

# *Software Architecture and Techniques*

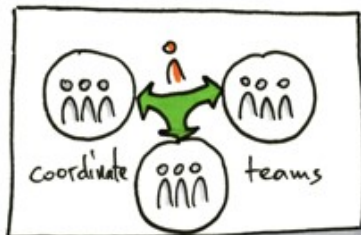
## Team and Technical Excellence for Architects

# Lecture Content

- *Why agile Architecture and Design?*
- *Evolution of Software Architecture over the last Decades*
- *What is agile Architecture?*
- *Agile Approaches with Scrum, XP, LeSS*
- *Refactoring*
- *Errors, Vulnerabilities, Smells*
- *Architecture of Components and Subsystems*
- *Quality Attributes of Software Architecture*
- *-ility Attributes*
- ***Architecture Documentation***
- ***Architecture Trends I***
- ***Architecture Trends II***
- ***Workshop***
- ***Team and Technical Excellence for Architects***

# Administration

- Please provide feedback to the content and form of the lecture
- Do you have any questions concerning examination or grading?



### lead technically



technology evangelist  
engineering practises  
technical spikes  
non-functional specs  
write code

### understand stakeholders

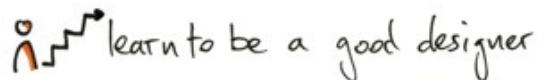
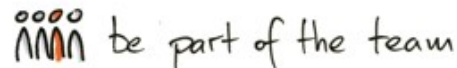


big picture  
talk to all stakeholders  
learn about all view points  
understand the user  
help the PO

### coach the team



architecture is team work  
coordinate  
pair program  
educate  
support SM

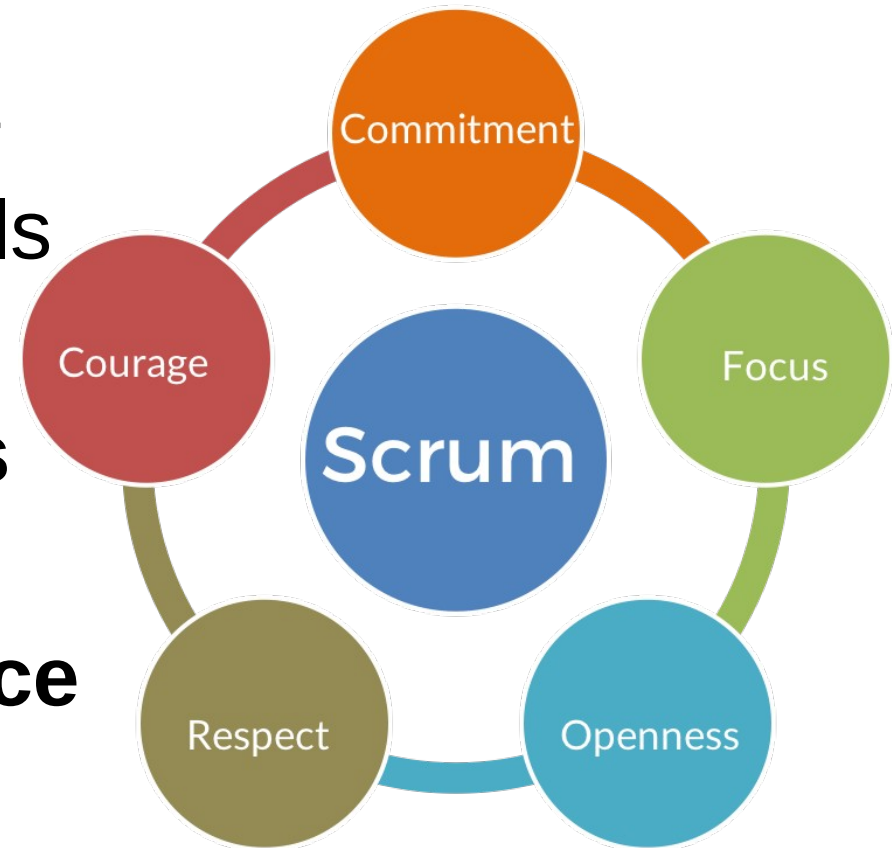


# Architect as a Teacher

- **Teaching** is a way to **learn**
  - Go deeper in the taught field because you must answer questions, write a blog, hold a workshop
  - Respect and like people you teach to
  - Understand what your teaching goals are and develop a strategy
  - Evaluate psychology to find out best learning practices for your target audience

# Architect as Tech Leader

- **Facilitate** creation of architecture, not enforcing it
- **Transition** architectural skills to team members
- Realize architectural **spikes** to learn and reduce risk
- **Mentor** in technical **guidance**



# Architect as a Coach

- A coach goal is to nurture developers and designers to become better than you are
- A coach is not the player playing the game, and only the player wins
- Often the coach goes away after tremendous successes and a new coach will bring fresh ideas

# Teaching Techniques

- Community of practice
- Pair programming
- Design workshop
- Coding dojo
- Design and code review
- Workshop and presentations



# Community of Practice

A community of practice (CoP) is, according to cognitive anthropologists Jean Lave and Etienne Wenger, a group of people who share an interest, a craft, or a profession



# Self-Teaching Techniques (1/2)

- Articles e.g. [Baeldung](#),
- Source code examples e.g. [StackOverflow](#)
- Java Magazines see HSLU library
- Technical blogs see [blog list](#)
- Youtube Channels
- Books see [books list](#)
- Open Source contributions

# Self-Teaching Techniques (2/2)

- Block a 4 hours session at least every two weeks to learn
- Have a budget to buy technical books
- Have a list of preferred blogs
- Ask questions in forums
- Be a member of a user group

# Coding Learning Techniques

- Pair programming
- Open source projects
- Refactoring sessions
- Mob programming
- IDE trainings
- Defect driven development

## Agile-Lean Practitioner

Applies Agile practices,  
lives Agile values

## Professional Coaching

Partnering with clients in a creative process that inspires their personal and professional potential (from ICF)

## Teaching

Instructing others in specific knowledge, skills and perspective

Sharing knowledge, skills & perspectives that foster the personal and professional growth of someone else

A neutral process holder who guides groups through processes that help them come to solutions and make decisions

## Mentoring

Technical expertise as a software craftsperson

## Technical Mastery

Expert at business-value driven innovation and product development

## Business Mastery

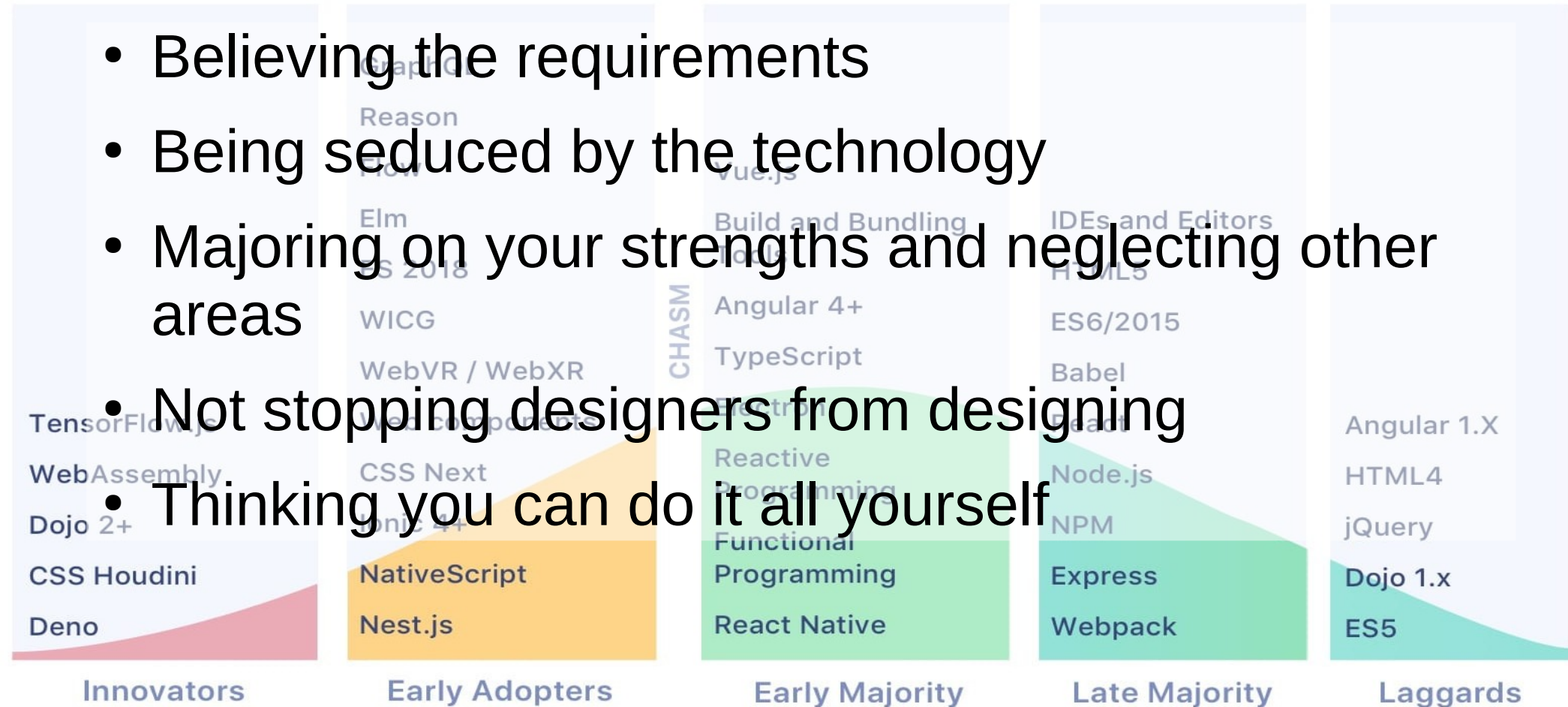
Expertise as an organizational development and change catalyst

## Transformation Mastery

## Facilitating

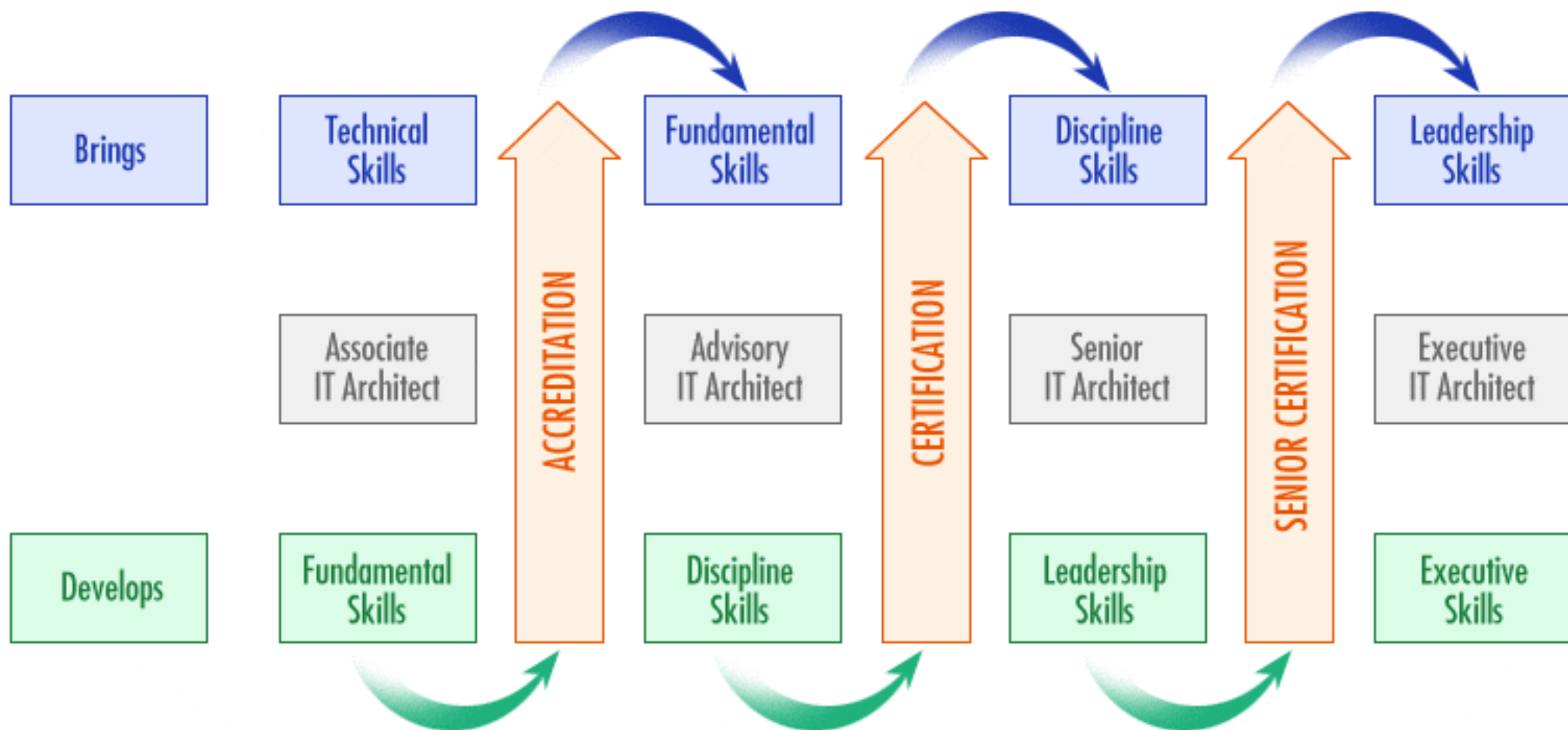
# Five Top Mistakes (IBM)

- Believing the requirements
- Being seduced by the technology
- Majoring on your strengths and neglecting other areas
- Not stopping designers from designing
- Thinking you can do it all yourself



# Ten Top Mistakes (Woods)

- Scoping Woes or Errors
- Not Casting Your (Stakeholder) Net Widely
- Focusing on Functions (Forgetting Qualities)
- Using Box and Line Descriptions (Too Simple Diagrams)
- Forgetting that *it Needs to be Built*
- Lack of Platform Precision
- Performance Assumptions
- Do-It-Yourself Security
- Lack of Disaster Recovery
- No Back-out Plan





Positive  
Inclusive  
Promotes Improvement

# THE AGILE FLUENCY™ MODEL

CHART YOUR AGILE PATHWAY

Fluency: Routine, Skillful Ease  
Comes From Investment In Learning

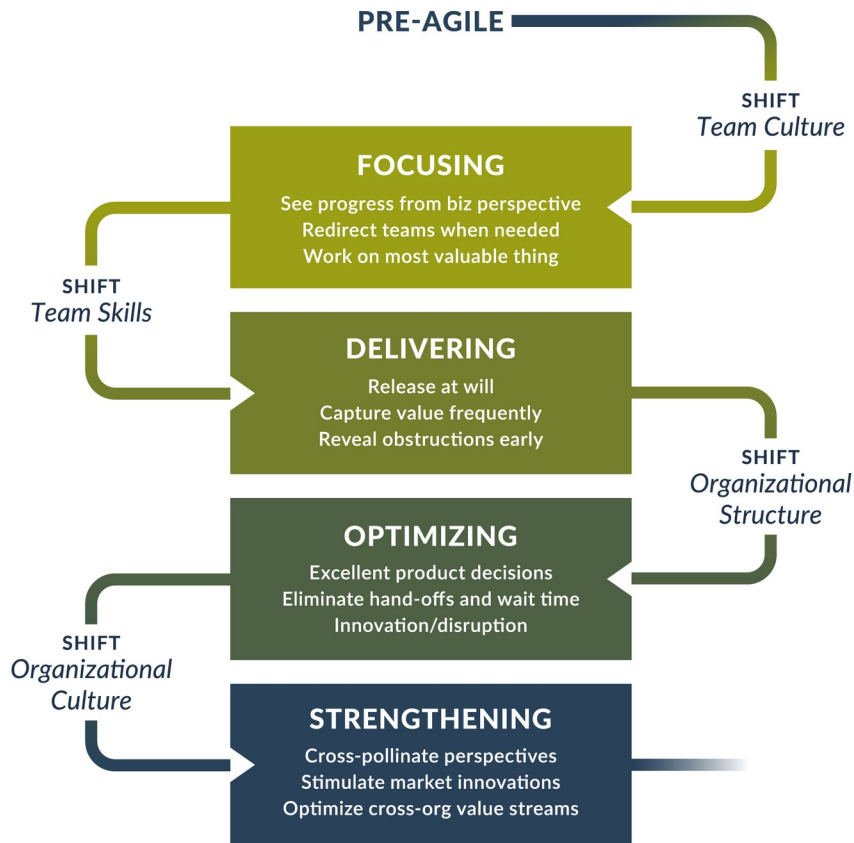
SWAT Goals

**INVEST IN:**  
Productivity dip  
DevOps, UX, etc. in team  
Technical training & mentoring

**AGILE SUSTAINABILITY: +3-24 MO.**  
Extreme Programming  
DevOps Movement

**INVEST IN:**  
Inventing new practices  
Cross-organization focus

**AGILE'S FUTURE**  
Complexity Theory  
Organization Design Theory  
Alternative Governance Structures



**INVEST IN:**  
Full-time team members  
Team workspace  
Business representation  
Team coaching  
Management training

**AGILE FUNDAMENTALS: 2-6 MO.**  
Basic Scrum  
Kanban

**INVEST IN:**  
Market focus  
Business expertise in team  
Team business ownership  
Management coaching

**AGILE'S PROMISE: +1-5 YR.**  
Lean Startup  
Lean Software Development  
Design Thinking  
Beyond Budgeting

**AGILE FLUENCY PROJECT**

[agilefluency.org](http://agilefluency.org)

Copyright 2012-2018 James Shore and Diana Larsen.  
"Agile Fluency" is a trademark of James Shore and Diana Larsen.  
You may reproduce this diagram in any form so long as this notice  
is preserved.

# CULTURE = "How we do things around here to succeed."

[=] "The Reengineering Alternative."  
William Schneider



# Agile HR - Also for Architects

**Collaborative networks** over hierarchical structures

→ Self-selecting teams, mob programming, remote-only teams

**Transparency** over secrecy

→ Access to all financial data

**Adaptability** over prescriptiveness

→ Rolling budgets

**Inspiration and engagement** over management and retention

→ Why are we working on this product?

**Intrinsic motivation** over extrinsic rewards

→ No bonuses, no MBO, evaluations by peers, 360 evaluation

**Ambition** over obligation

→ Curriculum based design,

# Agile Culture

- **Customer** Centric *instead of Plan Centric*
- **Small** Teams *instead of Departments*
- **Network** Connections *instead of Hierarchical Organizations*

# Self-Organizing Teams

Setting Overall Direction	Management Responsibilities			
Designing the Team and Its Context				
Monitoring and Managing Work Processes		Team Responsibilities		
Executing the Task				
	Manager-Led Team	Self-Managing (Self-Organizing) Team	Self-Designing Team	Self-Governing Team

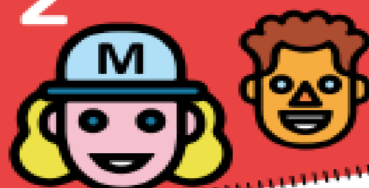
1



## Tell

I will tell them

2



## Sell

I will try and  
sell it to them

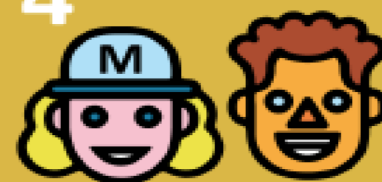
3



## Consult

I will consult  
and then decide

4



## Agree

We will agree  
together

5



## Advise

I will advise but  
they decide

6



## Inquire

I will inquire  
after they decide

7



## Delegate

I will fully  
delegate

# DELEGATION POKER

These cards are part of the Management 3.0 materials. They represent the 7 delegation levels for empowering organizations. You can find a description of their use at:

[www.management30.com/delegation-poker](http://www.management30.com/delegation-poker)

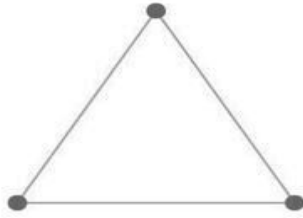
**MANAGEMENT 3.0**  
CHANGE AND INNOVATION PRACTICES

# True North - Scrum Values

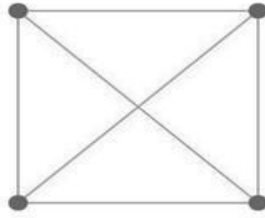




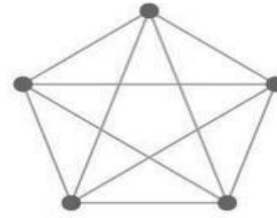
# Small Teams



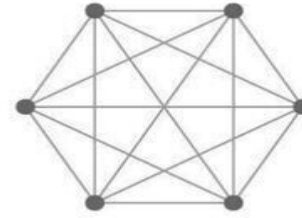
3 people, 3 lines



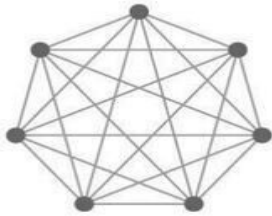
4 people, 6 lines



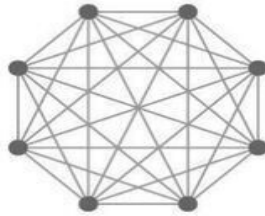
5 people, 10 lines



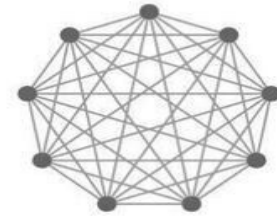
6 people, 15 lines



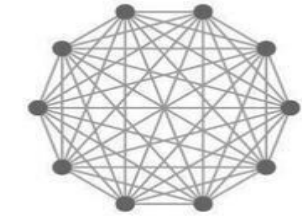
7 people, 21 lines



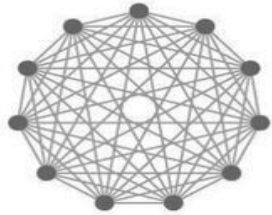
8 people, 28 lines



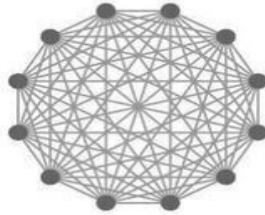
9 people, 36 lines



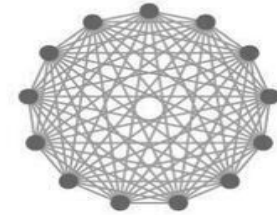
10 people, 45 lines



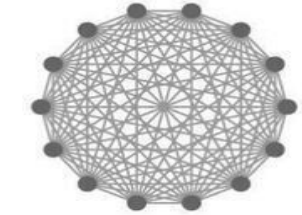
11 people, 55 lines



12 people, 66 lines



13 people, 78 lines



14 people, 91 lines



# Reflections (1/2)

Any **organization** that **designs** a system will produce a design whose structure is a **copy** of the organization's communication structure.

— Melvin E. Conway, 1967

- *Agile approach is a way to break Conway's Law*

# Reflections (2/2)

Do not fight to change a **culture**. Change the organization's **structure** and **processes**. The culture will naturally evolve.

Do not force common **goals** and **accountability**. Require **pair** and **mob programming** for example.

# Exercises (1/2)

- Read article “High Performance Teams: The Foundations”
- Draw your vision of agile architecture
- Write down why you want to be an architect – or why you do not want to be one -
- Reflect why software is strategic for the company you are working for – or will work for, or the company your parents are working for -

# Exercises (2/2)

- Themes brought by attendees