What Makes A Great Software Engineer?

Based on: Paul Luo Li, Andrew J. Ko, and Jiamin Zhu. 2015. What makes a great software engineer?. In *Proceedings of the 37th International Conference on Software Engineering - Volume 1* (ICSE '15), Vol. 1. IEEE Press, Piscataway, NJ, USA, 700-710.

Paul Luo Li[*+], Andrew J. Ko[+]

UNIVERSITY of WASHINGTON[+]

Microsoft[*]

ACM Highlights

- **Learning Center** tools for professional development: http://learning.acm.org
  - 4,000+ trusted technical books and videos by **O'Reilly, Morgan Kaufmann**, etc.
  - 1,000+ courses, virtual labs, test preps, live mentoring for software professionals covering programming, data management, cybersecurity, networking, project management, more
  - Training toward top vendor certifications (CEH, Cisco, CISSP, CompTIA, ITIL, PMI, etc.)
  - Learning Webinars from thought leaders and top practitioner
  - Podcast interviews with innovators, entrepreneurs, and award winners

- Popular publications:
  - Flagship *Communications of the ACM* ***(CACM)*** magazine: http://cacm.acm.org/
  - ***ACM Queue*** magazine for practitioners: http://queue.acm.org/

- **ACM Digital Library**, the world's most comprehensive database of computing literature: http://dl.acm.org.

- International **conferences** that draw leading experts on a broad spectrum of computing topics: http://www.acm.org/conferences.

- Prestigious awards, including the **ACM A.M. Turing** and Infosys: http://awards.acm.org/

- And much more… http://www.acm.org.

What Makes A Great Software Engineer?

Based on: Paul Luo Li, Andrew J. Ko, and Jiamin Zhu. 2015. What makes a great software engineer?. In *Proceedings of the 37th International Conference on Software Engineering - Volume 1* (ICSE '15), Vol. 1. IEEE Press, Piscataway, NJ, USA, 700-710.

Paul Luo Li[*+], Andrew J. Ko[+]

UNIVERSITY of WASHINGTON[+]

Microsoft[*]

> *“At the end of the day, to make a change [to software]… it takes a dev—a butt in a seat—to type [Source Depot] commit ”*
>
> -Partner Dev Manager, Windows

Essential To Know *What* Makes Great Engineers And *Why* Those Attributes Matter

Educators (like University of Washington): to train great engineers

Employers (like Microsoft): to hire and retain great engineers

Young engineers: to become great.

Knowledge About Software Engineering Expertise: Incomplete, Indirect, Or Abstract

Productive: finishing more tasks, faster, or with fewer mistakes [Sackman et al. ‘68] [Gugerty&Olson ‘86]

Collaborates effectively in teams; makes meaningful contributions [Kelly ‘99] [Begel&Simon ‘06] [Hewner&Guzdial ‘10]

Write/edit code; communicates with other engineers; acquires understanding [LaToza ‘06] [Ko ‘06]

Technical knowledge: Programming Fundamentals, Software Design, Software Modeling and Analysis, Software Verification and Validation, Project Management [ACM/IEEE ‘13]

(From industry reports): open-minded and desires to learn [McConnell ’04] [Bryant ‘13]

Knowledge About Software Engineering Expertise: Incomplete, Indirect, Or Abstract

Productive: finishing more tasks, faster, or with fewer mistakes [Sackman et al. ‘68] [Gugerty&Olson ‘86]

Collaborates effectively in teams; makes meaningful contributions [Kelly ‘99] [Begel&Simon ‘06] [Hewner&Guzdial ‘10]

Write/edit code; communicates with other engineers; acquires understanding [LaToza ‘06] [Ko ‘06]

Technical knowledge: Programming Fundamentals, Software Design, Software Modeling and Analysis, Software Verification and Validation, Project Management [ACM/IEEE ‘13]

(From industry reports): open-minded and desires to learn [McConnell ’04] [Bryant ‘13]

Knowledge About Software Engineering Expertise: Incomplete, Indirect, Or Abstract

Productive: finishing more tasks, faster, or with fewer mistakes [Sackman et al. ‘68] [Gugerty&Olson ‘86]

Collaborates effectively in teams; makes meaningful contributions [Kelly ‘99] [Begel&Simon ‘06] [Hewner&Guzdial ‘10]

Write/edit code; communicates with other engineers; acquires understanding [LaToza ‘06] [Ko ‘06]

Technical knowledge: Programming Fundamentals, Software Design, Software Modeling and Analysis, Software Verification and Validation, Project Management [ACM/IEEE ‘13]

(From industry reports): open-minded and desires to learn [McConnell ’04] [Bryant ‘13]

Knowledge About Software Engineering Expertise: Incomplete, Indirect, Or Abstract

Productive: finishing more tasks, faster, or with fewer mistakes [Sackman et al. ‘68] [Gugerty&Olson ‘86]

Collaborates effectively in teams; makes meaningful contributions [Kelly ‘99] [Begel&Simon ‘06] [Hewner&Guzdial ‘10]

Write/edit code; communicates with other engineers; acquires understanding [LaToza ‘06] [Ko ‘06]

Technical knowledge: Programming Fundamentals, Software Design, Software Modeling and Analysis, Software Verification and Validation, Project Management [ACM/IEEE ‘13]

(From industry reports): open-minded and desires to learn [McConnell ’04] [Bryant ‘13]

Knowledge About Software Engineering Expertise: Incomplete, Indirect, Or Abstract

Productive: finishing more tasks, faster, or with fewer mistakes [Sackman et al. ‘68] [Gugerty&Olson ‘86]

Collaborates effectively in teams; makes meaningful contributions [Kelly ‘99] [Begel&Simon ‘06] [Hewner&Guzdial ‘10]

Write/edit code; communicates with other engineers; acquires understanding [LaToza ‘06] [Ko ‘06]

Technical knowledge: Programming Fundamentals, Software Design, Software Modeling and Analysis, Software Verification and Validation, Project Management [ACM/IEEE ‘13]

(From industry reports): open-minded and desires to learn [McConnell ’04] [Bryant ‘13]

The Gap: Incomplete, Indirect, And Abstract Knowledge About Software Engineering Expertise

- Few studies examine software engineering expertise directly
- Lack holistic view of software engineering expertise
- Little rigorous understanding
- Missing definitions and explanations

Sought Knowledge From Expert Software Engineers At Microsoft

Microsoft: one of the largest, most successful, and most *diverse* software development organizations

Ad Platform, Bing, Corp Dev (e.g. Security), Dynamics, Office, Phone, Server & Tools, Windows, Windows Services, Xbox, Skype, etc.

Talented and experienced software engineers: at least Software Development Engineer Level 2 (3+ years of experience), specifically targeted *very experienced* engineers (15+ years of experience) :

Technical Fellow, Architect, Partner Dev Manager, Partner Dev Lead, Principal Dev Lead, Senior Dev Manager, Principal SDE

Analyzed 60+ Hours Of Interviews, 388,000+ Words Of Transcripts

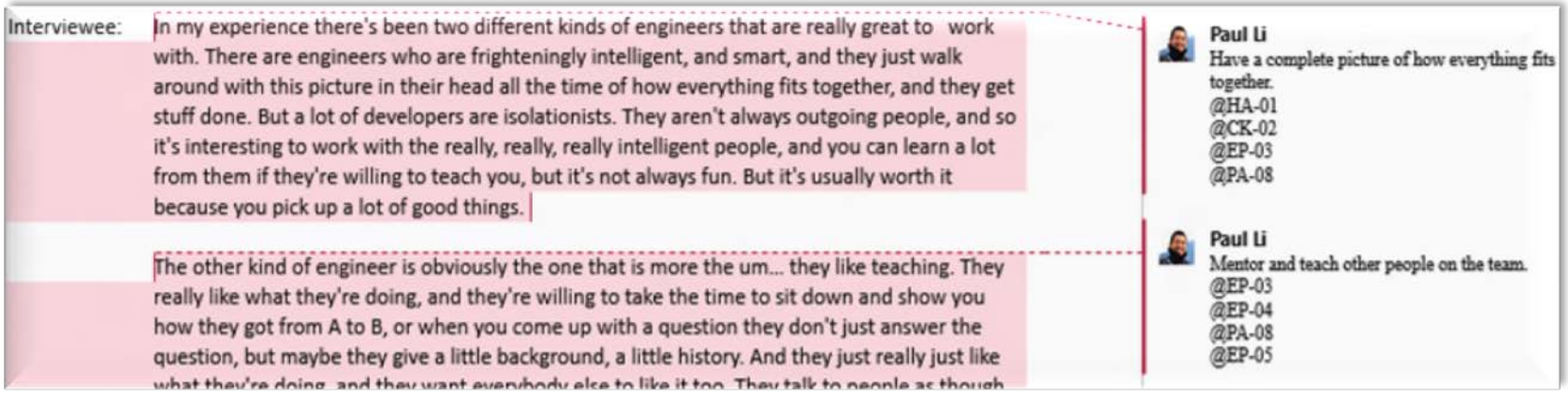
Hour-long interviews, with drill-ins:

What were attributes of great engineers they've worked?

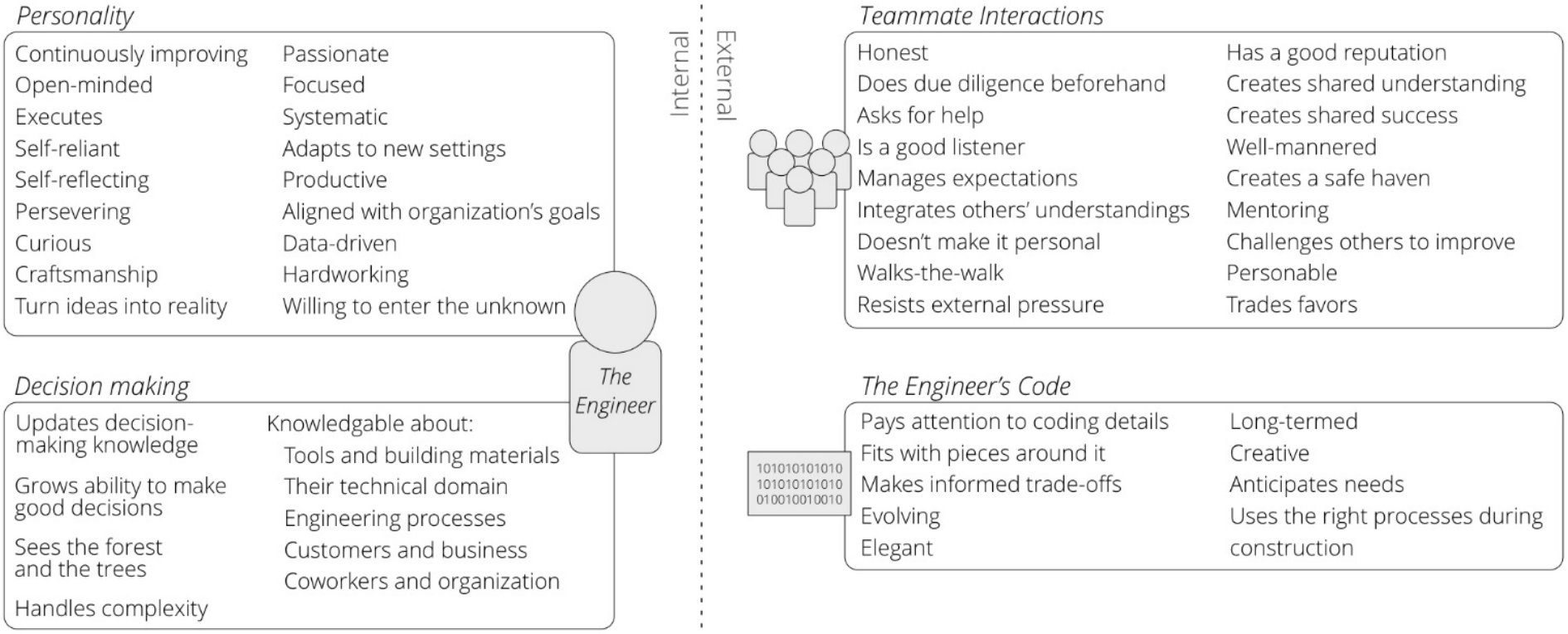
Why were those attributes were important?

Transcribed all interviews, then read them in detail, classifying sentiments

Validated by a Senior Software Engineer



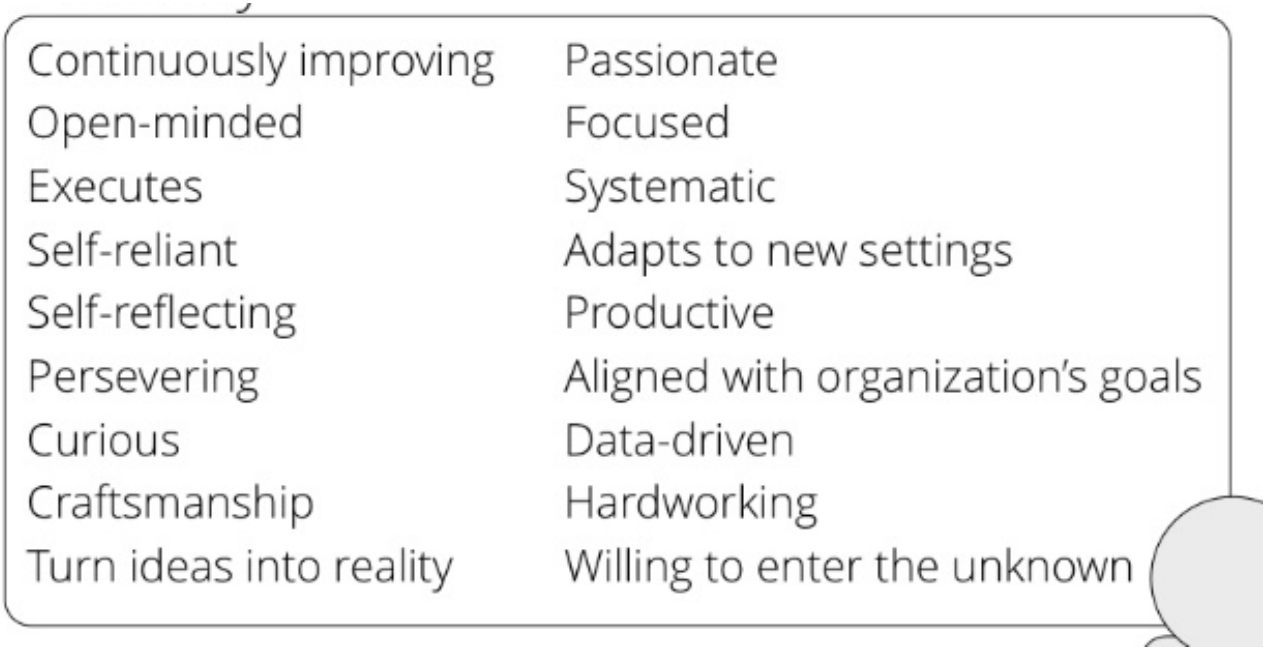
53 Attributes Of Great Software Engineers, Consisting Of Internal And External Attributes



Internal Personality Traits

That is something that can't be taught... they have just an inner desire to succeed, and I don't know why. It's not necessarily for the money, it's not necessarily for the recognition. It's just that whatever it is they do, they want to do it extremely well… I've seen a lot of smart people that have none of these characteristics.....

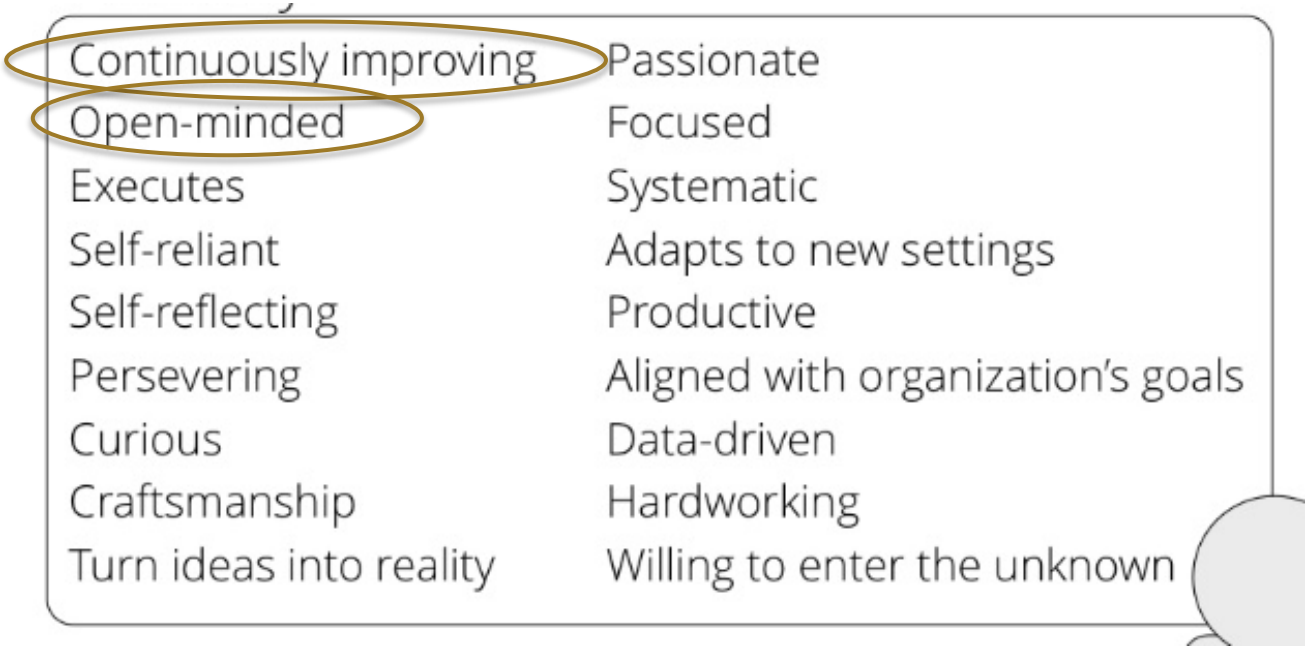
-Principal Dev Lead, Windows



Internal Personality Traits

That is something that can't be taught... they have just an inner desire to succeed, and I don't know why. It's not necessarily for the money, it's not necessarily for the recognition. It's just that whatever it is they do, they want to do it extremely well… I've seen a lot of smart people that have none of these characteristics.....

-Principal Dev Lead, Windows



Continuously Improving…

Computer technology, compared to other sciences or technology, it's pretty young. Every year there's some new technology, new ideas. If you are only satisfied with things you already learned, then you probably find out in a few years, you're out of date… good software engineer [sic], he keep investigate, investment. [sic]

-SDE2, Corp Dev

Not satisfied with the status quo and constantly looking to improve themselves, their product, and/or their surroundings.

…Becoming And Continuing Being Great

Engineers do not start out being great: young engineers need to learn and improve to become great

The software field moves rapidly: great engineers need to keep on learning to continue to be great

Open Minded…

No matter how much you know, the software industry is so large… there's so many other areas… If that person has something to say that hadn't occurred to me, I'll stop everything and say, ok, explain this. What did you see, that I didn't see?

-Senior SDE, Office

Willing to judiciously let new information change how they think, not taking the current understanding as gospel

…Avoiding Thinking You Know Everything

Outcomes (e.g. user reactions or commercial success) are difficult to predict: be open to changing your understanding

Software can be large, complex, and changing: be willing to consider understanding ideas of others

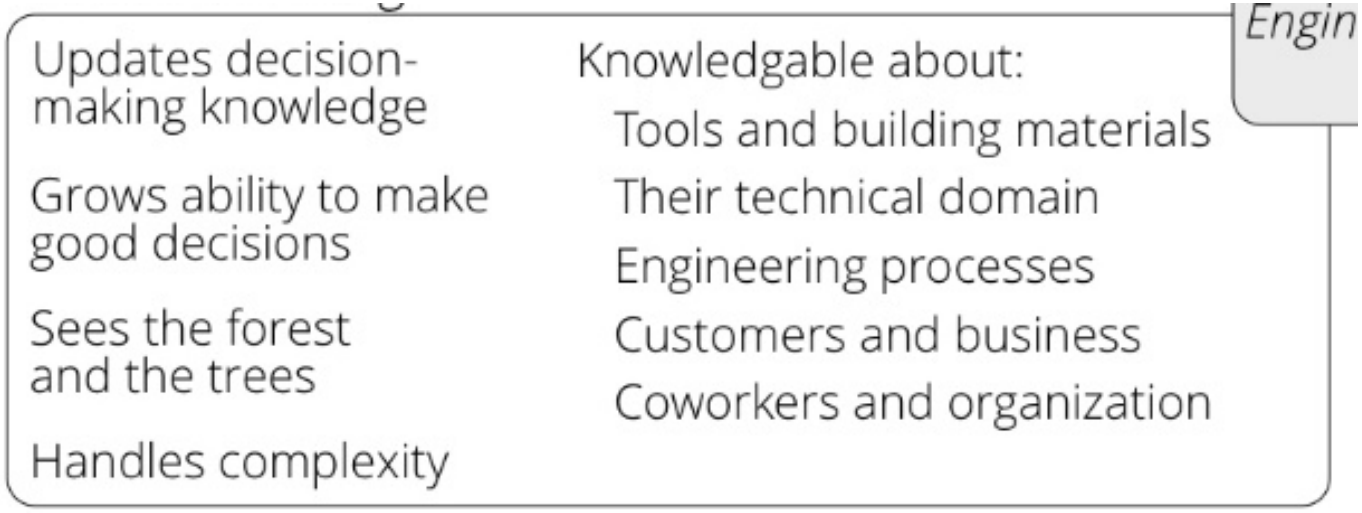
Take Away #1:

The *ability to learn* is more important than any individual technical skill

Making Good Decisions

How do we make, what I often call, 'robust decisions'? What's a decision we could make, depending on this range of potential outcomes, which we can't foresee?... if we can make a decision that is viable, whether A or B happens, then we don't have to fight about A or B right now.

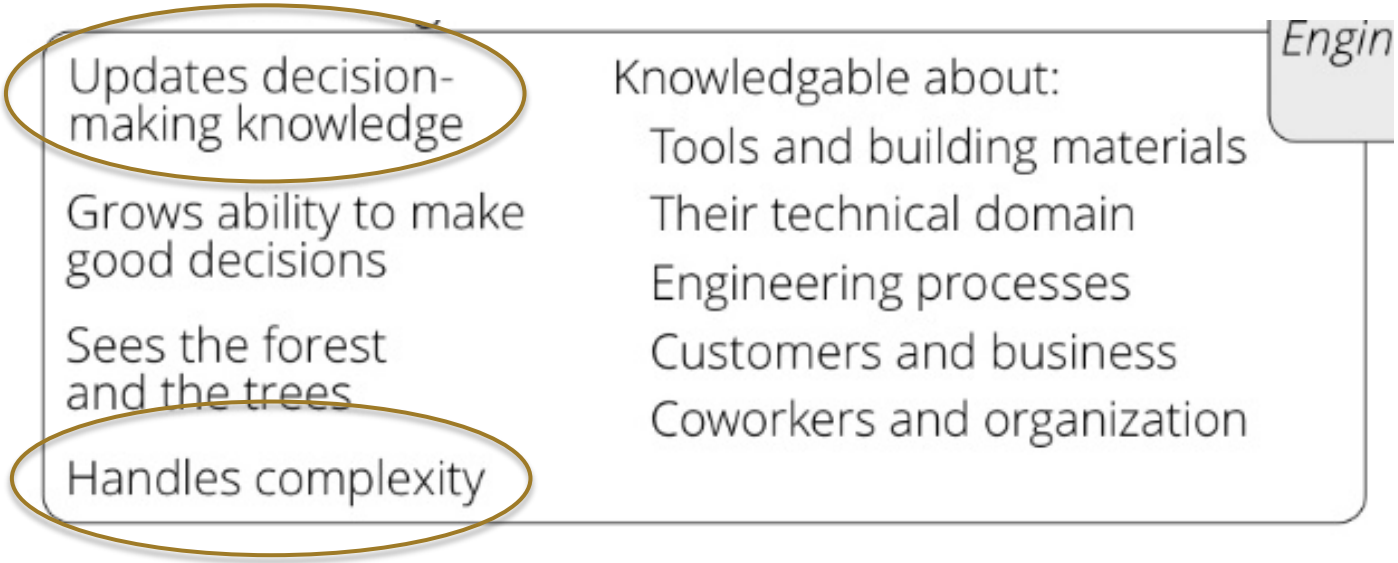
-Technical Fellow (division removed to preserve anonymity)



Making Good Decisions

How do we make, what I often call, 'robust decisions'? What's a decision we could make, depending on this range of potential outcomes, which we can't foresee?... if we can make a decision that is viable, whether A or B happens, then we don't have to fight about A or B right now.

-Technical Fellow (division removed to preserve anonymity)



Updating Decision Making Knowledge…

Sometimes what used to be a second or third order effect comes to dominate. So way back in the day, if you wanted to performance optimize something you counted instructions. Processors got faster and faster, but memory references didn't. There became a day when it made more sense to count memory references than it did to count instructions. Unless you're conscious of when those things will intersect, you'll be on the wrong side of history and be frustrated.

-Technical Fellow (division removed to preserve anonymity)

Continuously updating their mental models at all levels of abstraction—ranging from technical details to industry trends—by explicitly evaluating changes in their context

…Continuing To Make The Optimal Choice

New options become available: what was impossible yesterday, may be possible today

The computing context change: expected outcomes may change over time

Mentally Capable Of Handling Complexity…

To solve the problem, [great engineers] have to have the ability to connect things… You are always debugging layers of stacks of code… this layer talks to some other layer in the horizontal... you need to solve the problem and you don't know what's going on.

-Senior SDE, Windows Services

Grasping and reasoning about complex and intertwining ideas with agility

…Decisions, In Practice, Are Complicated

Software build on top of many layers of technology

Software interact with many other components and other software systems

Myriad considerations and constraints

Take Away #2:

Making good decisions is rarely discussed in the software engineering literature, but it is critical to being a great software engineer

Interactions With Teammates

The way [this great software engineer] just kind of touch people, just dissolves the conflicts right there… that magic to make people respect him. That's fun magic, I think that not everyone possess.

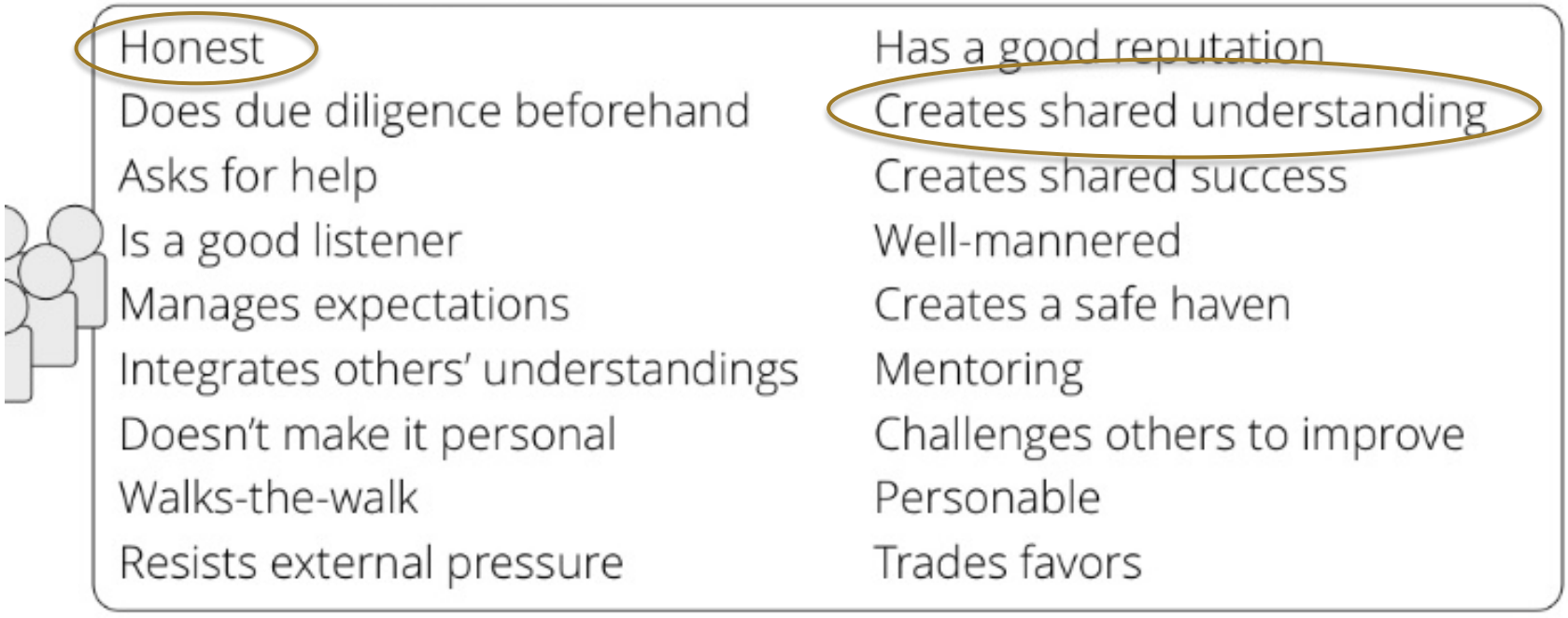
-Senior SDE, Windows

Honest	Has a good reputation
Does due diligence beforehand	Creates shared understanding
Asks for help	Creates shared success
Is a good listener	Well-mannered
Manages expectations	Creates a safe haven
Integrates others' understandings	Mentoring
Doesn't make it personal	Challenges others to improve
Walks-the-walk	Personable
Resists external pressure	Trades favors

Interactions With Teammates

The way [this great software engineer] just kind of touch people, just dissolves the conflicts right there… that magic to make people respect him. That's fun magic, I think that not everyone possess.

-Senior SDE, Windows



Honest…

Someone else trusting you… 'I know that this person always speaks the truth.' As a result of that, when they say something is good, I will totally believe them because they are not trying to kind of misrepresent something or make them look better...

-Principal Dev Manager, Windows Services

Provide credible information that others can act on

…Critical For Trust

Engineers want to be solving problems not be shifting blame

Lack of honesty paralyzes ability to make forward progress

Dishonesty: an important reason for *leaving* an organization

Creates Shared Understanding…

Our areas where the things are inherently difficult to talk about… business partners or with a customer… they think about things in much different terms… you have to kind of switch gears… why you should care about it and here is how you should think about it.

-Principal Dev Lead, Corp Dev

Adjusting the message to effectively mold another person's understanding of a situation

…Essential For Communicating Effectively

Communicating with others (e.g. partners and customers): need to adjust the language to be comprehensible

Marshalling resources to complete projects: completing large projects requires getting everyone ‘on the same page’

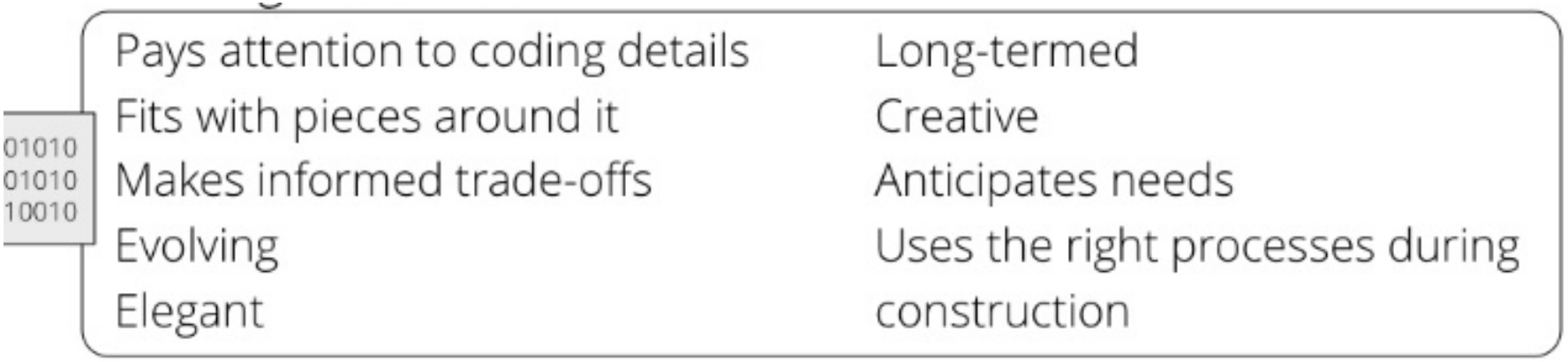
Take Away #3:

Software engineering is a *socio*technical undertaking

Engineering The Software Product

The style… always, an idea, and it was all clean… very concise. Just looking at it, you can say, "Okay, this guy, he knew what he was doing."… There's no extra stuff. Everything is minimally necessary and sufficient as it should be. It's well thought-out off screen.

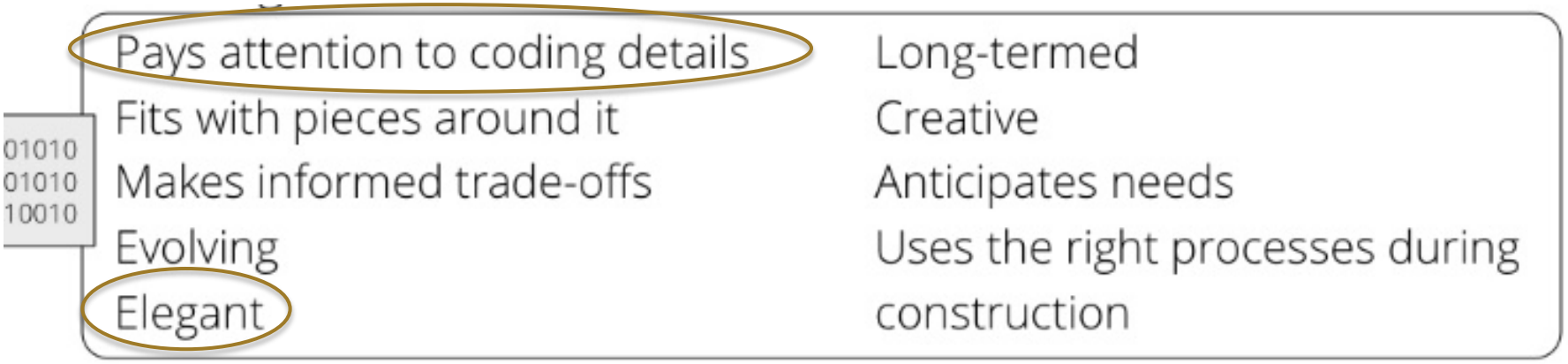
- Senior SDE, Windows



Engineering The Software Product

The style… always, an idea, and it was all clean… very concise. Just looking at it, you can say, "Okay, this guy, he knew what he was doing."… There's no extra stuff. Everything is minimally necessary and sufficient as it should be. It's well thought-out off screen.

- Senior SDE, Windows



Pays Attention To Coding Details…

This code is performance critical, compatibility sensitive, and is used in a huge variety of contexts. If a developer fails to handle an error, some customer will hit it, and we will likely need to issue a hotfix; if a developer implements an inefficient algorithm (N^2 is not ok)… consumes memory excessively in some environment…

-Principal SDE, Windows

Quality code that considers error handling, memory consumption, performance, security, and style

…Respected By Peers

"Greatness" is peer bestowed: engineers that cannot get the basics right are not respected

Elegant…

Never complicate any things… when you simplify things it becomes easier for you to maintain, going forward for customers… You get lesser number of issues reported by a customer.

-Senior Dev Lead, Dynamics

Simple and intuitive designs that another person (or themselves later) could easily understand.

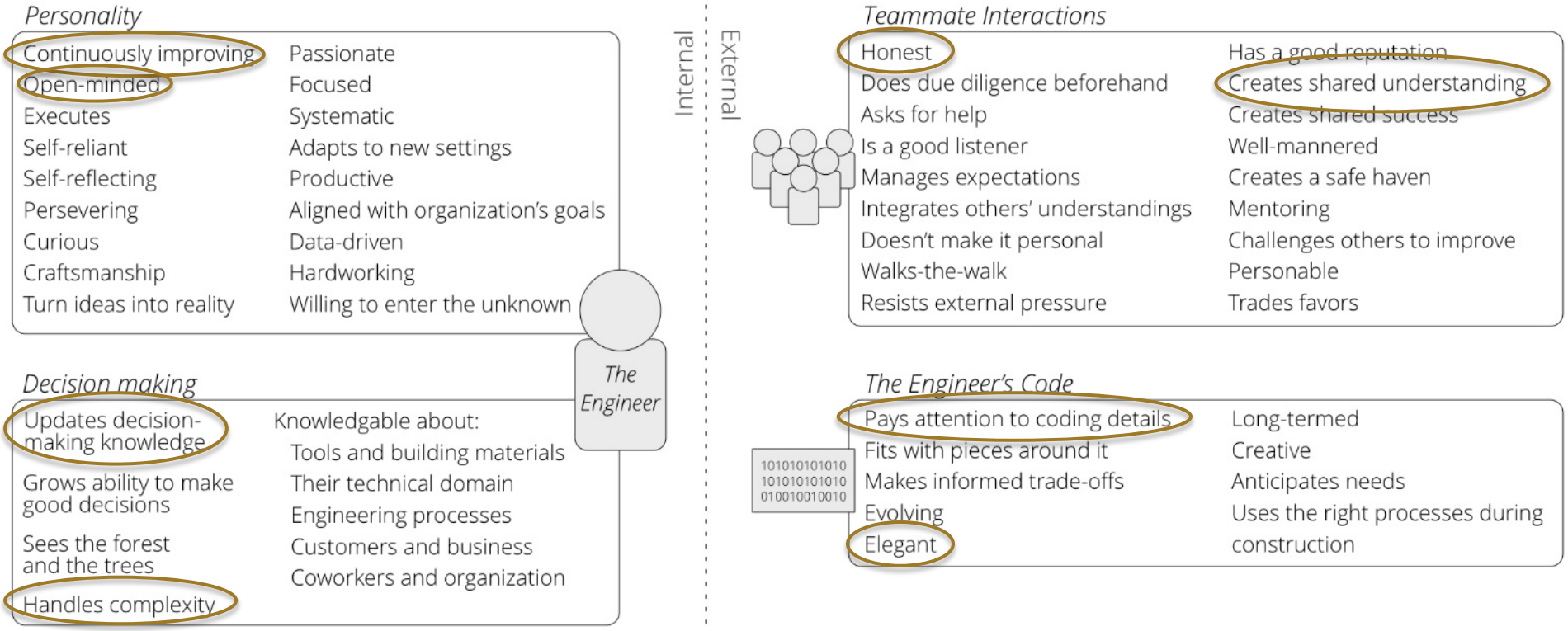
…Avoiding Complexity Can Be Difficult

Complexity is bad, but often unavoidable: those that can come up with elegant designs are revered

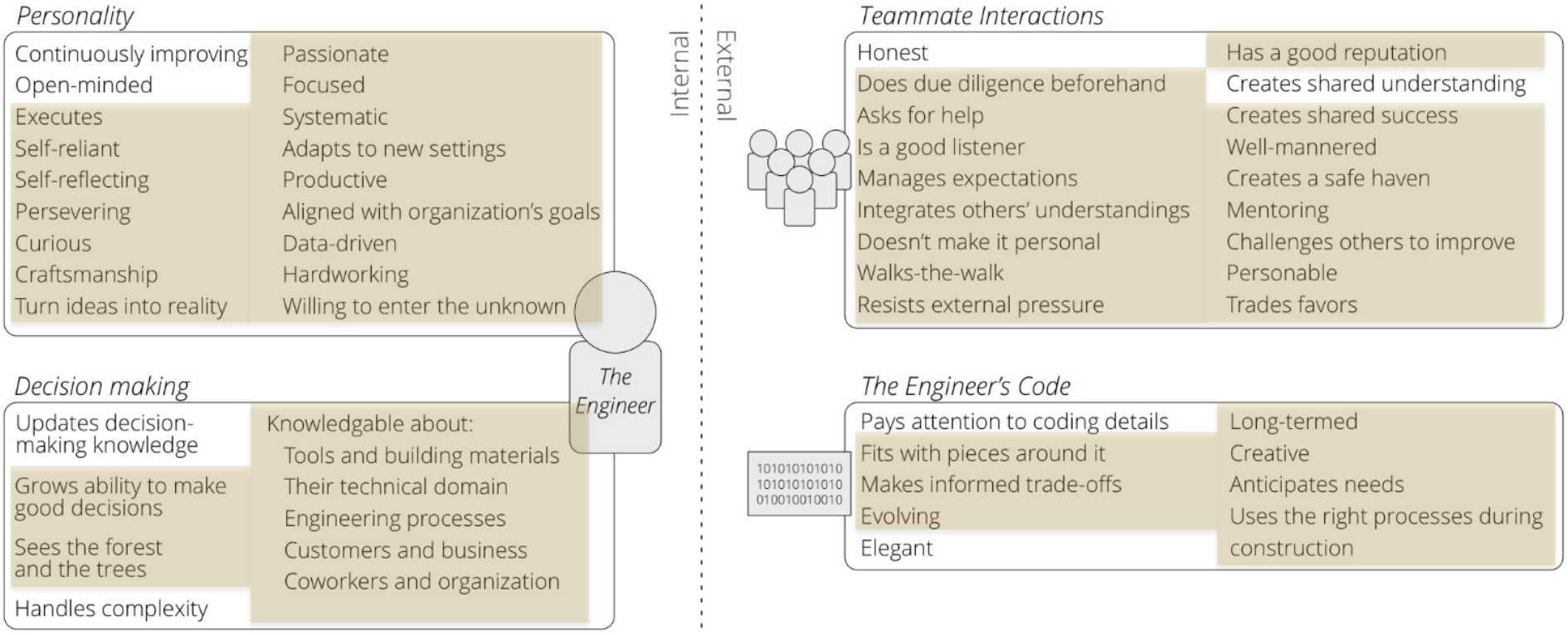
Take Away #4:

Delivering the code is often insufficient; complex contextual technical considerations abound.

Discussed Eight Attributes In The Four Areas…



Many Other Interesting and Important Attributes



http://dl.acm.org/citation.cfm?id=2818839

Help Leaders Of Engineers To...

Make better hiring decisions: especially when reasoning about non-technical attributes

Improve attributes associated with leadership

Cultivate desirable attributes within your team: avoid deleterious attributes that cause great engineers to leave

Help Young Engineers To...

Target areas for improvement

Find the right fit with teams: different teams emphasize various attributes differently

Better present yourself to potential employers

Help Educators To…

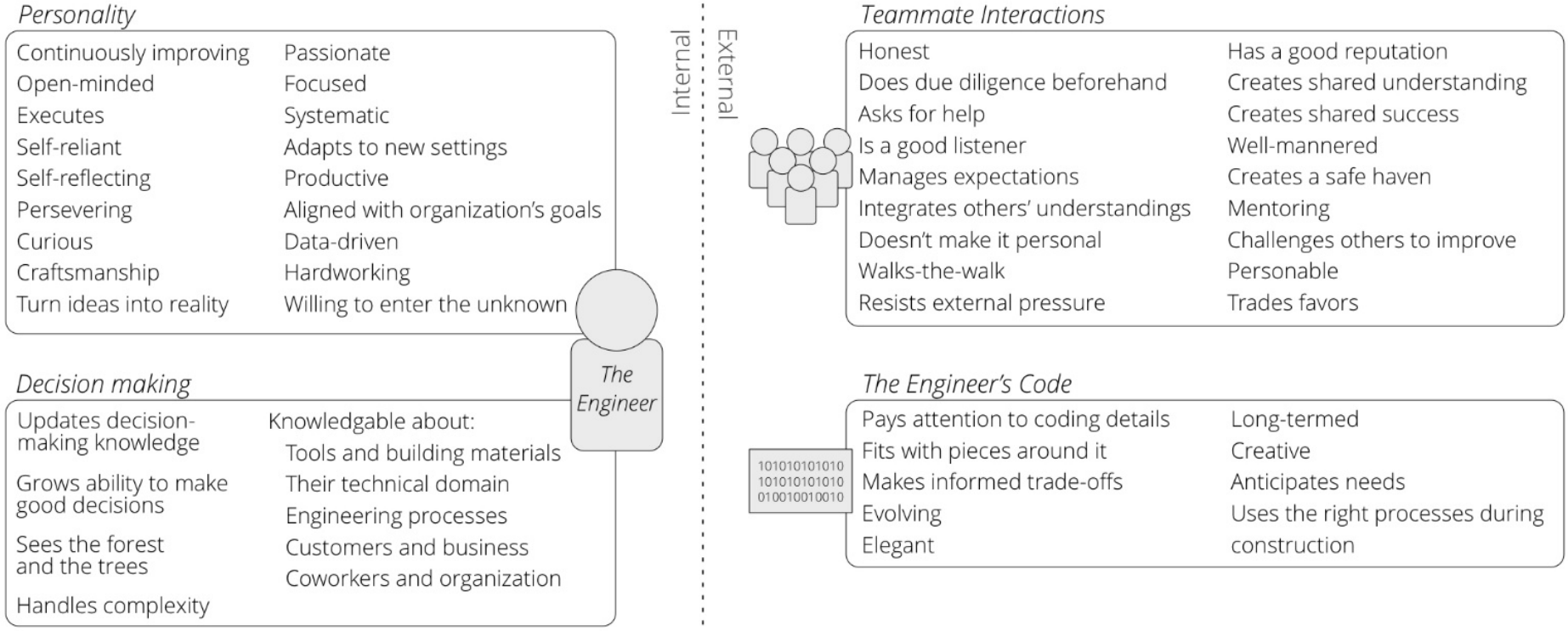
Consider new topics for software engineering curriculum: decision making (e.g. Herbert Simon)

Prepare students for necessary attributes not amenable to be taught in academic settings (e.g. self-reliant)

Thanks to our informants!

What makes a great software engineer?

http://dl.acm.org/citation.cfm?id=2818839



This work was supported in part by Microsoft, Google, and the National Science Foundation (NSF) under Grants CCF-0952733, CNS-1240786, and IIS-1314399.

ACM: The Learning Continues…

- Questions about this webcast? learning@acm.org
- ACM Learning Webinars (on-demand archive): http://learning.acm.org/webinar
- ACM Learning Center: http://learning.acm.org
- ACM Queue: http://queue.acm.org/
- ACM SIGSOFT: http://www.sigsoft.org/