The effect of complexity and value on architecture planning in agile software development

Michael Waterman, James Noble, George Allan
Victoria University of Wellington, New Zealand
XP 2013, Vienna
6 June 2013

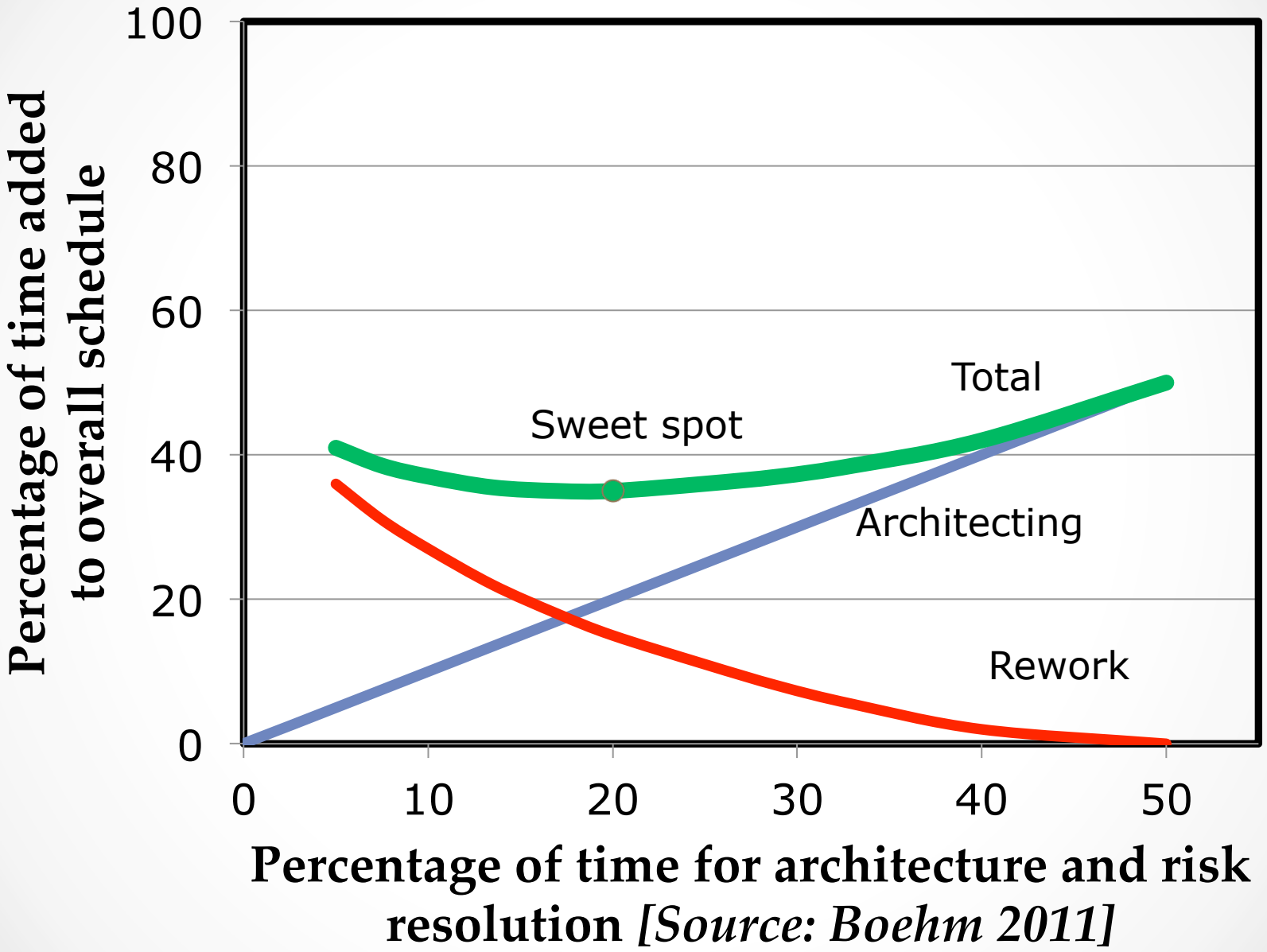
Architecture and agility

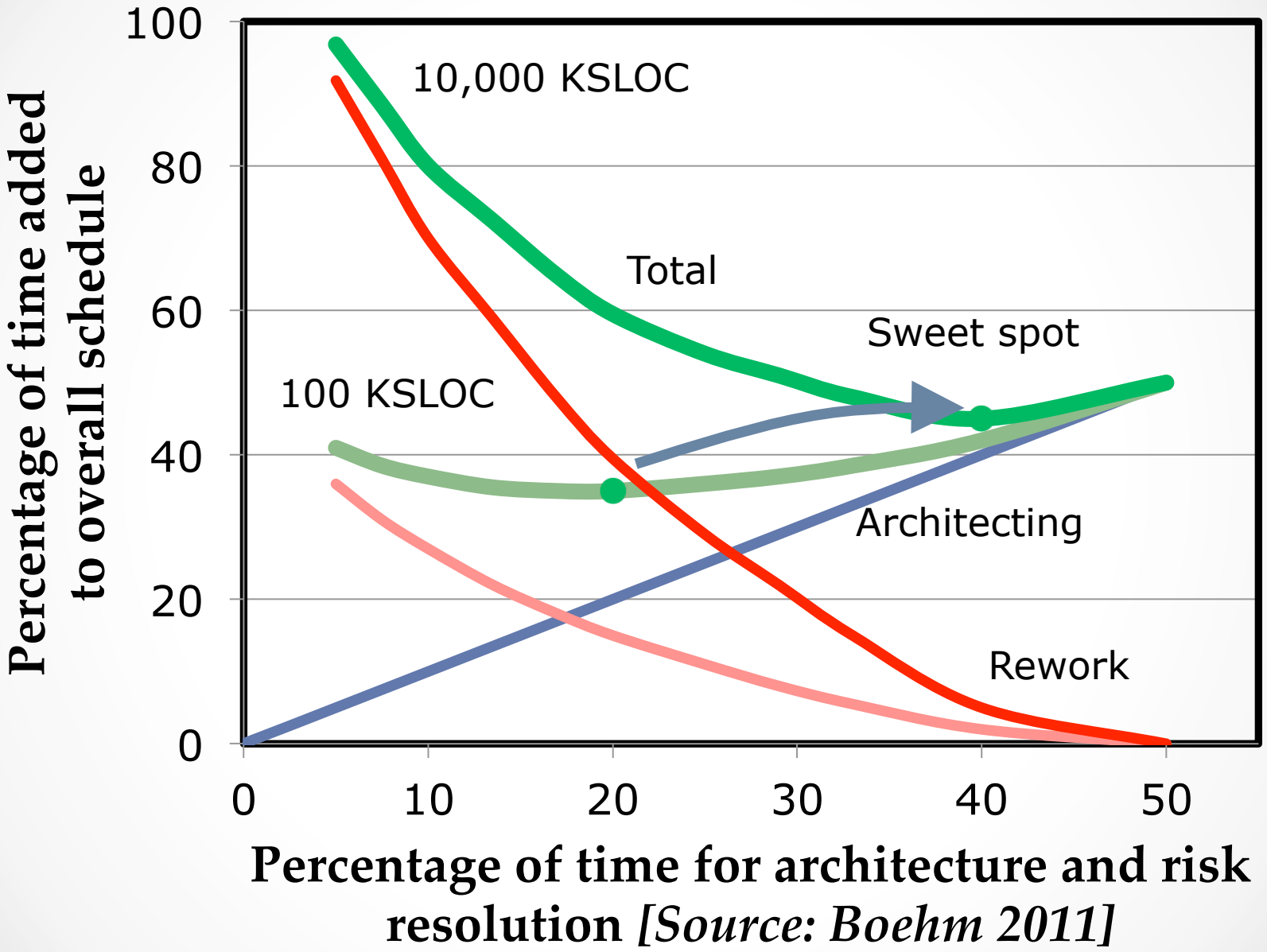
too much

- Agile development discourages planning ahead
- But architecture is about planning ahead…
- So how do teams strike a balance between architecture and being agile?

How much architecture?

- How much up-front architecture planning do teams do?
- What affects how much they do?
- What is the relationship between:
  - Complexity and size?
  - Value and cost?


Percentage of time added to overall schedule
100
80
60
40
20
0
Total
Sweet spot
Architecting
Rework
0
10
20
30
40
50
Percentage of time for architecture and risk resolution [Source: Boehm 2011]


Percentage of time added to overall schedule
100
80
60
40
20
0
10,000 KSLOC
Total
Sweet spot
100 KSLOC
Architecting
Rework
0
10
20
30
40
50
Percentage of time for architecture and risk resolution [Source: Boehm 2011]

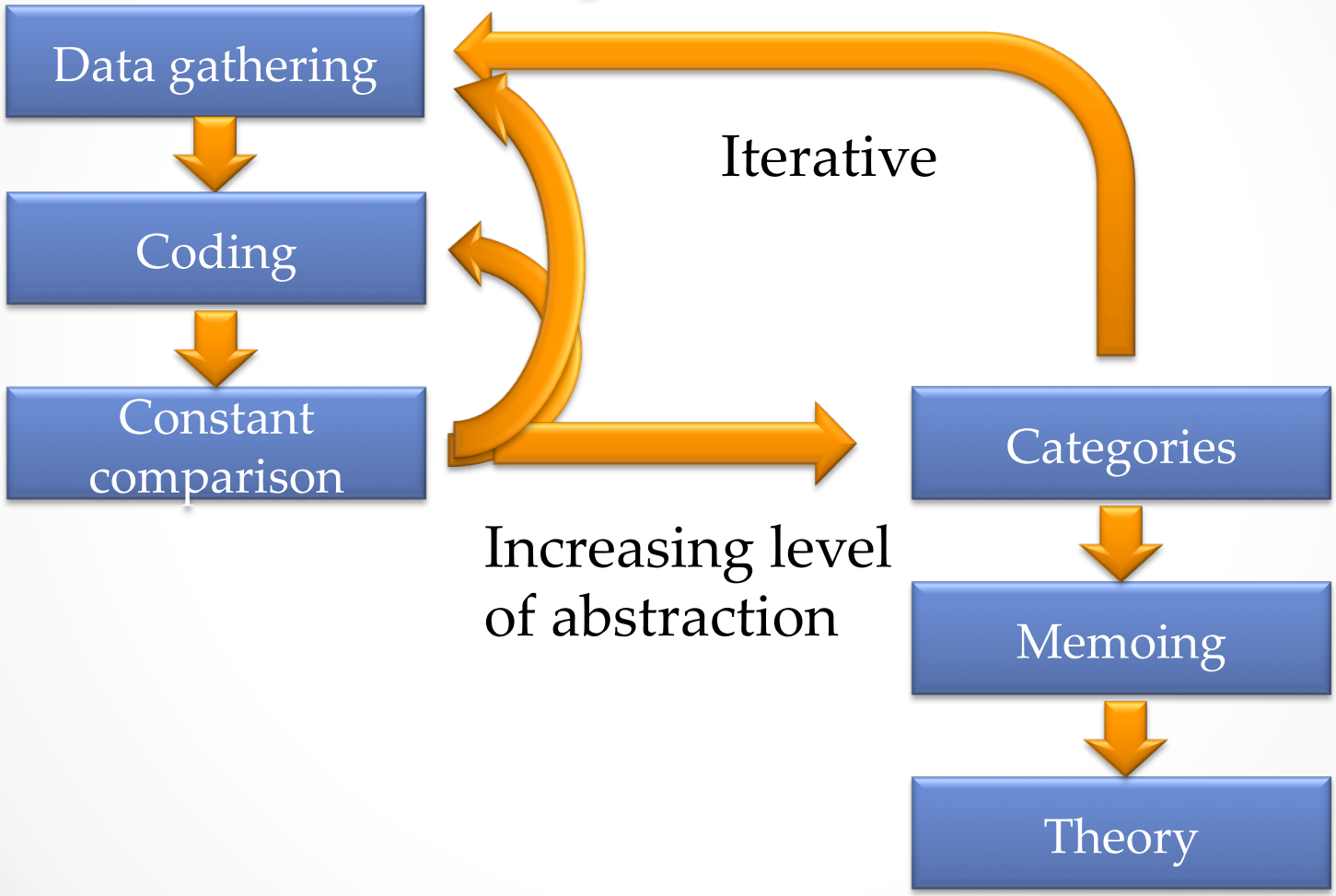
Architecture design is about people

- Creative and social activity:
  - Based more on knowledge, understanding, background, experience…
  - …and less on processes, methods, frameworks
- No single correct solution

Grounded theory

- This research is qualitative
- Inductive (theory follows research)
- Very little existing research
- Therefore, **grounded theory method**
- Methodical and rigorous

Steps of grounded theory method



Work done

- 36 interview participants (and documentation)
- Variety of roles
- Variety of domains
- Variety of project and system types

Modern development frameworks

- Standard solutions to common problems
- Preferred or "precooked" architectures
- **Reduce architectural effort**
  - **Easier to make decisions**
  - **Easier to change decisions**

Source: Waterman, Noble and Allan 2012

Modern development frameworks

- Standard solutions to common
- "Go with what's proven, go with what works. [...] We don't have architectural discussions – we don't need to – the problem's [already] been solved." (P27, CEO/agile coach)

The effect of complexity on up-front effort [1]

- Complexity is typically the result of demanding requirements
- Complexity pushes the limits of what development frameworks can do
  - Not common problems so cannot provide standard solutions

The effect of complexity on up-front effort [2]

- Complex systems may:
  - have bespoke components,

The effect of complexity on up-front effort [2]

- Complex systems may:
  - have bespoke components,
  - have multiple technologies,

The effect of complexity on up-front effort [2]

- Complex systems may:

"If it's really horribly complex and you've got to request all sorts of bits of infra-structure from all over the show to get it to work then it definitely slows down iteration zero." (P29, development manager)

The effect of complexity on up-front effort [2]

- Complex systems may:
  - have bespoke components,
  - have multiple technologies,
  - involve legacy systems,

The effect of complexity on up-front effort [2]

- Complex systems may:
  - have bespoke components,
  - have multiple technologies,
  - involve legacy systems,
  - have many integration points.

The effect of complexity on up-front effort [2]

- Complex systems may:

"Today's systems […] have a lot more interfaces to external systems than older systems which are typically standalone. They have a lot higher level of complexity for the same sized system." (P14, solutions architect)

The effect of complexity on up-front effort [2]

- Complex systems may:
  - have bespoke components,
  - have multiple technologies,
  - involve legacy systems,
  - have many integration points.
- Complexity leads to additional design
- **Complexity is an important determinant of how much up-front planning teams do**

The relationship between complexity and size

- System size and complexity are related
- But… a large system entirely within the boundaries of the framework will have less complexity
- And a small system may have demanding requirements and require a lot of planning.

The effect of size on up-front effort

- Size is not as important as complexity when determining the amount of up-front planning

The effect of size on up-front effort

- Size is not as important as complexity

"If we have size that just extends the time, it's of little concern to us. It's just a slightly larger backlog, management overhead." (P32, development director)

The effect of size on up-front effort

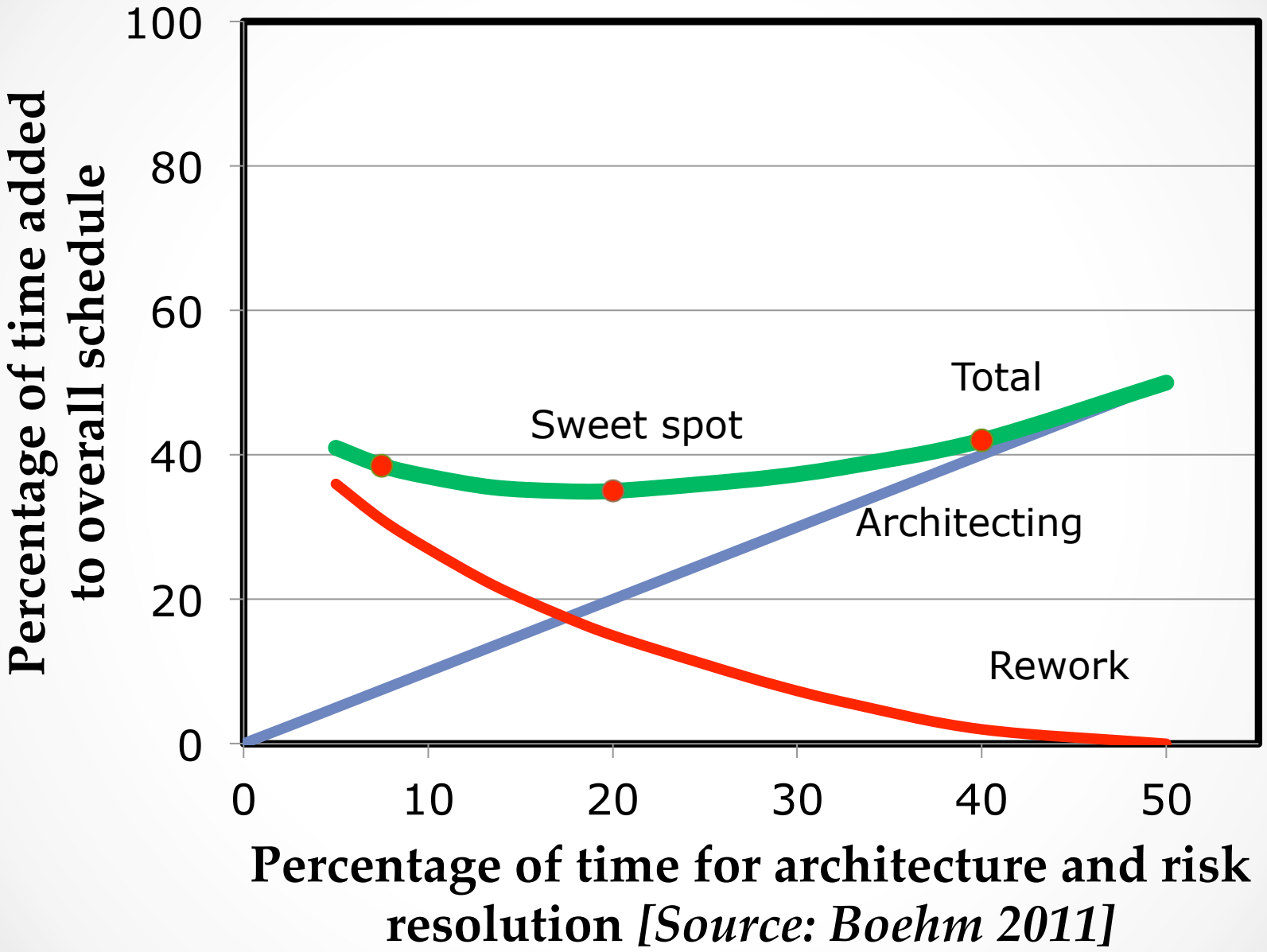
- Size is not as important as complexity when determining the amount of up-front planning
- Non-demanding requirements + appropriate development framework:
  - reduced complexity
  - reduced up-front architecture effort
  - Increased agility

The effect of value and cost on up-front effort

- Agile teams do not always aim for sweet spot
- Less planning means early product release and early revenue:
  - Cash flow will pay for later architectural rework
  - Value is more important than cost
- Early feedback

The effect of value and cost on up-front effort

- Agile teams do not always aim for
- "Designing for a million users is a problem you can have once you've got a million users and you've got a million users worth of revenue" (P27, CEO/Agile coach)
- Early feedb[illegible]k


Percentage of time added to overall schedule
100
80
60
40
20
0
Total
Sweet spot
Architecting
Rework
0
10
20
30
40
50
Percentage of time for architecture and risk resolution [Source: Boehm 2011]

Conclusion

- Sweet spot graph is based on data that predates agile
- **Complexity** is a more important driver of up-front effort than project **size**
- Many agile teams are driven by **maximising value** rather than **minimising cost**
  - Particularly those who can release early!

Next steps

- Validation is an important part of grounded theory
- Feedback from agile practitioners!
- Complete the theory

Michael Waterman

Victoria University of Wellington, New Zealand

Michael.Waterman@ecs.vuw.ac.nz
http://ecs.victoria.ac.nz/Main/GradMichaelWaterman
http://nz.linkedin.com/in/michaelwaterman
@waterman_m