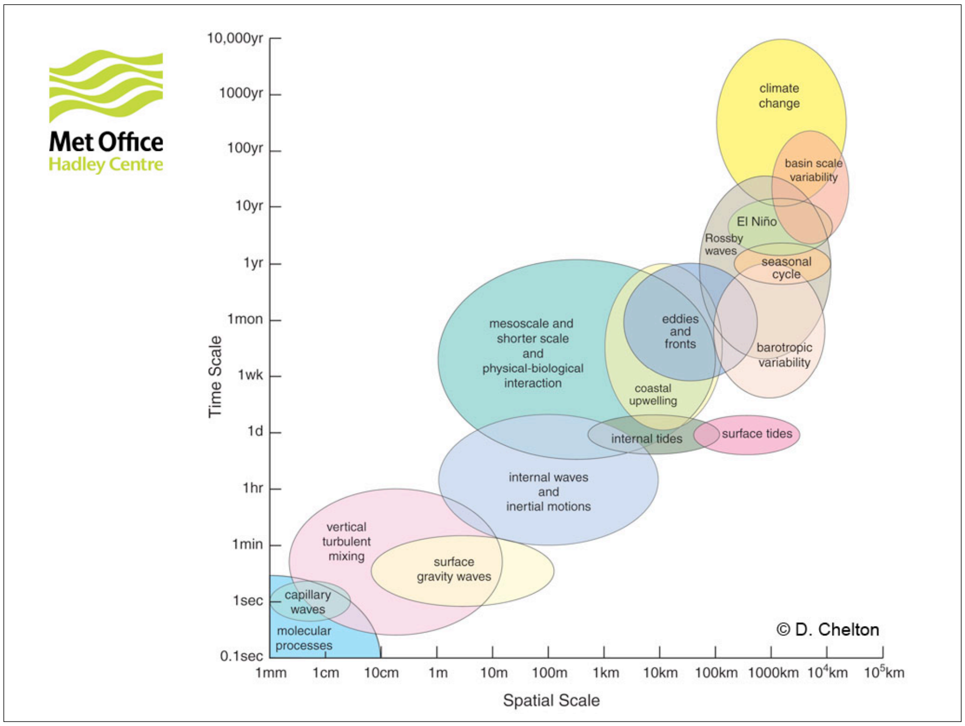


Time Scales

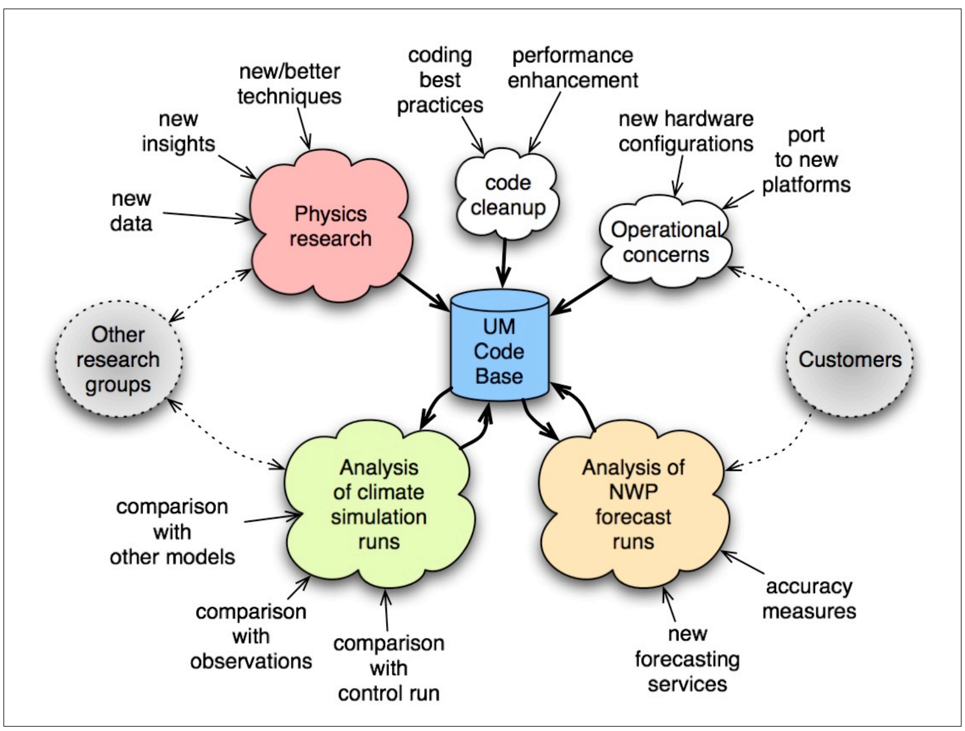


Timescale of climate processes

(Source: McGuffie & Henderson-Sellers, 2005)



Met Office
Hadley Centre
10,000yr
1000yr
100yr
10yr
1yr
1mon
1wk
1d
1hr
1min
1sec
0.1sec
Time Scale
climate change
basin scale variability
El Niño
Rossby waves
seasonal cycle
eddies and fronts
barotropic variability
mesoscale and shorter scale and physical-biological interaction
coastal upwelling
internal tides
surface tides
internal waves and inertial motions
vertical turbulent mixing
surface gravity waves
capillary waves
molecular processes
© D. Chelton
1mm
1cm
10cm
1m
10m
100m
1km
10km
100km
1000km
10^4km
10^5km
Spatial Scale



new/better techniques
coding best practices
performance enhancement
new insights
new hardware configurations
port to new platforms
new data
Physics research
code cleanup
Operational concerns
Other research groups
UM Code Base
Customers
Analysis of climate simulation runs
Analysis of NWP forecast runs
comparison with other models
comparison with observations
comparison with control run
new forecasting services
accuracy measures

Met Office
Hadley Centre

Some Conflicting Goals

- Same code used for Weather Prediction and Climate Research
  - NWP: Must be fast, give accurate forecasts
  - CR: Must be fast, reproducible, scientifically valid
- Components with different origins:
  - developed in-house (tightly controlled)
  - consortium models
  - community models (cf open source)
- Code Forking

Met Office
Hadley Centre

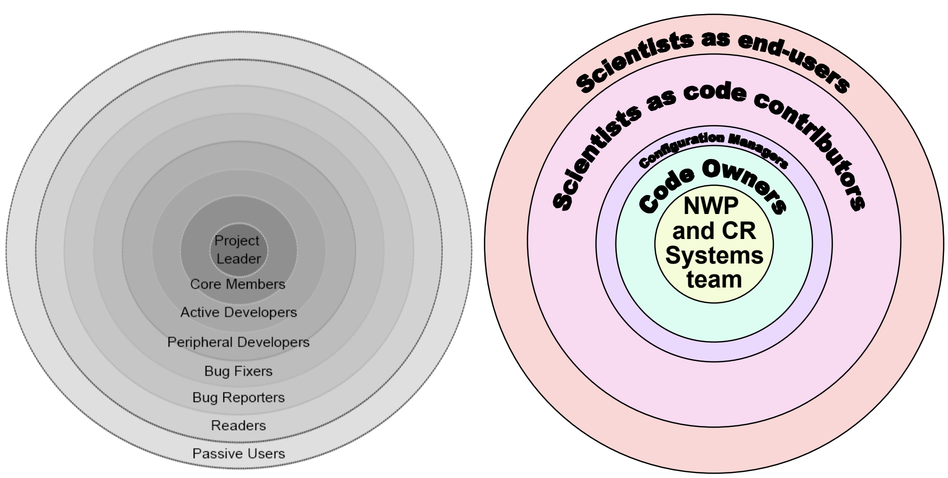
Code Management Tools

- Subversion - version tracker
- Trac - simplified bug tracker, wiki, and source browser
- Xxdiff - graphical diff and code merge
- Custom UI - simplifies process for branch and merge
- Custom Fortran 9X build system - simplified script to generate makefiles
- Custom code extract system - merge code from different branches and external sources

Coordination "informalisms"

- Core GCM developed in-house
  - Transplanted modules (e.g. MOM) are 'naturalized'
  - (Is this changing with NEMO, UKCA, Jules,…)
- Single, large, open plan office environment
- Many communication channels:
  - wiki, newsgroups, email, Trac
  - 'talk to the expert'
  - cross-functional, interdisciplinary teams
  - open meetings, workshops, etc.

Team Organisation

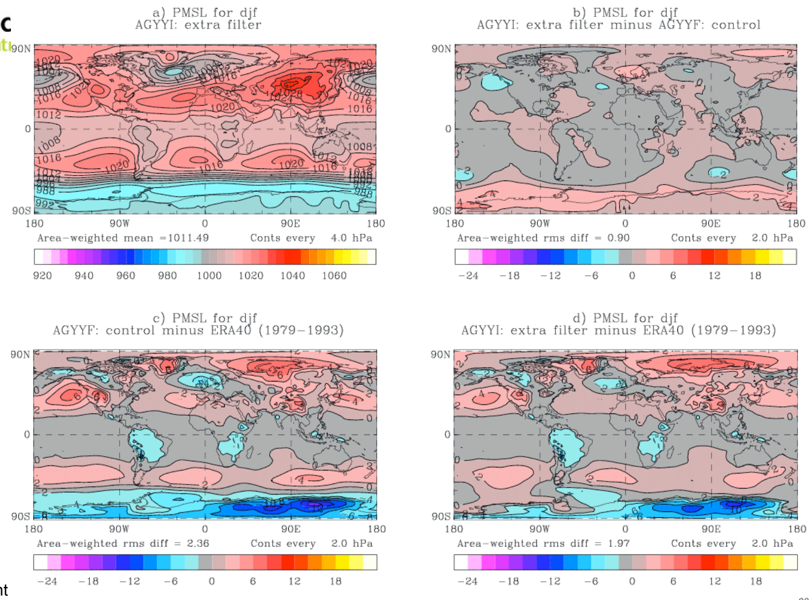


General Open Source Project   Met Office

Verification and Validation

- Desk checking
  - Informal unit test, occasional (rare) use of debuggers
- Science Review and Code Review
  - Science review by project managers
  - Code review by designated code owners
- Continuous testing as Science Experiments
- Automated test harness on main trunk
- Bit Reproducibility as strong constraint

Continuous Integration Testing



e.g. Testing HadGEM3

	N96L38			N96L70			N144L38
atm	coupled	test		atm	coupled	test	atm
afsob			(Control run - 20 years)				aftmc
I					biogenic & sulphur off		I
afsod			(package A convection - excluding convection diagnosis)				I
I		afsoe	(minus sulphor, biogenic aerosols and river routing)	afsof			I
afsoh			(0.5 CMT)	I			I
afsoj			(new boundary layer solver)	I			I
afsom			(revised shallow parcel perturbation)	I			I
afson			(improved convection diagnosis)	I			I
afsor			(retuned CCA decay)	afsoq			I
afsos			(Include L70 mods in L38)	I			I
I			(Make L70 stable)	afsot			I
agbxc	afyge		(making up-to-date with HadGEM2-A frozen (agaoa))	agbxb			agbxf
I		agcpa	(PC2:66 test)	I			I
I	aggtn		(removing new boundary layer solver from coupled model)	I	afygh		I
I	afygg		(adding PC2:66 into coupled model)	I			I
agcpb			(1 to 0.25 CMT)	agbxg			I
I			(PC2:66e test)	I	afygm	agcph	I
agbxh			(new soil ancil - MODIS albedo)	I			I
I		aghoc	(orca1 land/sea mask & new basic ancillaries)	I			I
I		aghom	(as aghoc+AMIP sst, sice+static veg)	I			I
I		aghoo	(as aghom+new seasonal veg func+new dust)	I			I
agbxi			(upgrade CCRad)	agbxj			I
I			(L85 version of agbxj)	I		agbgm	I
I			(test of shallow adaptive detr)	I		agbgj	I
I				I		I	I
I			(test of coastal tiling ("buddy") fix)	I		agbgk	I
I			(30 BL levels in L70)	agbxn			I
I					biogenic & sulphur on		I
agbxk			(improved SO2+O3 reaction)	agbxr	afygp		I
I		agbxl	(organic carbon from fossil fuels)	I			I
agbxm			(PC2:66)	agbxs	afygq		I
I		agbxo	(No coastal tiling)	I			I
I		agbxq	(Manoj's Coastal drag fix)	I			I
I		agbgi	(test of shallow adaptive detr)	I			I
agbxt			(Include agkfc's improvements)	I			I
I		agcpo	(Soil thermal conductivity improvements)	I			I
aguka	afygr		(Coastal ("buddy"), sh adap & thermal con)	agukb			I
I			(L85 version)	I		agbgn	I
I		ahbma	(L60 version but with 4x erosion)	I			I
I		ahbmb	(L60 version)	I			I
I			(L63 version)	I			I

'Virtual' lab notebook

EditWYSIWYGAttachPrintable
r3 - 27 May 2008 - 12:36:46 - DanCopseyYou are here: TWiki > Main Web > HadGEM3Project > HadGEM3Evolution > HadGEM3agyyi

HadGEM3-A agyyi

Author: Dan Copsey

Owner: Dan Copsey

Predecessor: HadGEM3-A agyyf

Description: Using a larger polar filtered zone to try and stop blow ups at 87S.

Current status: Complete

Model Start/end times: From 1 Sep 1978 to 30 Dec 1988 (10 years and 4 months)

Details of changes with respect to predecessor:

Larger polar filtered zone (Terry Davies):

- Sec by sec/sec 13 diffusion/combi/polar filter start latitude: 87 --> 85

Notes

Online validation notes and data

10 year short validation note comparing this job with its predecessor is here: http://www-hc/~hadco/valid2/agyyi_v_agyyf/short.html

Results of this model verses its predecessor

- Increased sea level pressure over Antarctica in DJF, improving the model. Associated with this is reduced surface wind stress in the southern ocean, also improving the model.
- Increased 200 hPa air temperature over Antarctica in DJF, improving the model.

Back to HadGEM3Evolution

-- Main.DanCopsey - 15 May 2008

Edit | WYSIWYG | Attach | Printable | Raw View | Backlinks: Web, All Webs | History: r3 < r2 < r1 | More topic actions

© Crown Copyright

Model inter-comparison

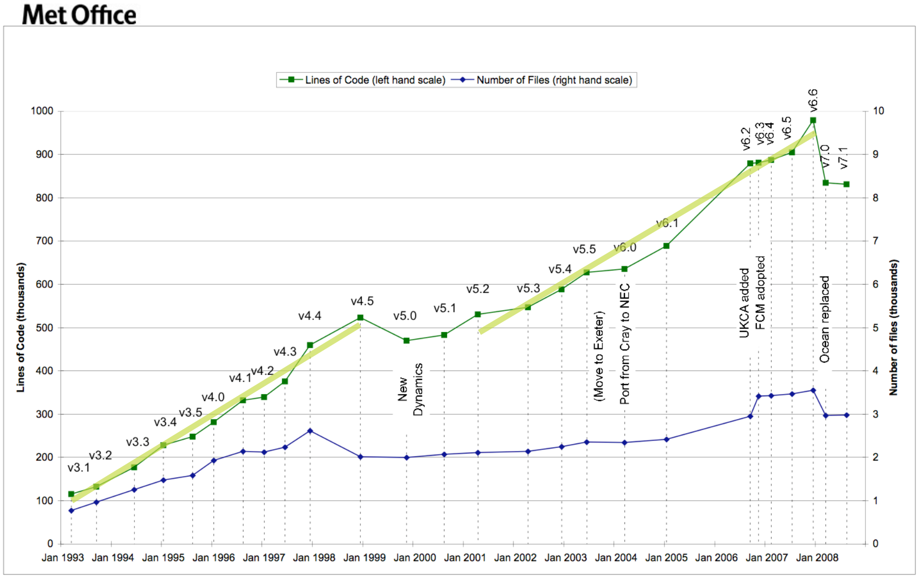
- Informal model comparisons
  - Used for diagnosing modeling errors
- Model Inter-comparison Projects (MIPs)
- Model Ensembles
  - Models from different labs on a common scenario
  - Variants of a single model to compare schemes
  - Perturbed physics ensembles
  - Single model with varied initial conditions

The Good News

What works, and why it works

Growth (in functionality?) isn't slowing

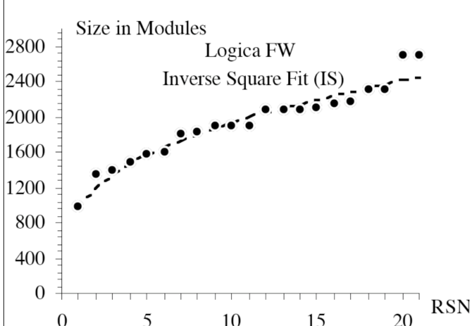
Met Office



Comparators

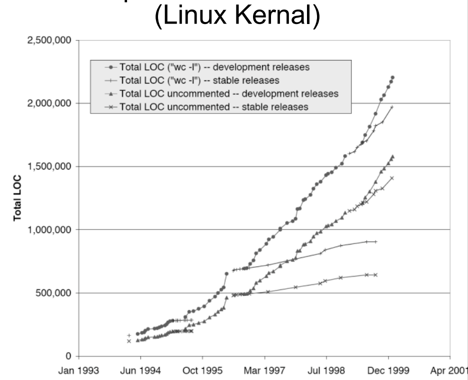
Met Office
Hadley Centre

Financial Software System (Logica)



(Source: Lehman et al, 2000)

Open Source Software (Linux Kernal)



(Source: Godfrey & Tu, 2000)

Met Office
Hadley Centre

Software “defect rates”

Some comparisons:

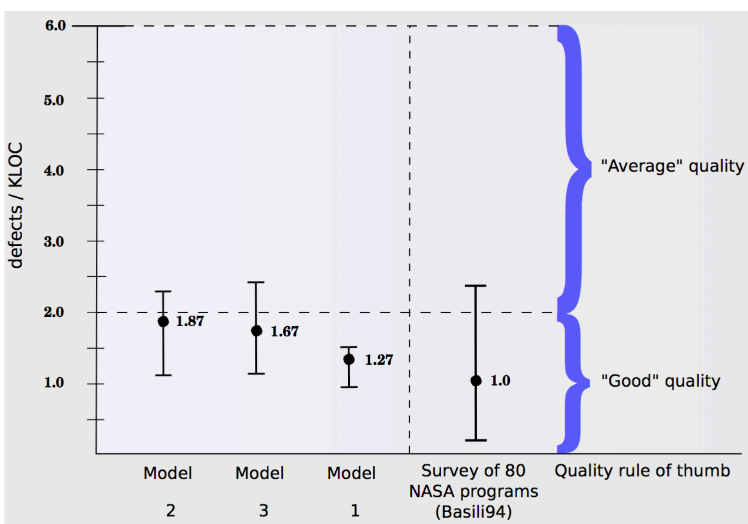
- Worst military systems: 55 faults/KLOC
- Best military systems: 5 faults/KLOC
- “Extreme Programming”: 1.4 faults/KLOC
- Apache (open source): 0.5 faults/KLOC
- NASA Space shuttle: 0.1 failures/KLOC

Unified Model:

- avg of 24 “bug fixes” per release
- avg of 50,000 lines edited per release
- ⇒ 2 defects / KLOC make it through to released code
- ⇒ expected defect density in current version:
  24 / 830,000 ≈ 0.03 faults/KLOC

Met Office
Hadley Centre

A more detailed study



Few Defects Post-release

Met Office
Hadley Centre

- Obvious errors:
  - Model won't compile / won't run
  - Model crashes during a run
  - Model runs, but variables drift out of tolerance
  - Runs don't bit-compare (when they should)
- Subtle errors (model runs appear "valid"):
  - Model does not simulate the physical processes as intended (e.g. some equations / parameters not correct)
  - The right results for the "wrong reasons" (e.g. over-tuning)
  - Expected improvement not achieved

Met Office
Hadley Centre

Critical Success Factors

Which aspects of practice contribute particularly to the successes

Key Success Factors

- Highly tailored software development process
  (software development is "doing science")
- Single Site Development
- Software developers are domain experts
- Shared ownership and commitment to quality
- Openness ("Many eyes" validation)
- Benchmarking (e.g MIPS & ensembles)
- Unconstrained Release Schedule

Highly Adapted Processes

"Agile" vs "Sturdy"

Iterative ⟷ Planned

Small increments ⟷ Analysis before design

Adaptive planning ⟷ Prescriptive planning

Embrace change ⟷ Control change

Innovation and exploration ⟷ High ceremony

Trendy ⟷ Traditional

Highly fluid ⟷ Upfront design / architecture

Feedback driven ⟷ Negotiated requirements

Individuals and Interactions ⟷ Processes and Tools

Human communication ⟷ Documentation

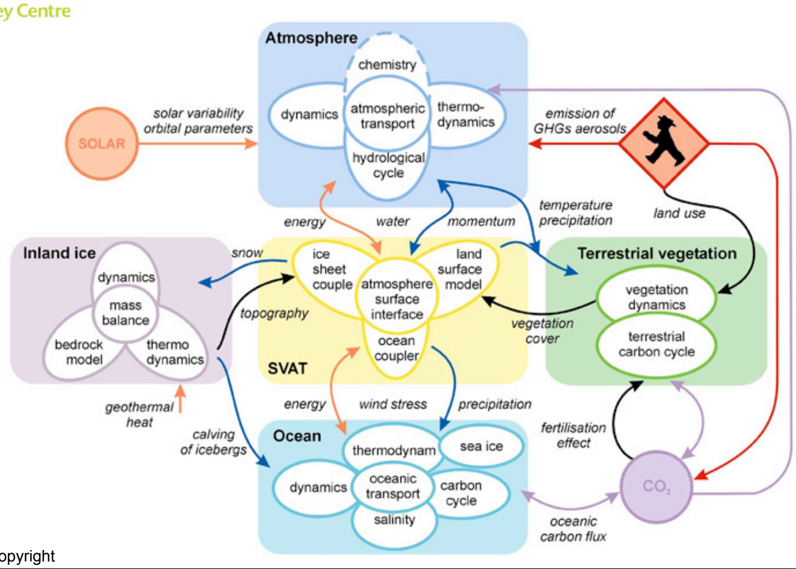
Small teams ⟷ Large teams

Use of Agile practices:

- ✓ Collective Ownership
- ✓ Configuration Management
- ✓ Continuous Integration
- ✓ Feature-driven devl.
- ~ Frequent small releases
- ✓ Onsite customer
- ~ Organization-wide process
- ~ Organizational training
- ✗ Pair programming
- ✗ Planning game
- ✓ Peer reviews
- ~ Process & product quality assurance
- ✓ Project monitoring & control
- ✓ Project planning
- ✗ Refactoring
- ? Requirements management
- ~ Retrospective
- ✓ Risk Management
- ✓ Simple design
- ✓ Tacit knowledge
- ✗ Test-driven development

Met Office
Hadley Centre

Shared Conceptual Architecture



Met Office
Hadley Centre

Comparison with Open Source Projects

- Release schedule not driven by commercial pressures
- Developers are domain experts
- Core group of code owners control trunk
- Community operates as meritocracy
- Developers have “day jobs” (as scientists!)
- V&V based on extensive use by developers themselves

Challenges and Opportunities

What needs fixing?

Challenges

- Improve coordination across code branches
- Coordination with external users
- Multi-site development
- Make model configurations easier to define and validate
- Improve access to model result datasets

Coordination and Shared Understanding

- Coordinating the teams is a major challenge
  - Keeping your branch up to date
  - Knowing what changes are happening elsewhere
  - Configuration dependencies and hand-edits
- Heavy reliance on informal communication
  - problems solved by “knowing who to talk to”
- External users using “old” versions
- Other development sites use different processes

Summary & Lessons Learned

Key insights for Software Engineering and for Climate Science

Hadley study: initial questions

- How do scientists assess “correctness” of the code?
  - ~~“correctness”~~ → “model skill”
  - continuous re-assessment of modeling tradeoffs
- How do they ensure experiments can be reproduced?
  - Releases are frozen (no bug-fixes), configs archived
  - Bit reproducibility across architectures and versions
- How do they maintain a shared understanding?
  - Single site, agile practices, many informal comms channels
- How do they prioritize model developments?
  - Organic, bottom-up, code owners provide longer view
- How do they detect/prevent errors in the software?
  - Continuous integration testing, model intercomparisons,…

Future Work

Where next?

- Comparison with other Climate Modeling Centres:
  - e.g. CCCma (very small team)
  - e.g. NCAR (community model)
  - e.g. MPI-M, GFDL, IPSL, …
- Compare Validation processes with other scientific models
  - e.g. economics models used in climate policy
  - e.g. other environmental science models