

# FOSTERING INTERDISCIPLINARY TEAMS

(Process and Team Reflections)

Christian Kaestner

Required reading: Kim, Miryung, Thomas Zimmermann, Robert DeLine, and Andrew Begel. "[Data scientists in software teams: State of the art and challenges](#)." IEEE Transactions on Software Engineering 44, no. 11 (2017): 1024-1038.

# LEARNING GOALS

- Understand different roles in projects for AI-enabled systems
- Plan development activities in an inclusive fashion for participants in different roles
- Diagnose and address common teamwork issues
- Describe agile techniques to address common process and communication issues

# **CASE STUDY: DEPRESSION DETECTION ON SOCIAL MEDIA**



# THE PROJECT

- Social media company of about 1400 employees in the US, 300 developers and data scientists
- Use sentiment analysis on video data (and transcripts) to detect depression
- Planned interventions through recommending different content and showing ads for getting support, design for small group features
- Collaboration with mental health professionals and ML researches at top university



**Data  
Scientists**

**Software  
Engineers**

# SOFTWARE ENGINEER

## DATA SCIENTIST

- Often fixed dataset for training and evaluation (e.g., PBS interviews)
- Focused on accuracy
- Prototyping, often Jupyter notebooks or similar
- Expert in modeling techniques and feature engineering
- Model size, updateability, implementation stability typically does not matter

- Builds a product
- Concerned about cost, performance, stability, release time
- Identify quality through customer satisfaction
- Must scale solution, handle large amounts of data
- Detect and handle mistakes, preferably automatically
- Maintain, evolve, and extend the product over long periods
- Consider requirements for security, safety, fairness

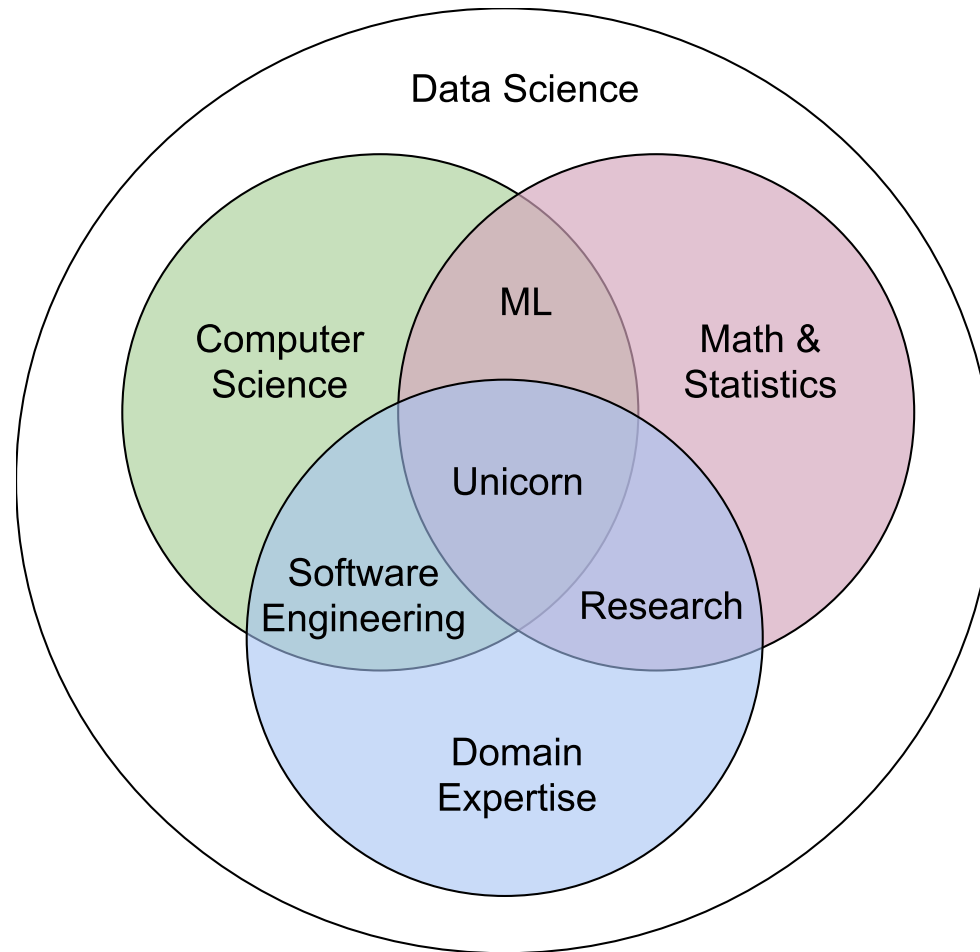
# CONTINUUM OF SKILLS

- Software Engineer
- Data Engineer
- Data Scientist
- Applied Scientist
- Research Scientist

Talk: Ryan Orban. [Bridging the Gap Between Data Science & Engineer: Building High-Performance Teams](#). 2016







By Steven Geringer, via Ryan Orban. [Bridging the Gap Between Data Science & Engineer: Building High-Performance Teams](#). 2016

# DATA SCIENTISTS AT MICROSOFT

- Mostly analyzing product and customer data
- User engagement (which features users like and use, satisfaction, retention)
- Software productivity (bug prioritization, monitoring)
- Domain-specific problems (NLP quality, stock pricing, power prediction)
- Business intelligence (predicting investment, demand, sales)

Kim, Miryung, Thomas Zimmermann, Robert DeLine, and Andrew Begel. "[Data scientists in software teams: State of the art and challenges.](#)" IEEE Transactions on Software Engineering 44, no. 11 (2017): 1024-1038.g

# DATA SCIENCE ROLES AT MICROSOFT

- Polymath
- Data evangelist
- Data preparer
- Data shaper
- Data analyzer
- Platform builder
- 50/20% moonlighter
- Insight actors

Kim, Miryung, Thomas Zimmermann, Robert DeLine, and Andrew Begel. "[Data scientists in software teams: State of the art and challenges](#)." IEEE Transactions on Software Engineering 44, no. 11 (2017): 1024-1038.

# MANY OTHER ROLE DESCRIPTIONS

- Data scientist
- Data analyst
- Data architect
- Data engineer
- Statistician
- Database administrator
- Business analyst
- Data and analytics manager

e.g. Martijn Theuwissen. [The different data science roles in the industry](#). 2015

# MANY OTHER ROLE DESCRIPTIONS

- Product Data Analyst (feature analysis)
- Business Intelligence, Analytics & Reporting (marketing)
- Modeling Analyst (financial forecasting)
- Machine Learning Engineer (user facing applications)
- Hybrid Data Engineer/Data Scientist (data pipelining)
- Hybrid Data Visualization Expert (communication, storytelling)
- Data Science Platforms & Tools Developer (supporting role)

e.g. Yorgos Askalidis . [Demystifying data science roles](#). 2019

# EVOLUTION OF DATA SCIENCE ROLES



*More or less engineering focus? More or less statistics focus? ...*

# SOFTWARE ENGINEERING SPECIALIZATIONS

- Architectures
- Requirements engineers
- Testers
- Site reliability engineers
- Devops
- Safety
- Security
- UX
- Distributed systems, cloud
- ...



# NEEDED ROLES IN AI SYSTEMS PROJECTS?



# OTHER ROLES IN AI SYSTEMS PROJECTS?

- **Domain specialists**
- Business, management, marketing
- Project management
- Designers, UI experts
- Operations
- Safety, security specialist
- Big data specialist
- Lawyers
- Social scientists, ethics
- ...

# INTERDISCIPLINARY TEAMS

# UNICORNS -> TEAMS

- Domain experts
- Data scientists
- Software engineers
- Operators
- Business leaders

# NECESSITY OF GROUPS

- Division of labor
- Division of expertise (e.g., security expert, ML expert, data cleaning expert, database expert)

# TEAM ISSUES

- Process costs
- Groupthink
- Social loafing
- Multiple/conflicting goals

# **TEAM ISSUE: PROCESS COSTS**

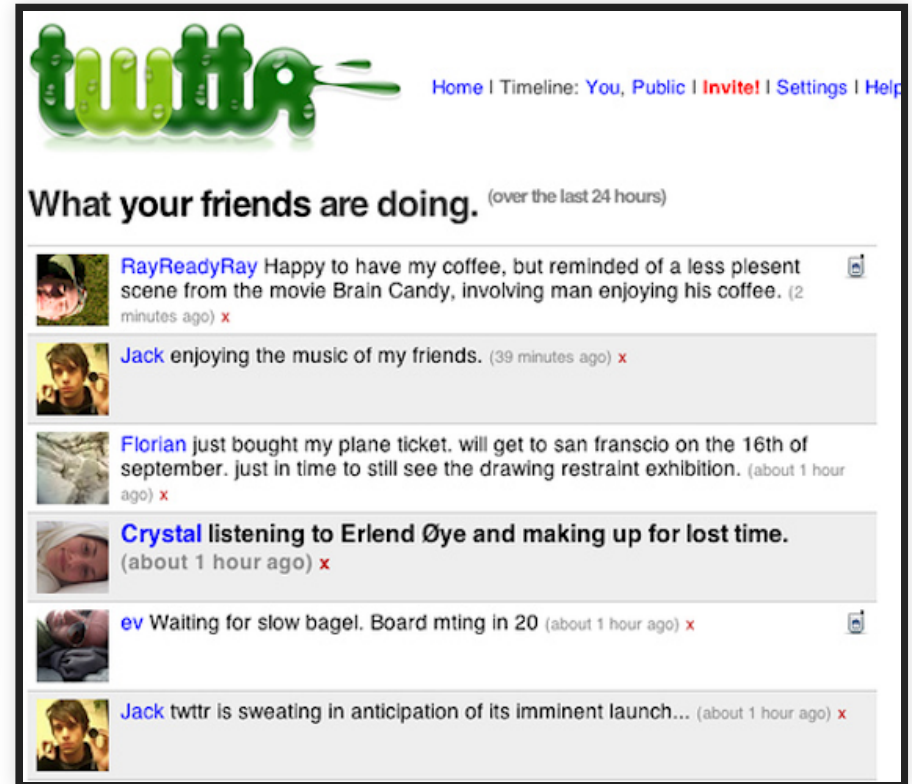
# CASE STUDIES

Disclaimer: All pictures represent abstract developer groups or products to give a sense of scale; they are not necessarily the developers of those products or developers at all.



# HOW TO STRUCTURE TEAMS?

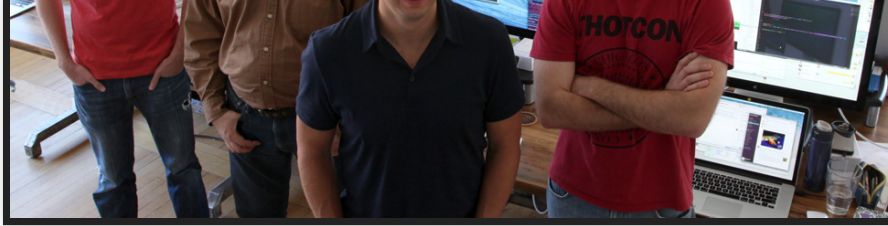
Microblogging platform; 3 friends



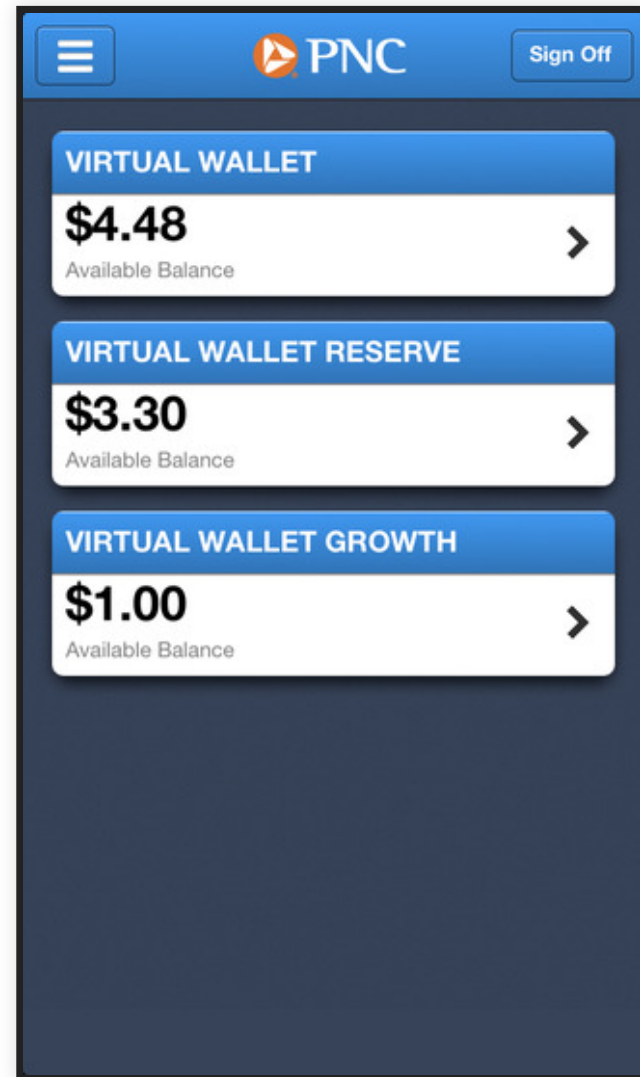
# HOW TO STRUCTURE TEAMS?

Banking app; 15 developers and data analysts





*(Instagram had 13 employees when they were bought for 1B in 2012)*



# HOW TO STRUCTURE TEAMS?

Mobile game; 50ish developers; distributed teams?





# HOW TO STRUCTURE TEAMS?

Mobile game; 50ish developers; distributed teams?



**FROM TWO PEOPLE TO 200**





# HOW TO STRUCTURE TEAMS?

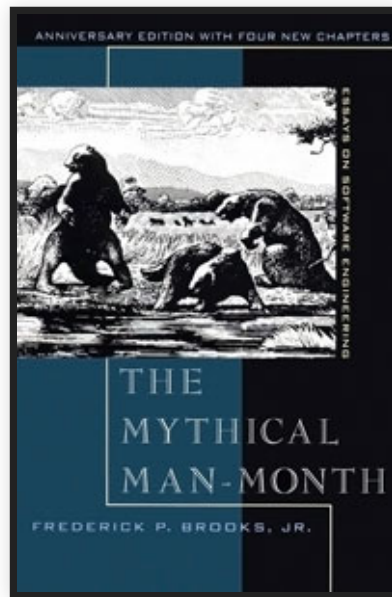
Self-driving cars; 1200 developers and data analysts





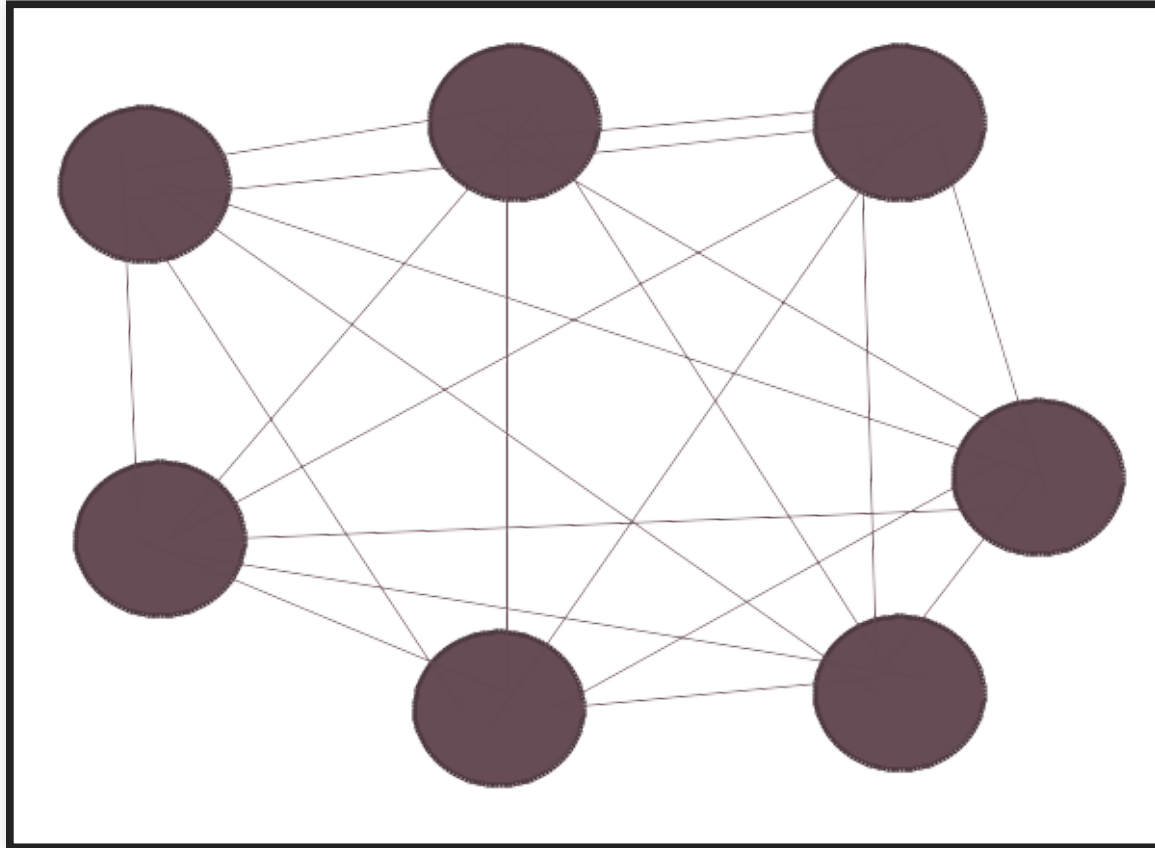
# MYTHICAL MAN MONTH

*Brooks's law: Adding manpower to a late software project makes it later*



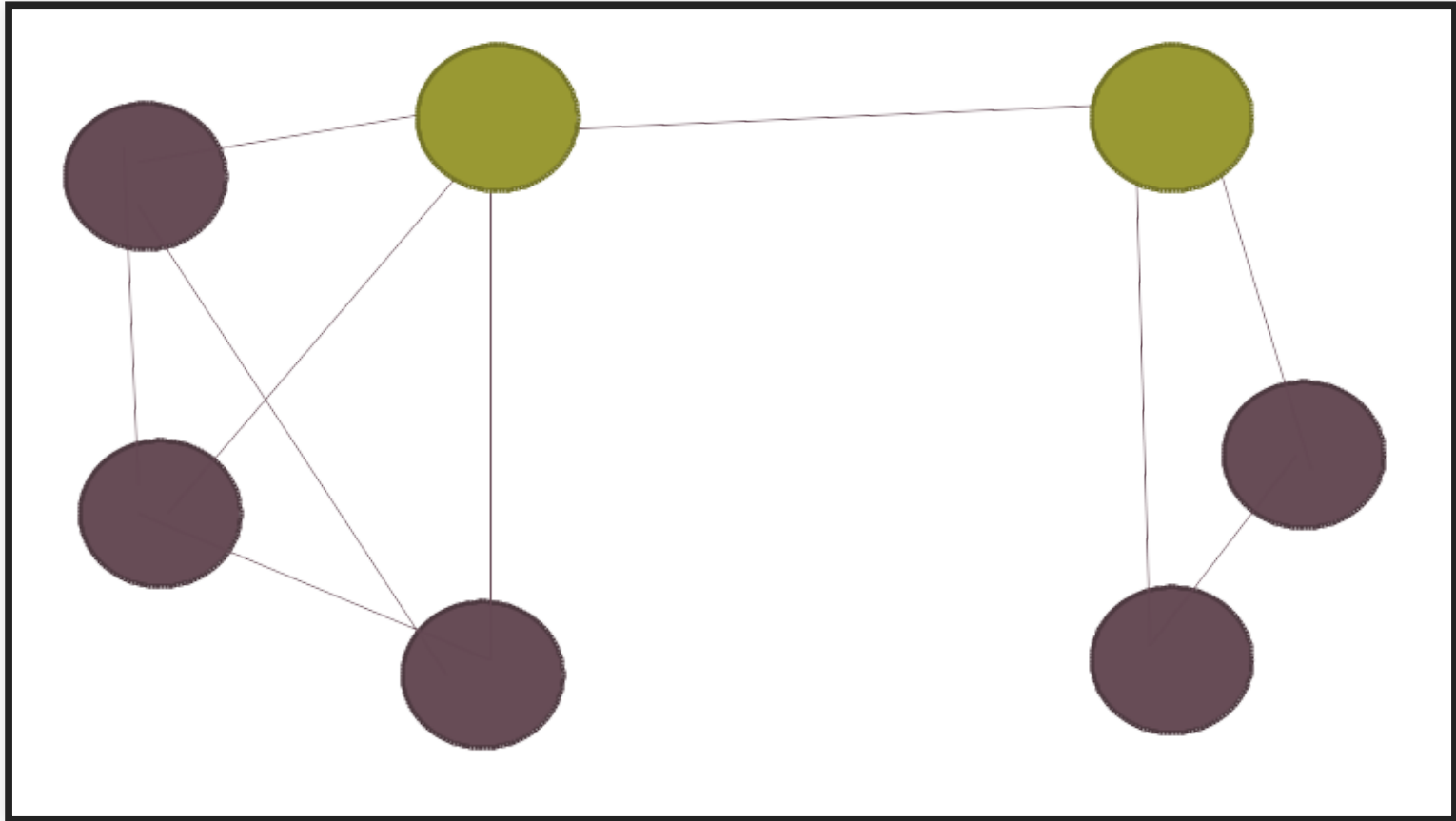
1975, describing experience at IBM developing OS/360

# PROCESS COSTS



$n(n - 1) / 2$  communication links

# PROCESS COSTS



# BROOK'S SURGICAL TEAMS

- Chief programmer – most programming and initial documentation
- Support staff
  - Copilot: supports chief programmer in development tasks, represents team at meetings
  - Administrator: manages people, hardware and other resources
  - Editor: editing documentation
  - Two secretaries: one each for the administrator and editor
  - Program clerk: keeps records of source code and documentation
  - Toolsmith: builds specialized programming tools
  - Tester: develops and runs tests
  - Language lawyer: expert in programming languages, provides advice on producing optimal code.

Brooks. The Mythical Man-Month. 1971

Speaker notes

Would assume unicorns in today's context.

# MICROSOFT'S SMALL TEAM PRACTICES

- Vision statement and milestones (2-4 month), no formal spec
- Feature selection, prioritized by market, assigned to milestones
- Modular architecture
- Allows small federated teams (Conway's law)
- Small teams of overlapping functional specialists

(Windows 95: 200 developers and testers, one of 250 products)

# MICROSOFT'S FEATURE TEAMS

- 3-8 developers (design, develop)
- 3-8 testers (validation, verification, usability, market analysis)
- 1 program manager (vision, schedule communication; leader, facilitator) – working on several features
- 1 product manager (marketing research, plan, betas)

# MICROSOFT'S PROCESS

- "Synchronize and stabilize"
- For each milestone
  - 6-10 weeks feature development and continuous testing frequent merges, daily builds
  - 2-5 weeks integration and testing (“zero-bug release”, external betas )
  - 2-5 weeks buffer



# AGILE PRACTICES (E.G., SCRUM)

- 7±2 team members, collocated
- self managing
- Scrum master (potentially shared among 2-3 teams)
- Product owner / customer representative

*Large teams (29 people) create around six times as many defects as small teams (3 people) and obviously burn through a lot more money. Yet, the large team appears to produce about the same mount of output in only an average of 12 days' less time. This is a truly astonishing finding, through it fits with my personal experience on projects over 35 years. - Phillip Amour, 2006, CACM 49:9*

# ESTABLISH COMMUNICATION PATTERNS

- Avoid overhead
  - Ensure reliability
  - Constraint latency
- 
- e.g. Issue tracker vs email; online vs face to face

# AWARENESS

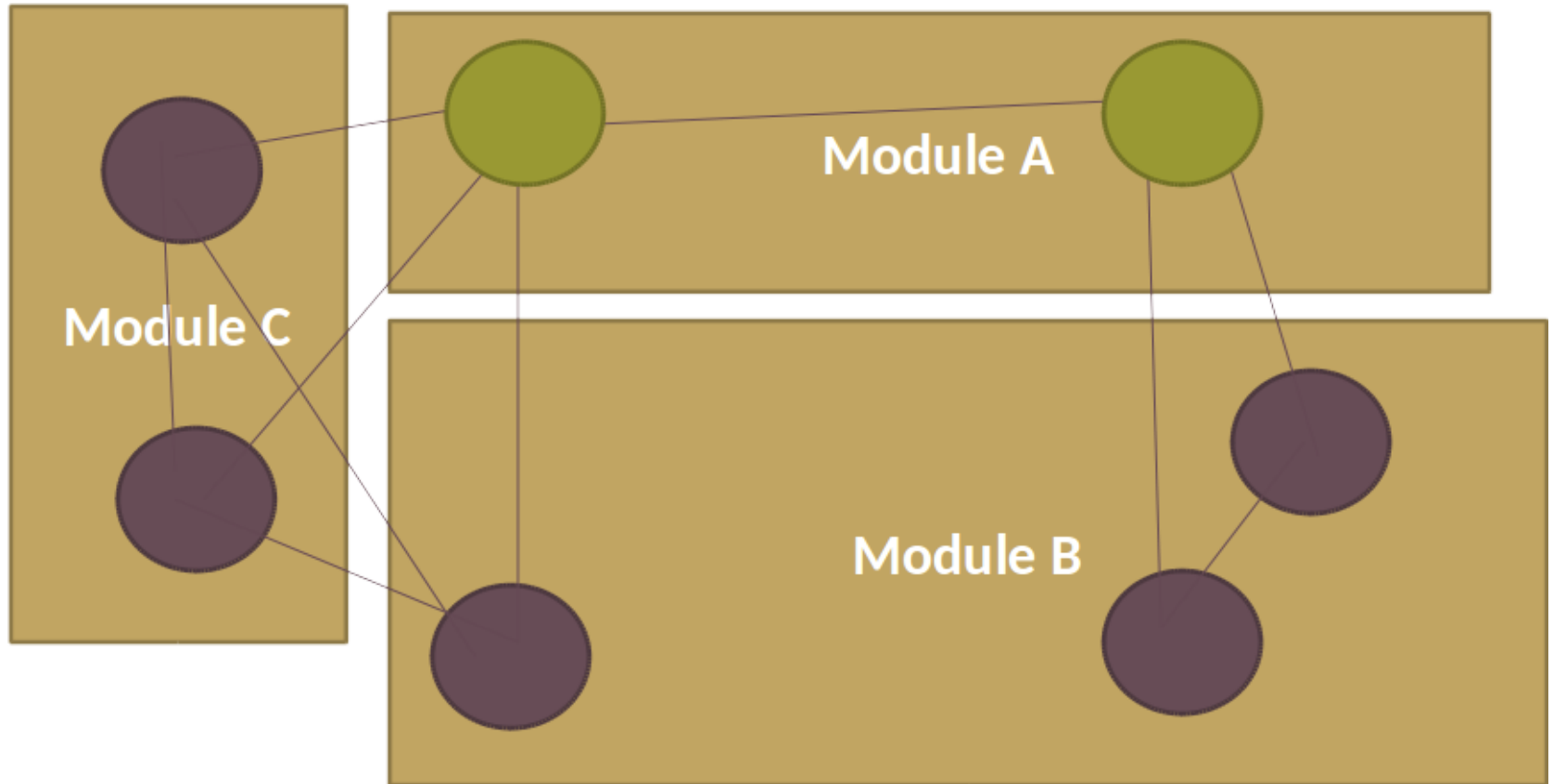
- Notifications
- Brook's documentation book
- Email to all
- Code reviews

# CONWAY'S LAW

*“Any organization that designs a system (defined broadly) will produce a design whose structure is a copy of the organization's communication structure.” — Mel Conway, 1967*

*“If you have four groups working on a compiler, you'll get a 4-pass compiler.”*

# CONGURENCE



Structural congruence, Geographical congruence, Task congruence, IRC communication congruence

# TEAM STRUCTURE FOR TRANSCRIPTION SERVICE?

the-changelog-318 ← Dashboard | Quality: High ⓘ Last saved a few seconds ago ... Share

00:00 Offset 00:00 01:31:27

Play Back 5s **1x** Speed Volume

**NOTES**  
Write your notes here

**Speaker 5** ▶ 07:44

Yeah. So there's a slight story behind that. So back when I was in, **uh**, Undergrad, I wrote a program for myself to measure a, **the** amount of time I did data entry **from** my father's business and I was on windows at the time and there wasn't a function called time dot **[inaudible]** time, **uh**, which I **needed** to parse dates to get back to time, **top** of representation, **uh**, I figured out a way to do it and I gave it to what's called the python cookbook because it just seemed like something other people could use. So **it was** just trying to be helpful. **Uh**, subsequently I had to figure out how to make it work **because** I didn't really have to. Basically, it bothered me that you had to input all the **locale** information and I figured out how to do it over **the subsequent months**. And actually as a graduation gift from my Undergrad, the week following, I solved it and wrote it all out.

**Speaker 5** ▶ 08:38

And I asked, **uh**, Alex Martelli, the editor of the Python Cookbook, which had published my original recipe, **a**, how do I get this into python? I think it might help

How did we do on your transcript? ☆☆☆☆☆





# ENGINEERING RECOMMENDATIONS FOR STRUCTURING AI-ENABLED SYSTEMS?



*How to design the AI-enabled system to enable better collaboration?*

# ENGINEERING RECOMMENDATIONS FOR STRUCTURING AI-ENABLED SYSTEMS

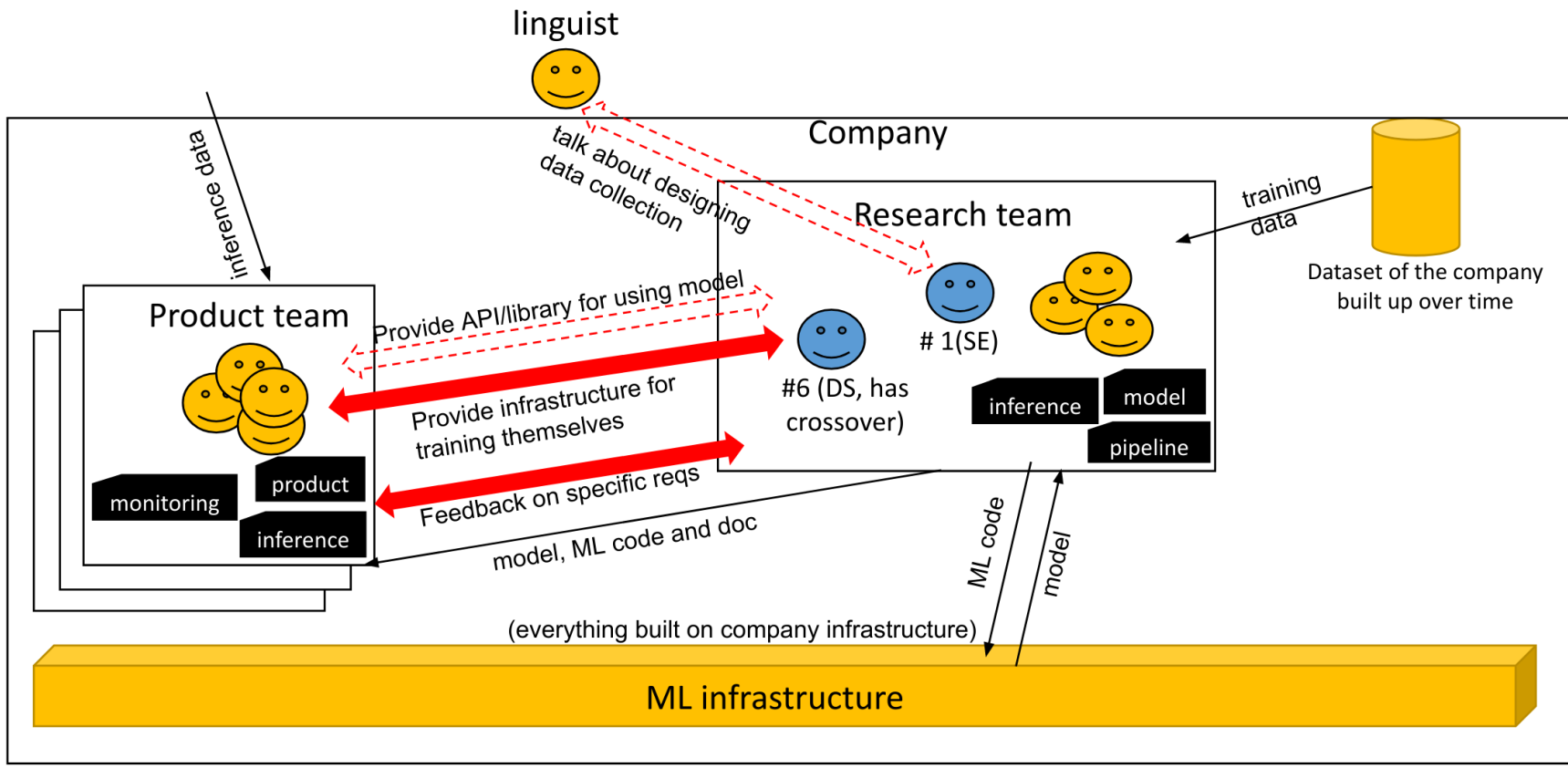
- Decompose the system
- Independent components (e.g. microservices)
- Isolate AI if possible
- Clear, stable interfaces, minimal coupling
- Monitoring to observe contracts and quality

# COMMON CHALLENGE: ESTABLISHING INTERFACES

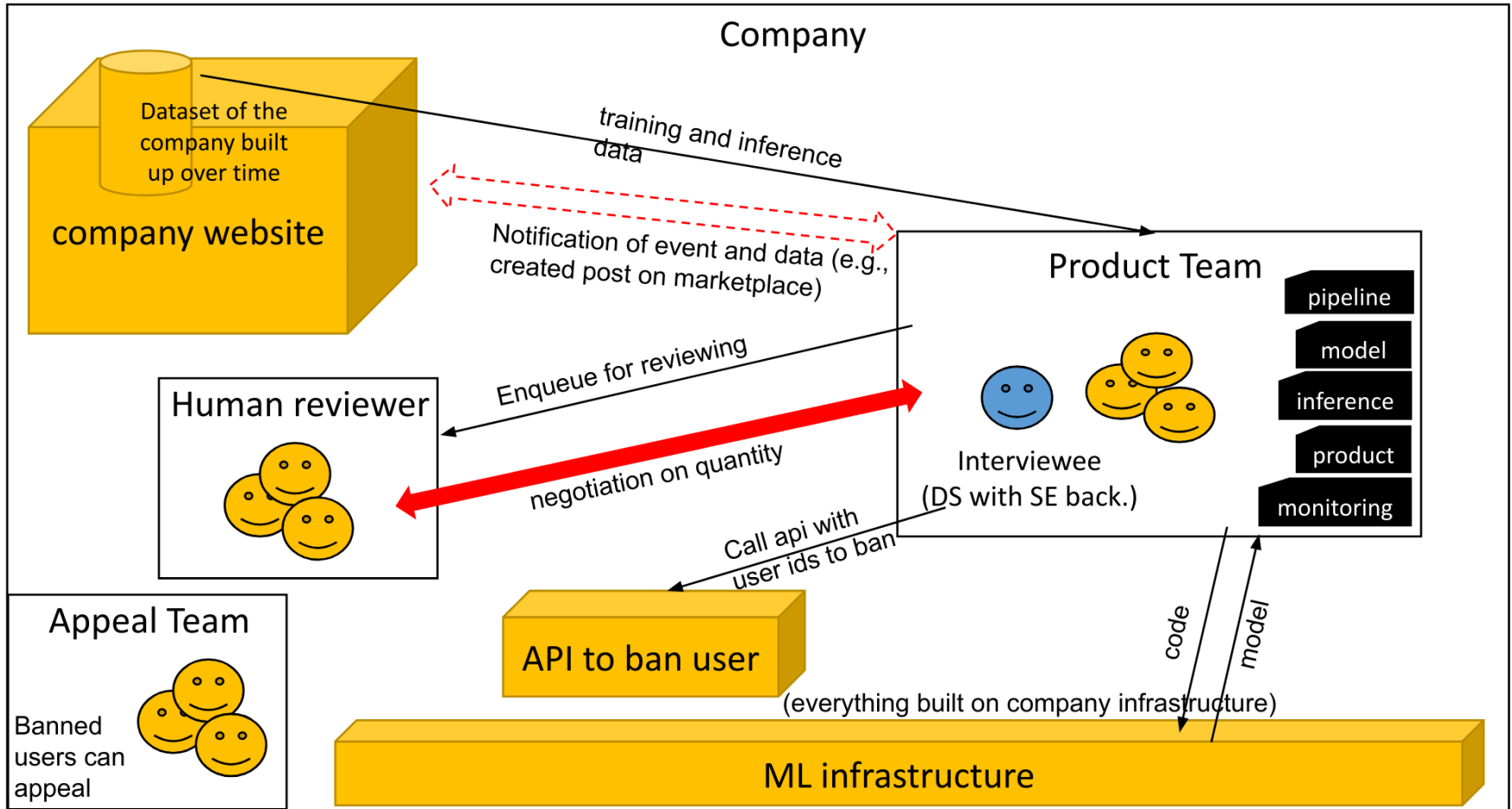
- When dividing work, need to agree on interface
- Common source of mismatch and friction
- **Examples?**
  - Team A uses data produced by Team B
  - Team C deploys model produced by team A
  - Team D uses model and needs to provide feedback to Team A
  - Team D waits for improvement/feature from model A



Research team in big tech company creates NLP-ML components used within various products in the company



# Building product to detect scam on company platform



# COMMON CHALLENGE: ESTABLISHING INTERFACES

- Formal vs informal agreements?
- Service level agreements and automated enforcement?
- Close collaboration vs siloed teams?
  
- Many concerns: prediction accuracy, generalization, execution time, scalability, data quality, data quantity, feedback latency, privacy, explainability, time estimation, ...
- Formal agreements and enforcement expensive, slowing development? see technical debt

**Story time**

# COLLABORATION POINTS (PRELIMINARY)

1. Understanding system requirements and ML capabilities
2. Understanding ML-specific requirements at the system level, reasoning about feedback loops
3. Project planning and architecture design
4. Data needs, data quality, data meaning
5. Documenting model output
6. Planning and monitoring for drift
7. Planning ML component QA (offline, online, monitoring)
8. Planning system QA (integration, interaction, safety, feedback loops)
9. Tool support for data scientists
10. From prototype to production (pipelines, versioning, operations, user interactions, ...)



# BREAKOUT: TEAM STRUCTURE FOR SOCIAL MEDIA/DEPRESSION PROJECT?

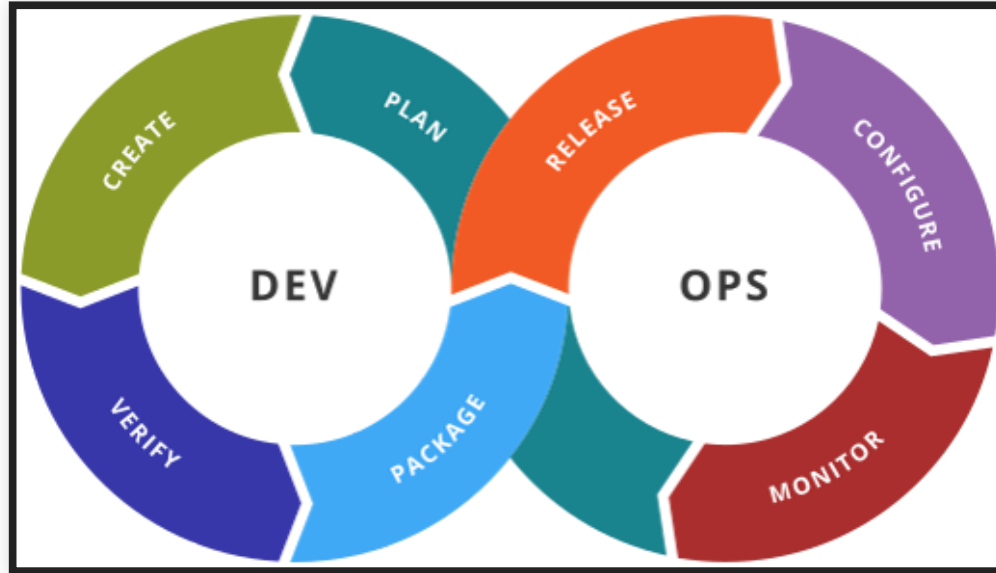


*What teams? What roles? What are important interfaces between teams?*

# **TEAM ISSUES: MULTIPLE/CONFLICTING GOALS**

(Organization of Interdisciplinary Teams)

# CONFLICTING GOALS?



# CONFLICTING GOALS?



A Venn diagram consisting of two overlapping circles. The left circle is light green and contains the text 'Data Scientists'. The right circle is light orange and contains the text 'Software Engineers'. The overlapping area in the center is a darker shade of brownish-orange.

**Data  
Scientists**

**Software  
Engineers**

# CONFLICTING GOALS?



A Venn diagram consisting of two overlapping circles. The left circle is light green and contains the text 'Data Scientists'. The right circle is pink and contains the text 'Compliance Lawyers'. The overlapping area in the center is a darker shade of red.

Data  
Scientists

Compliance  
Lawyers

# CONFLICTING GOALS?



# HOW TO ADDRESS GOAL CONFLICTS?



# T-SHAPED PEOPLE

*Broad-range generalist + Deep expertise*

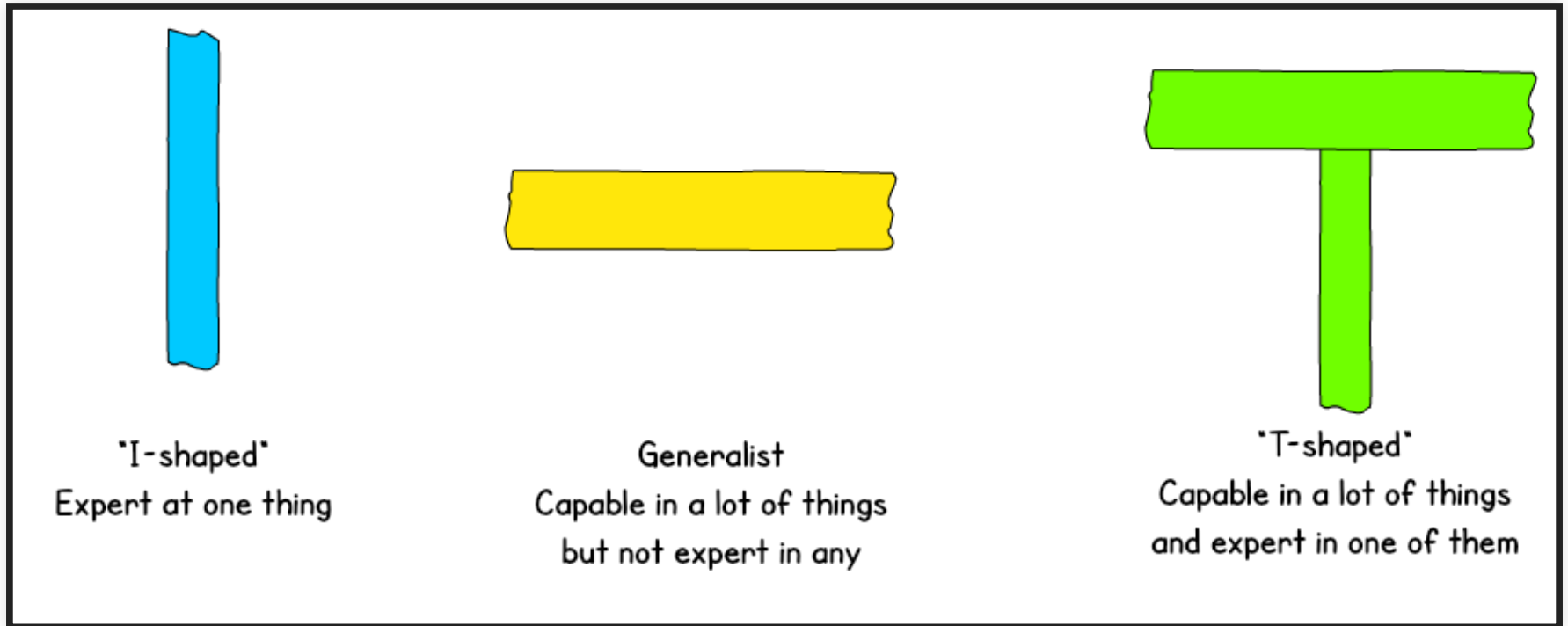


Figure: Jason Yip. [Why T-shaped people?](#). 2018



# T-SHAPED PEOPLE

*Broad-range generalist + Deep expertise*

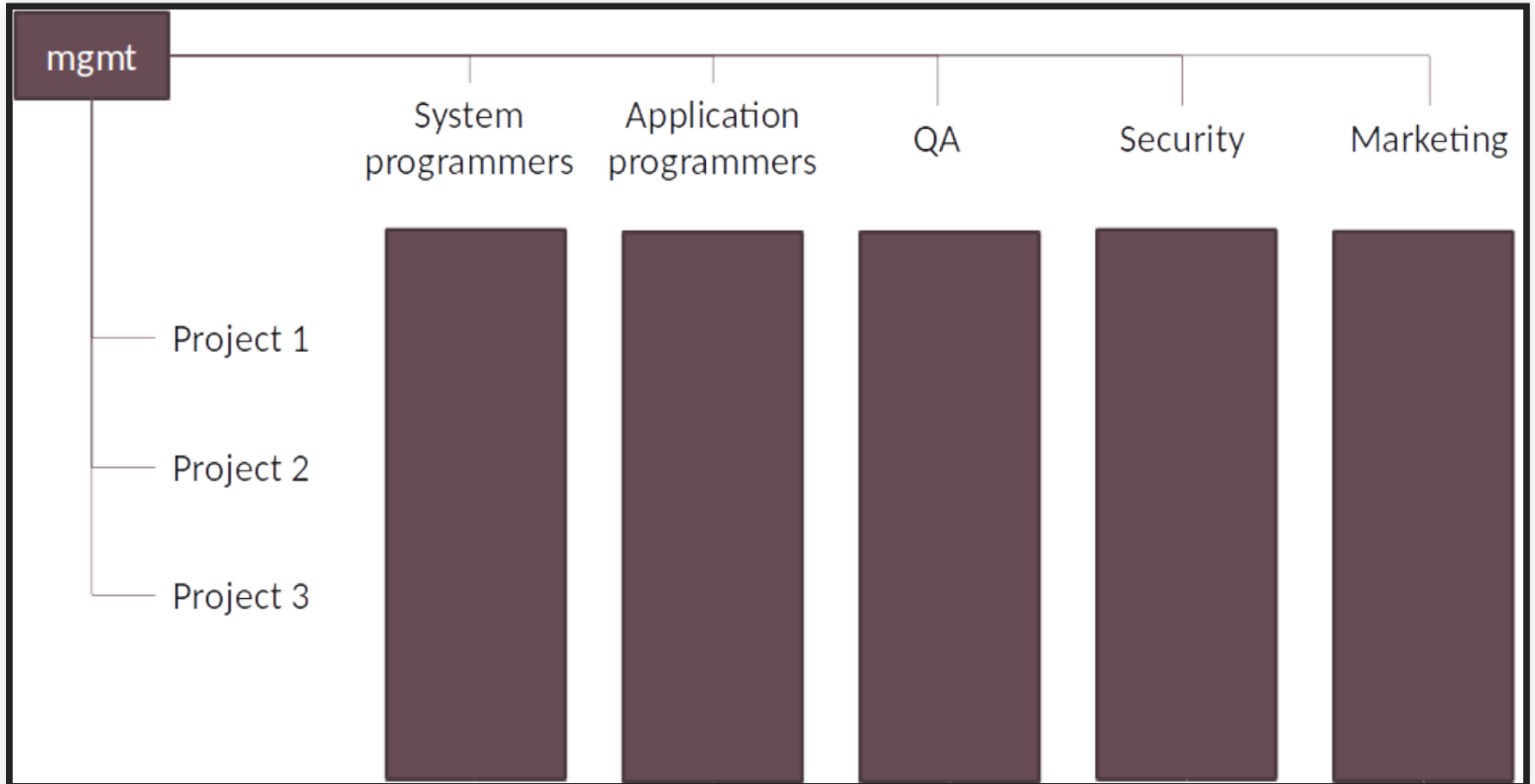
Example:

- Basic skills of software engineering, business, distributed computing, and communication
- Deep skills in deep neural networks (technique) and medical systems (domain)

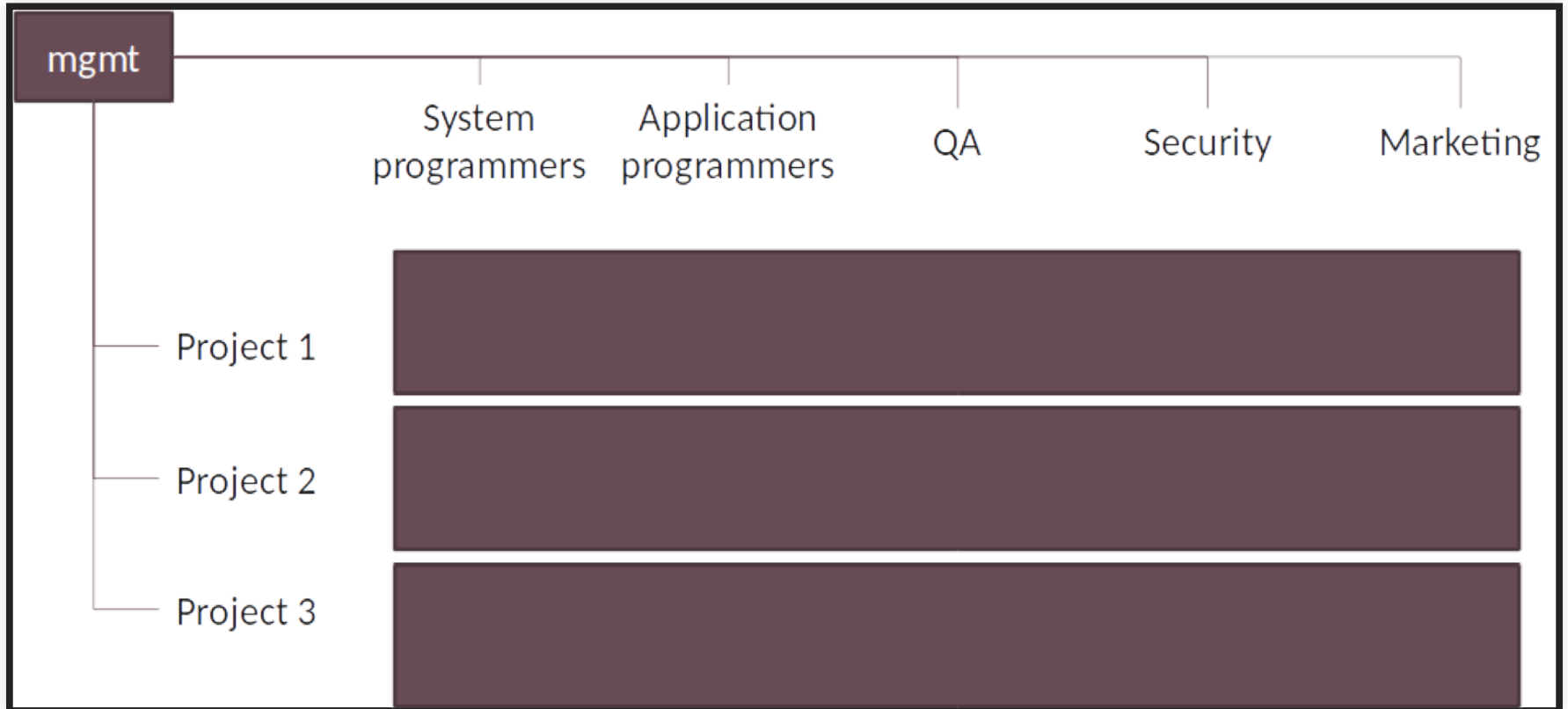
# TEAM COMPOSITION

- Cover deep expertise in all important areas
- Aim for overlap in general skills
  - Fosters communication, same language

# MATRIX ORGANIZATION



# PROJECT ORGANIZATION



# CASE STUDY: BRØDERBUND

*As the functional departments grew, staffing the heavily matrixed projects became more and more of a nightmare. To address this, the company reorganized itself into “Studios”, each with dedicated resources for each of the major functional areas reporting up to a Studio manager. Given direct responsibility for performance and compensation, Studio managers could allocate resources freely.*

*The Studios were able to exert more direct control on the projects and team members, but not without a cost. The major problem that emerged from Brøderbund’s Studio reorganization was that members of the various functional disciplines began to lose touch with their functional counterparts. Experience wasn’t shared as easily. Over time, duplicate effort began to appear.*

Mantle, Mickey W., and Ron Lichty. [Managing the unmanageable: rules, tools, and insights for managing software people and teams.](#) Addison-Wesley Professional, 2012.

# SPECIALIST ALLOCATION (ORGANIZATIONAL ARCHITECTURES)

- Centralized: development teams consult with a core group of specialists when they need help
- Distributed: development teams hire specialists to be a first-class member of the team
- Weak Hybrid: centralized group of specialists and teams with critical applications hire specialists
- Strong Hybrid: centralized group of specialists and most teams also hire specialists

**Tradeoffs?**

# EXAMPLE: SECURITY ROLES

- Everyone: “security awareness” – buy into the process
- Developers: know the security capabilities of development tools and use them, know how to spot and avoid relevant, common vulnerabilities
- Managers: enable the use of security practices
- Security specialists: everything security

# ALLOCATION OF DATA SCIENCE/SOFTWARE ENGINEERING EXPERTISE?





# COMMITMENT & ACCOUNTABILITY

- Conflict is useful, expose all views
- Come to decision, commit to it
- Assign responsibilities
- Record decisions and commitments; make record available

# BELL & HART – 8 CAUSES OF CONFLICT

- Conflicting resources.
- Conflicting styles.
- Conflicting perceptions.
- Conflicting goals.
- Conflicting pressures.
- Conflicting roles.
- Different personal values.
- Unpredictable policies.

*Understanding causes helps design interventions. Examples?*

Bell, Art. (2002). [Six ways to resolve workplace conflicts](#). University of San Francisco

# AGILE TECHNIQUES TO ADDRESS CONFLICTING GOALS?



# TEAM ISSUES: GROUPTHINK



# GROUPTHINK

- Group minimizing conflict
- Avoid exploring alternatives
- Suppressing dissenting views
- Isolating from outside influences
- -> Irrational/dysfunctional decision making

# EXAMPLE: TIME AND COST ESTIMATION



# EXAMPLE: USE OF HYPE TECHNOLOGY

(agile, block chain, machine learning, devops, AIOps, ...)



# CAUSES OF GROUPTHINK

- High group cohesiveness, homogeneity
- Structural faults (insulation, biased leadership, lack of methodological exploration)
- Situational context (stressful external threats, recent failures, moral dilemmas)



# SYMPTOMS

- Overestimation of ability: invulnerability, unquestioned believe in morality
  - Closed-mindedness: ignore warnings, stereotyping; innovation averse
- Pressure toward uniformity: self-censorship, illusion of unanimity, ...



# DIVERSITY

*“Men and women have different viewpoints, ideas, and market insights, which enables better **problem solving**. A gender-diverse workforce provides easier **access to resources**, such as various sources of credit, multiple sources of information, and wider industry knowledge. A gender-diverse workforce allows the company to serve an **increasingly diverse customer base**. Gender diversity helps companies **attract and retain talented women**.”*

*“Cultural diversity leads to **process losses** through task conflict and decreased social integration, but to process gains through increased creativity and satisfaction.”*

Stahl, Günter K., et al. "[Unraveling the effects of cultural diversity in teams: A meta-analysis of research on multicultural work groups.](#)"  
Journal of international business studies 41.4 (2010): 690-709.

Sangeeta Badal. [The Business Benefits of Gender Diversity](#). Gallup, 2014

# GROUPTHINK AND AI-ENABLED SYSTEM PROJECTS?



# GROUPTHINK AND AI

- Need of AI
- Selection of learning method
- Fairness
- Safety requirements (e.g. Pitt delivery robot)
- Ethics

# MITIGATION STRATEGIES



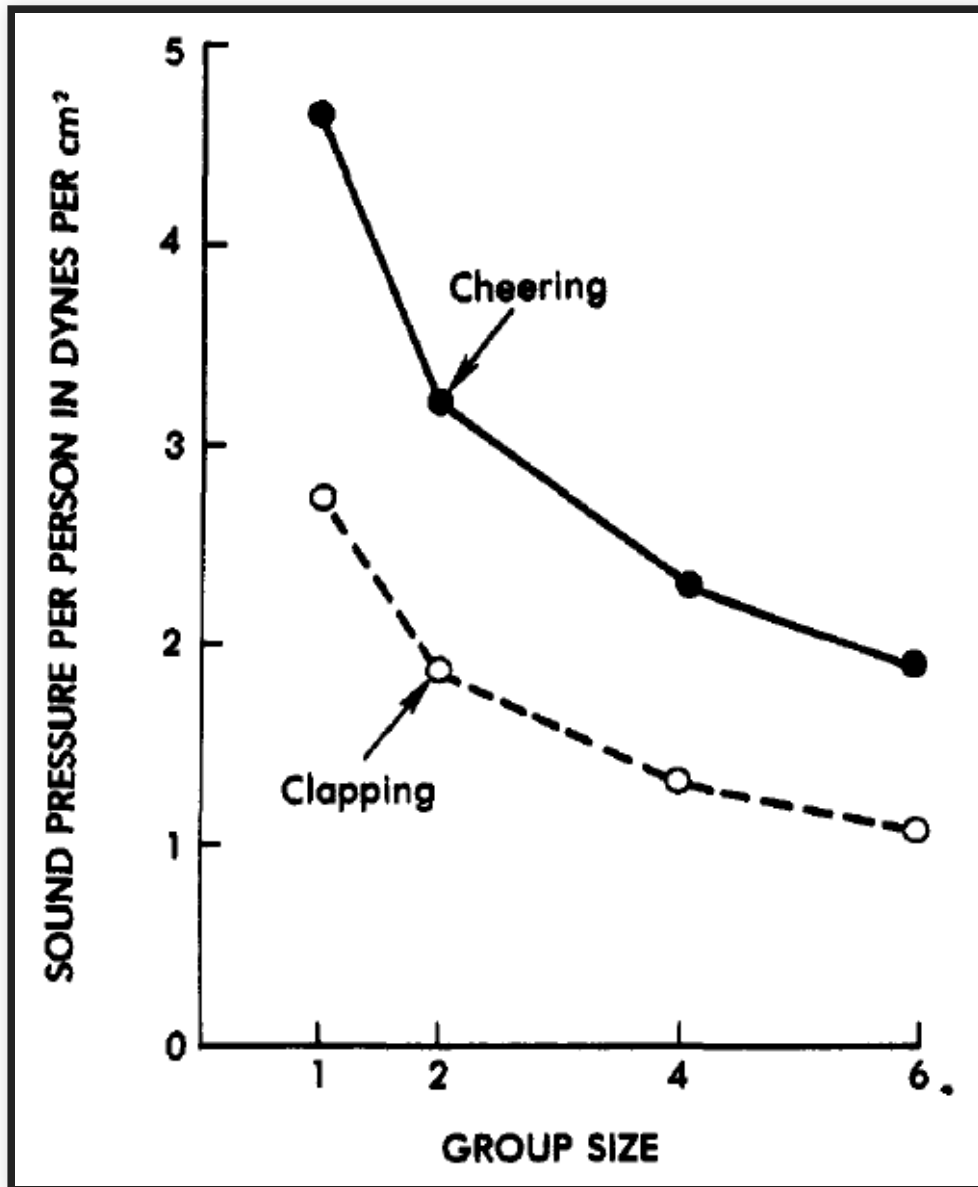
# MITIGATION STRATEGIES

- Diversity in team composition
- Culture of open conflicts
- Appoint devil's advocate in discussions, moderate and rotate speaker order, leaders hide opinions in discussions
- Involve outside experts
- Always request a second solution
- Monitoring and process measurement
- Agile techniques as planning poker, on-site customer

# TEAM ISSUES: SOCIAL LOAFING







Latane, Bibb, Kipling Williams, and Stephen Harkins. "[Many hands make light the work: The causes and consequences of social loafing.](#)" *Journal of personality and social psychology* 37.6 (1979): 822.

# SOCIAL LOAFING

- People exerting less effort within a group
- Reasons
  - Diffusion of responsibility
  - Motivation
  - Dispensability of effort / missing recognition
  - Avoid pulling everybody / "sucker effect"
  - Submaximal goal setting
- “Evaluation potential, expectations of co-worker performance, task meaningfulness, and culture had especially strong influence”

Karau, Steven J., and Kipling D. Williams. "[Social loafing: A meta-analytic review and theoretical integration.](#)"  
Journal of personality and social psychology 65.4 (1993): 681.

# MITIGATION STRATEGIES



# MITIGATION STRATEGIES

- Involve all team members, colocation
- Assign specific tasks with individual responsibility
  - Increase identifiability
  - Team contracts, measurement
- Provide choices in selecting tasks
- Promote involvement, challenge developers
- Reviews and feedback
- Team cohesion, team forming exercises
- Small teams

# RESPONSIBILITIES & BUY-IN

- Involve team members in decision making
- Assign responsibilities (ideally goals not tasks)
- Record decisions and commitments; make record available

# MOTIVATION

Autonomy \* Mastery \* Purpose

**THE NEW YORK TIMES TOP 10 BESTSELLER**

**'PROVOCATIVE AND FASCINATING'**  
**MALCOLM GLADWELL**

**'ENERGETIC'**  
**FINANCIAL TIMES**

**'INSPIRING'**  
**GUARDIAN**



**THE SURPRISING TRUTH  
ABOUT WHAT MOTIVATES US**

**DANIEL H. PINK**



# SUMMARY

- Team dysfunctions well studied
- Know the signs, know the interventions
- Small teams, crossfunctional teams
- Create awareness and accountability
- Further Readings:
  - Mantle and Lichy. Managing the Unmanageable. Addison-Wesley, 2013
  - DeMarco and Lister. Peopleware. 3rd Edition. Addison Wesley, 2013

