

Collective Ownership & Coding Standards

Steve Hayes
Cogent Consulting

Ownership

Who owns what code?

Traditionally a developer owns their
subsystems

No one touches someone else's code

No one would want to - it all looks
different, and faintly disgusting

And no one understands the intricacies of
the “other” code

So....

**When you want something changed, you
have to wait for the owner to be available**

Hope they're not on vacation!

This leads to scheduling problems

and really assumes business requirements
come in fairly predictable patterns

when the business would often like to be
responsive, adaptive and flexible

You also end up shipping your
organisational structure

(and not just in software - see “The Innovator’s Dilemma”)

This isn't all bad

You get consistency of vision for each
subsystem

You have a gate for changes to each subsystem, for better or for worse

Collective Ownership

Anyone can change anything at anytime...

provided they comply with all our
conventions

acceptance tests

coding standards

unit tests

simple design

relentless refactoring

pair programming

test driven development

metaphor

passes continuous integration

Everyone should change things that are
wrong

some changes are required simply because
the context has changed

many are also opportunities for education

we should be able to get anyone working
on anything

but we don't expect everyone to be of
equal ability in all areas

In the end, we succeed or fail as a team

Coding Standards

“Reduce the cognitive friction, Mr Scott!”

“Aye, aye, Captain”

If anyone can work anywhere...

We'd prefer all the code to look the same

Not just layout, but other things as well

Naming standards, coding idioms, design patterns, ...

... and code metrics!

- Method length
- NPath complexity
- Number of Exceptions Thrown
- Cyclomatic Complexity
-

Enforcing standards

Standards should be a safety net, not a
trigger for punishments

“Fear is the mind killer”

Fear makes us move slowly, and we want
to go as fast as we can

Agile methods acknowledge that software
is developed by people, who will make
mistakes

but tries to make the cost of a mistake as
low as possible

often through automation

We use automation to enforce standards
where ever we can

but some things demand aesthetic
judgement

Discussion