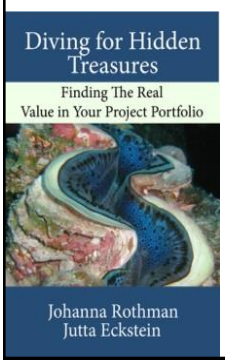
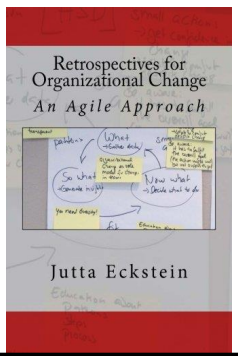
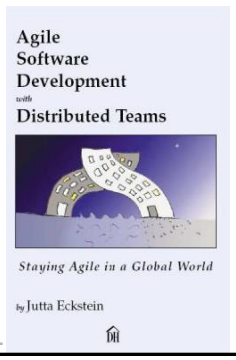
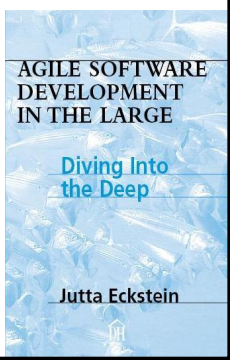
Agile Software Development with Distributed Teams

Jutta Eckstein
Jutta@JEckstein.com
www.JEckstein.com









1

Agenda

- Setting the context
- Self-organizing teams supported by business people
- Deliver working software frequently
- Building personal relationships through communication and trust
- Bridging cultural differences by focusing on similarities

Setting the Context

Agility based on Value System

Agile Manifesto:

Individuals and interactions
over processes and tools

Working software
over comprehensive documentation

Customer collaboration
over contract negotiation

Responding to change
over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Source: http://agilemanifesto.org

Agile Principles

- **Value system is based on the following principles:**
  - Early and continuous delivery of valuable software
  - Welcome changing requirements
  - Deliver working software frequently
  - Business people and developers work together
  - Trust motivated individuals
  - Face-to-face conversation
  - Working software is the primary measure of progress
  - Promote sustainable development
  - Technical excellence and good design
  - Simplicity is essential
  - Self-organizing teams
  - Team reflection and adjustment

Self-Organizing Teams supported by Business and Technical People

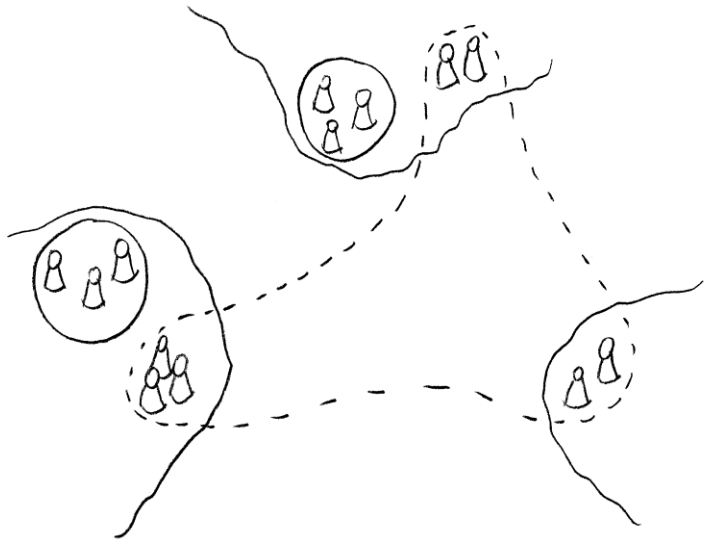
Building Teams

- **Avoid the typical structure**
  - – According activities and know-how
    - Analysis in Denmark, UI in India, middleware in Ireland...
    - Enforcement of incompatible interfaces
    - Achievement of business value only at the end of the project
- **Instead structure along features**
  - – For ensuring the business value and the customer‘s advantage
  - – Features shouldn‘t be split across teams
    - The feature provides a joint goal and thus enforces team spirit

Self-Responsible Feature Team

- **Comprehends (or gains) all necessary roles & know-how**
- **Ensures to complete valuable stories in an iteration**

Distributed and Dispersed Teams

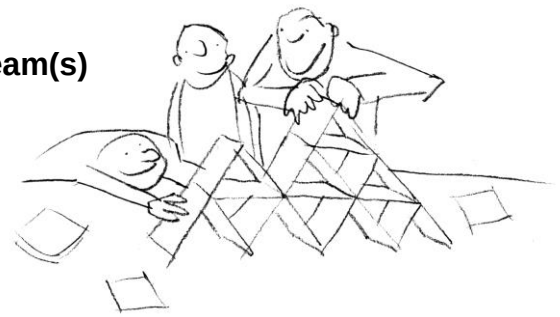


Supporting Whole Teams

- Every feature team needs the product owner‘s support
- One product owner might not be enough

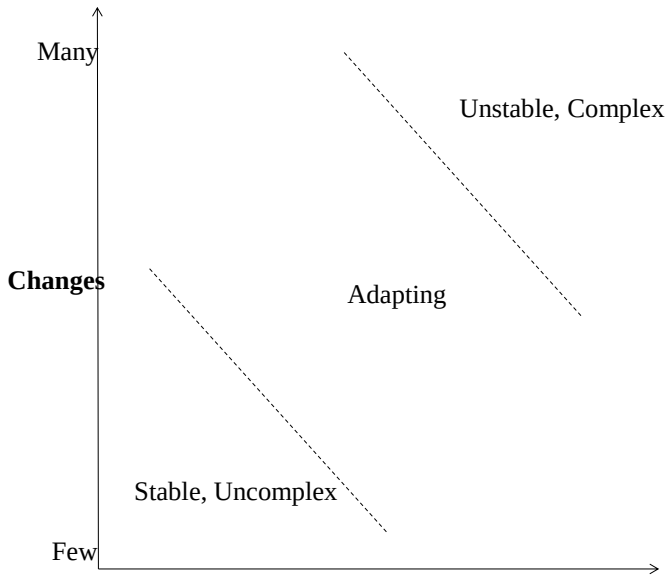
Supporting Technical Perspective

- **Depending on technology and project size**
- **One architect per feature team**
- **Or 1-x architects support several feature teams**
  - But: For ensuring simplicity there is always one chief architect!
- **Technical service team(s)**



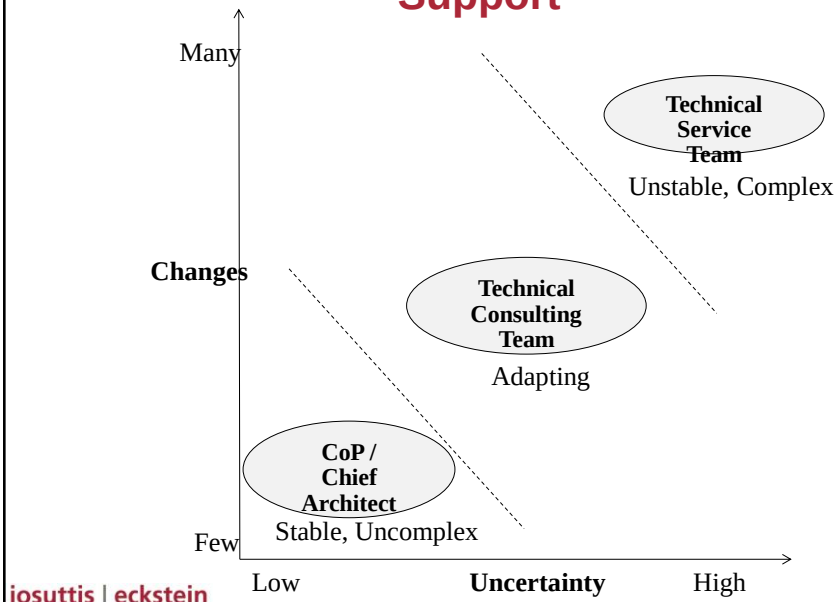
josuttis | eckstein
IT communication

Complexity of Systems



josuttis | eckstein
IT communication

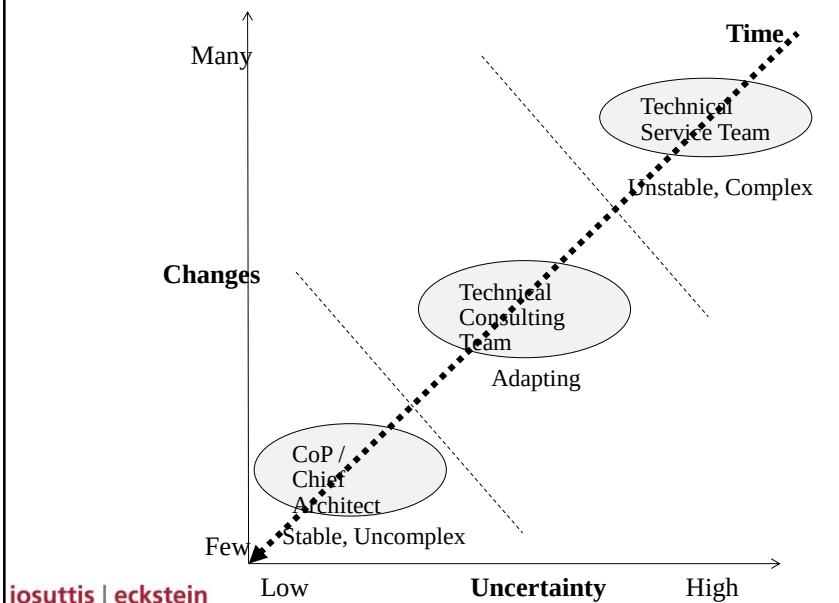
Different Models for Architectural Support



josuttis | eckstein
IT communication

13

Complexity Decreases over Time



josuttis | eckstein
IT communication

14

Deliver Working Software Frequently

Development Cycles

- **No need to prolong cycles**
  - Frequent feedback for steering in the right direction
  - Short cycles to reduce all risks
- **Two-week iterations have been proven**
  - Balance feature accomplishment with risk reduction
  - Ensure delivery at the end of the iteration
- **Same heartbeat across all sites**
  - Holidays can require some adaptation

Iteration Review and Planning

- **Ensure iteration turnover is in the mid of the week**
- **For dispersed teams:**
  - Get together in person from time to time
  - Use different communication media
    - Phone, webcam, NetMeeting (or the like), video, ...

Iteration Planning

- **Pre-Planning in the middle of the iteration**
  - Product owners and architects decide on features for diverse teams
- **Each subteam plans individually**
  - Guided by coach and customer (product owner)
  - Outcomes are visible and accessible at prominent place

Integration and Build

- **Before spreading over several sites**
  - Ensure integration and build works at one site
  - The later you are addressing these problems the more difficult they get
- **Don‘t underestimate the complexity and required effort**
  - Ensure you have full-time people being responsible
  - Assign 10% of your development effort

Release Iteration

- **If a release iteration (sprint) is required for a bigger delivery**
  - Each team who delivers to the release sends a representative (in person) to the integration site
  - Integration sites alternate

Building Personal Relationships through Communication and Trust

Trust

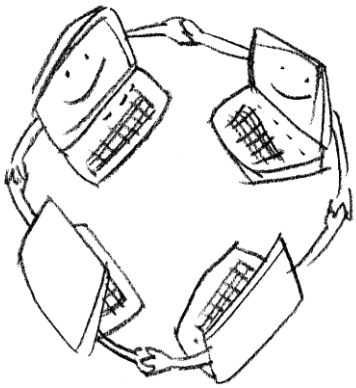
Trust always goes ahead:

By giving trust you're gaining trust.

Tom DeMarco

Dispersed Daily Scrum

- **Ensure mutual respect**



Trust needs Touch

- **Face-to-face should always be preferred**
  - Frequency and duration depend on distance
- **Meet face-to-face from time to time**
  - More and longer at the beginning
    less frequent after a while
  - Couple of days every week
  - Regular for specific events
  - Rotating people over sites

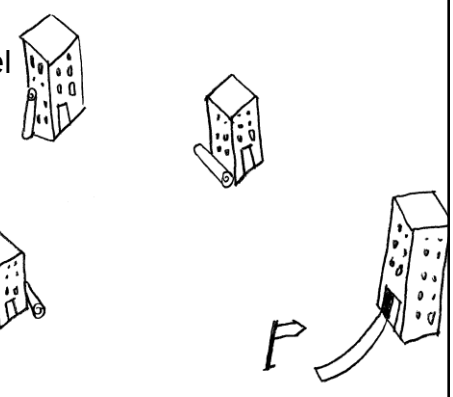
Face-to-Face has a Price

You will pay the costs of a face-to-face meeting, regardless of whether you have one or not.

Ken Pugh

Communication and Trust

- **Trust is based on mutual respect**
- **Different meeting locations**
  - Change who will be the host and who needs to travel
- **Pay attention to the vocabulary**
  - Nightly build
  - Morning roll call
  - Remote site

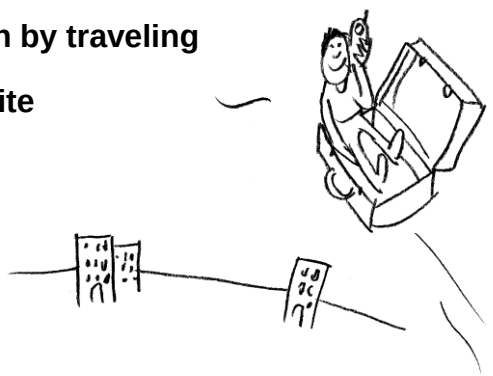


Trust Threshold

- **Trust threshold by ignoring necessary proximity**
  - A trusted relationship typically lasts 8-12 weeks
- **Threshold is close, if**
  - Communication, i.e. emails are misunderstood
- **Trust can be broken in an instant**
  - It is harder to re-establish trust than to establish it at first

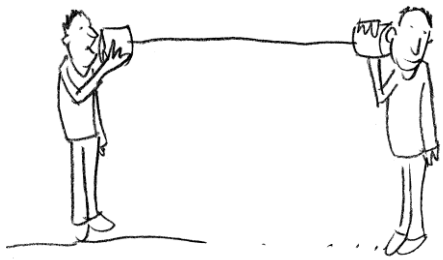
Keep the Sites in Touch

- **Ensure communication by traveling**
- **Ambassador at each site**



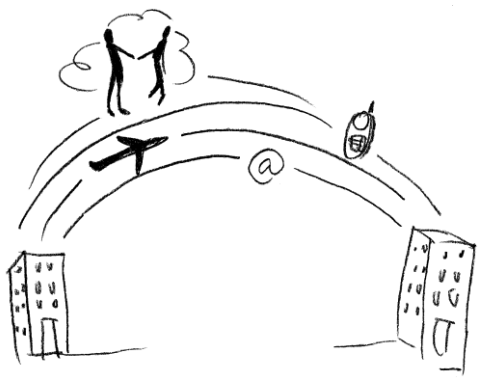
Communication Channels

- Different sensory modalities require different channels
- Different needs
- Balance media



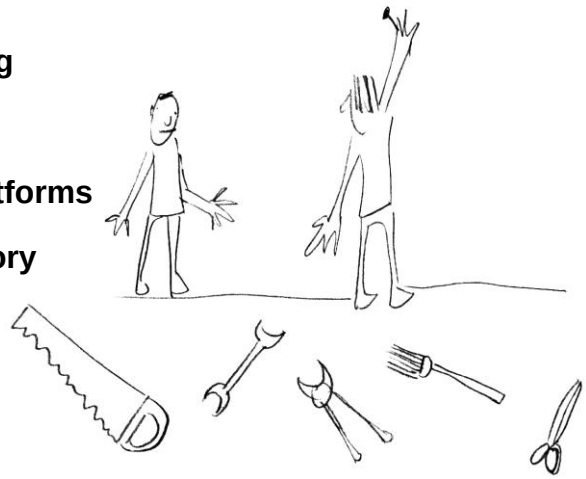
Communication Channels

- Direct connections
- Synchronous ←→ Asynchronous
- Video ← → Audio



Tools

- Instant messaging
- Email
- Collaboration platforms
- Common repository



Bridging Cultural Differences by focusing on Similarities

What defines Culture?

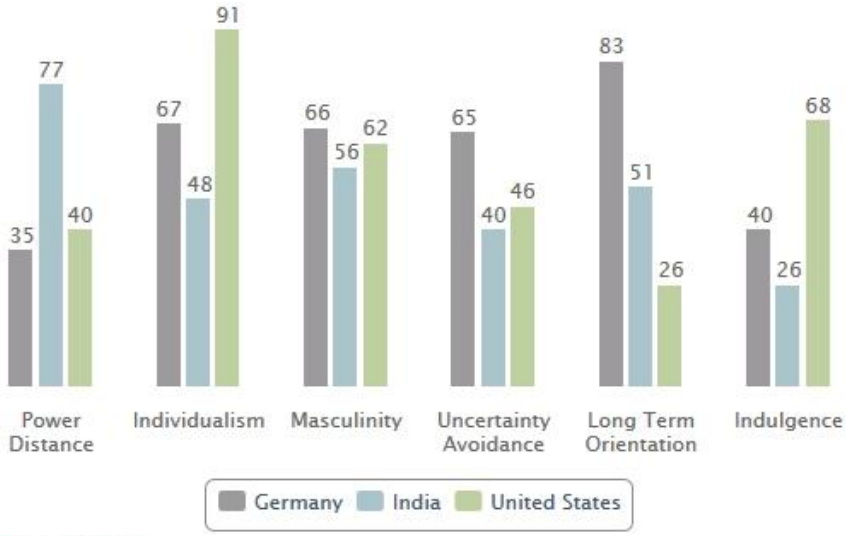
- **Geography**
- **Language**
- **Strategies**
- **Politics**
- **Values**
- **History**

Some Cultures are Closer than Others…

- **Geert Hofstede™ six cultural dimensions**
  - Power Distance Index (PDI)
    - Acceptance and expectation that power is distributed unequally
  - Individualism vs. Collectivism (IDV)
    - Strong self-responsibility vs. caring societies
  - Masculinity vs. Femininity (MAS)
    - Very assertive and competitive vs. modest and caring
  - Uncertainty Avoidance Index (UAI)
    - Strict laws and rules vs. as few rules as possible
  - Long Term vs. Short Term Orientation (LTO)
    - Thrift and preserverance vs. social obligation (protecting one‘s face and respect for tradition)
  - Indulgence vs. Restraint (IVR)
    - Enjoying life vs. Strict social norms

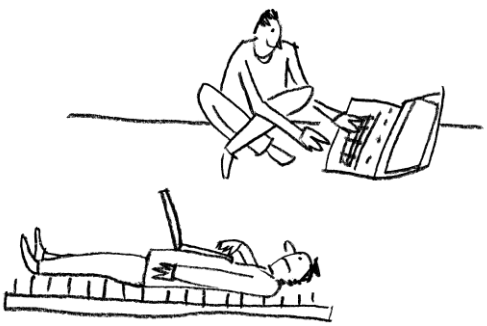
Comparing Cultures:

Source: http://www.geert-hofstede.com/



Who defines Culture?

- **You have to deal with different cultures everywhere**
  - Nation
  - Company
  - Divisions in company
  - Groups
  - Family
  - Personal

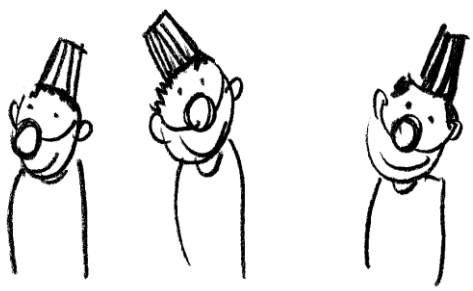


Creating a Community via Retrospectives

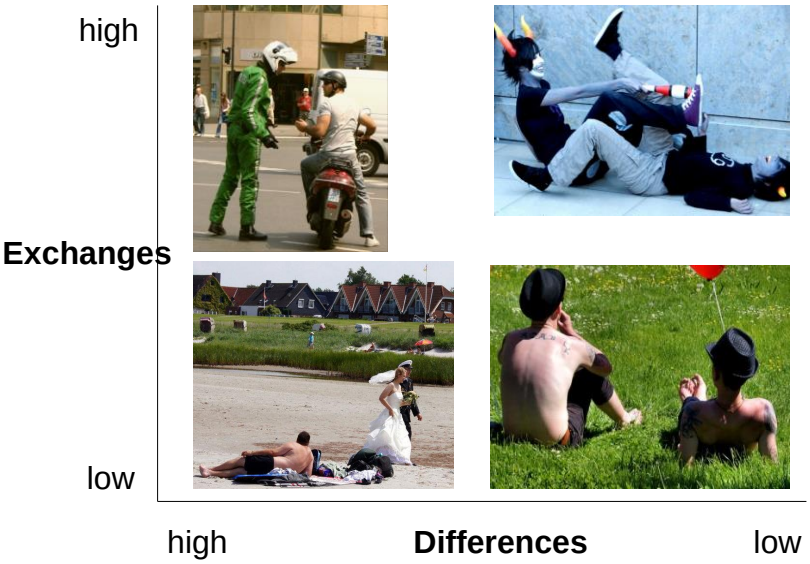
- **Regularly**
- **Different kinds of retrosepctives: team, site, cross-team**
- **Virtual retrospectives**
  - A new example tool:
    - http://retrium.com

Focus on Cultural Similarities…

- **… allows creating joint culture in terms of**
  - Language
  - Strategies
  - Politics
  - Values
  - History



Changing Differences & Exchanges



High Exchanges & High Differences

Self-Organizing:

- **Decrease exchanges by:**
  - Reflection
  - Separation
  - Rest
- **Decrease differences by:**
  - Identifying similarities
  - Creating common group stories

High Exchanges & Low Differences

Reinforcing:

- **Decrease exchanges by:**
  - Reflection
  - Separation
  - Rest
- **Increase differences by:**
  - Identifying differences
  - Asking for opinions

Low Exchanges & Low Differences

Belonging:

- **Increase exchanges by:**
  - Using questions and stories
- **Increase differences by:**
  - Identifying differences
  - Asking for opinions

Low Exchanges & High Differences

Uncoupling:

- **Increase exchanges by:**
  - Using questions and stories
- **Decrease differences by:**
  - Identifying similarities
  - Creating common group stories

Changing Differences & Exchanges

high

Exchanges

low

- **Decrease exchanges by:**
  - Reflection
  - Separation
  - Rest

- **Decrease differences by:**
  - Identifying similarities
  - Creating common group stories
- **Increase differences by:**
  - Identifying differences
  - Asking for opinions
- **Increase exchanges by:**
  - Using questions and st

high **Differences** low

Lessons Learned

Summary

- **Agility provides a culture in its own**
- **Focusing on commonalities enables us**
  - To bridge the distance
- **Diversification allows us**
  - To have different perspectives on the same thing
- **Thus,**
  - There is a positive flip side to distributed development

Many Thanks!

Jutta Eckstein
Jutta@JEckstein.com, www.JEckstein.com

www.distributed-teams.com