Viewing Software Development as Learning

Allan Kelly -
http://www.allankelly.net

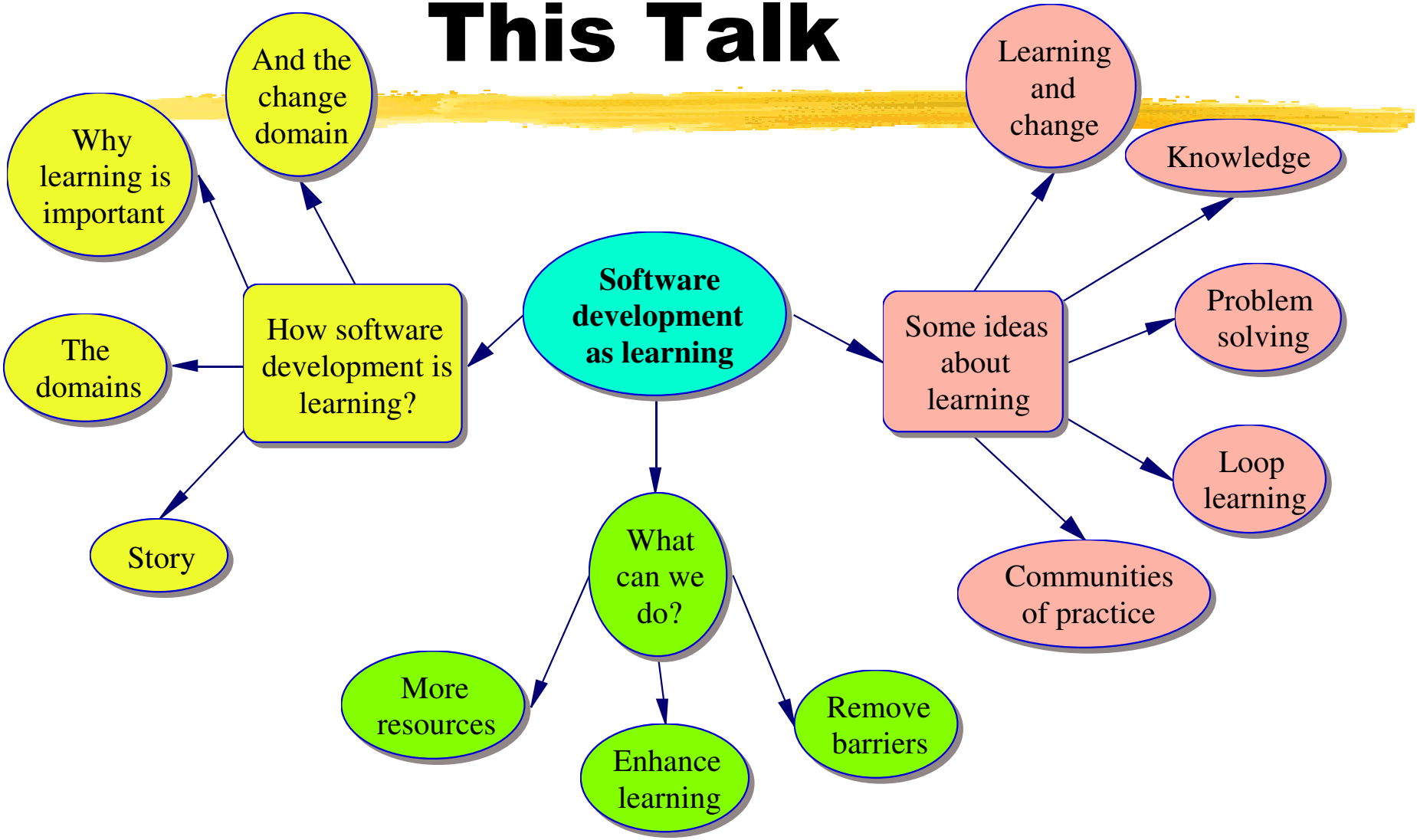
ACCU Conference

Oxford, April 2005

A story…

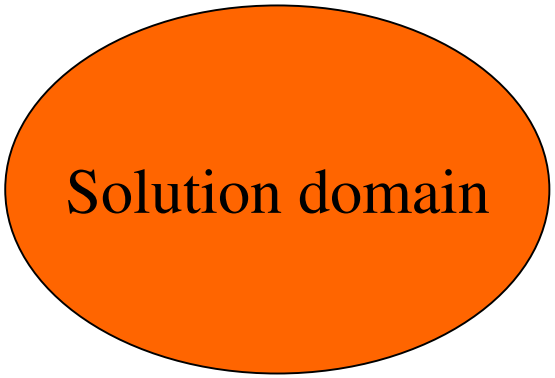
- My first job… and the electricity company
  - Learn about electricity trading markets
  - Learn about the computer model
  - Learn the code of the computer model
  - Learn Borland Pascal
  - …

This Talk


Software development as learning
How software development is learning?
Why learning is important
And the change domain
The domains
Story
Some ideas about learning
Learning and change
Knowledge
Problem solving
Loop learning
Communities of practice
What can we do?
More resources
Enhance learning
Remove barriers

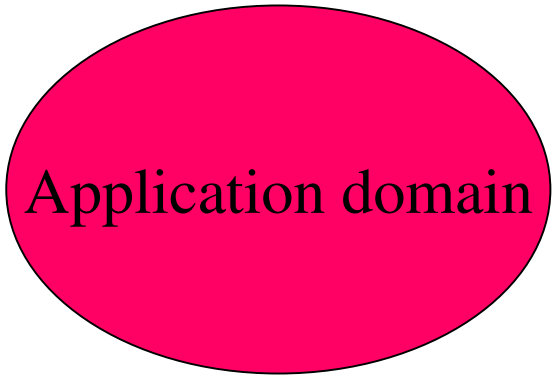
3 Learning Domains
#1 - *Solution domain*

- What we are at this conference for
  - Technology
  - C++, Java
  - Unix, Windows
  - UML, OO
  - ...



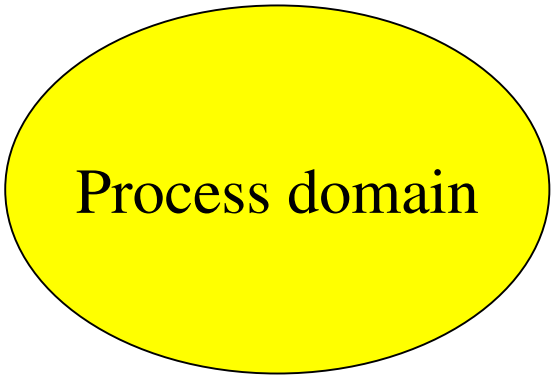
3 Domains of learning
#2 - *Application domain*

- What the application does
- Your customers business e.g.
  - Insurance,
  - Banking
  - Telecoms
  - ...



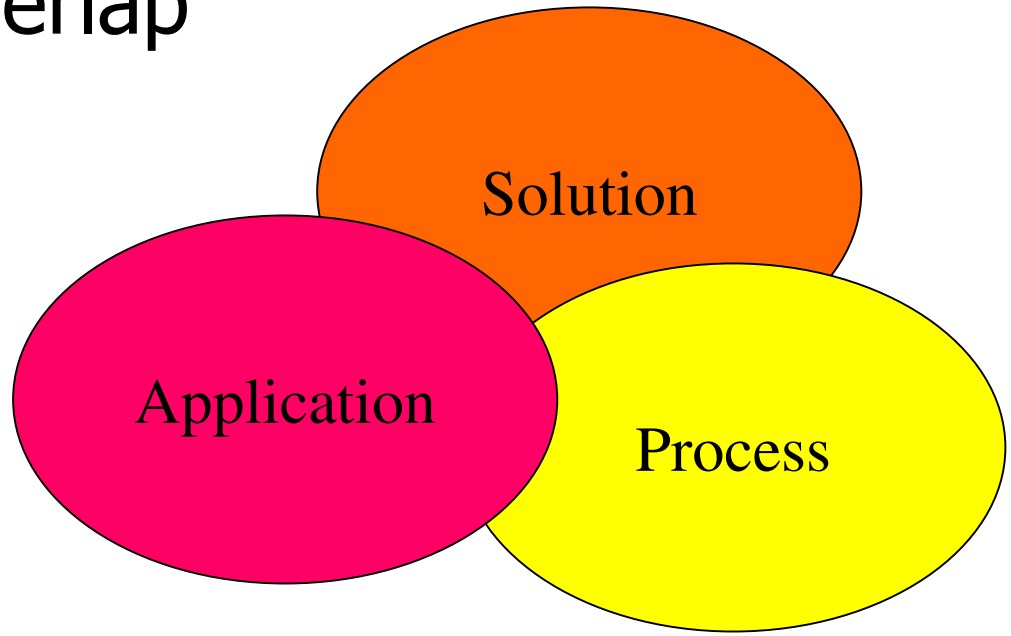
3 Domains of learning
#3 - *Process Domain*

- How do we build software?
- What do we do?
- Who does what?
  - Bug handling
  - Feature requests
  - Product management
  - ...



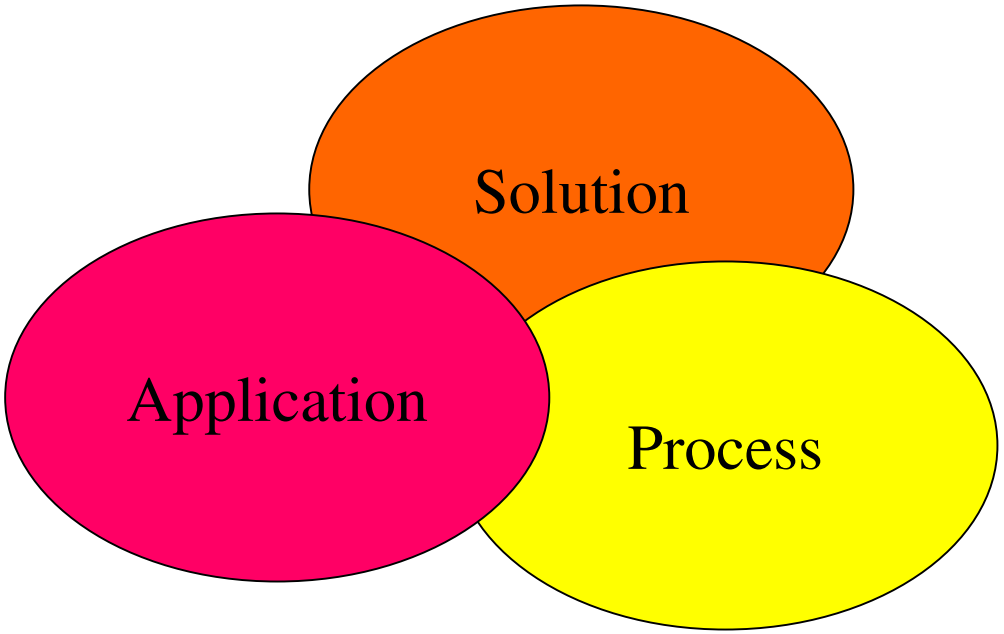
3 Domains of learning

- Domains overlap
- Interact



3 Domains embedded

- Need knowledge from all 3 domains to create software
- Software embeds knowledge from all domains



Why is learning important to software developers?

- Knowledge is the basis for commercial competition
  - Software is the embodiment of much knowledge
- Learning continues in domains when software is done
  - Software must be capable of change

Why is learning important to software developers?

- Developing software causes learning to happen
  - for developers
  - for customer
- We need to learn in all 3 domains

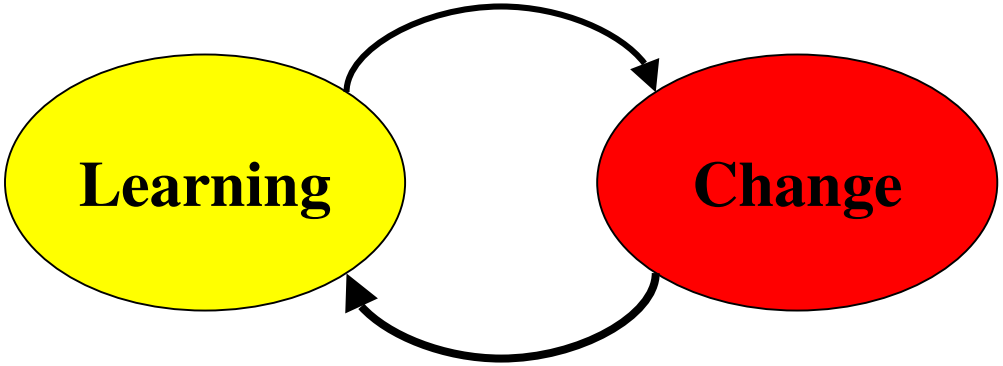
Why is learning important to software developers?

- The better we are at learning...
  - the better our software will be
  - the quicker we will developer
  - better serve our customers

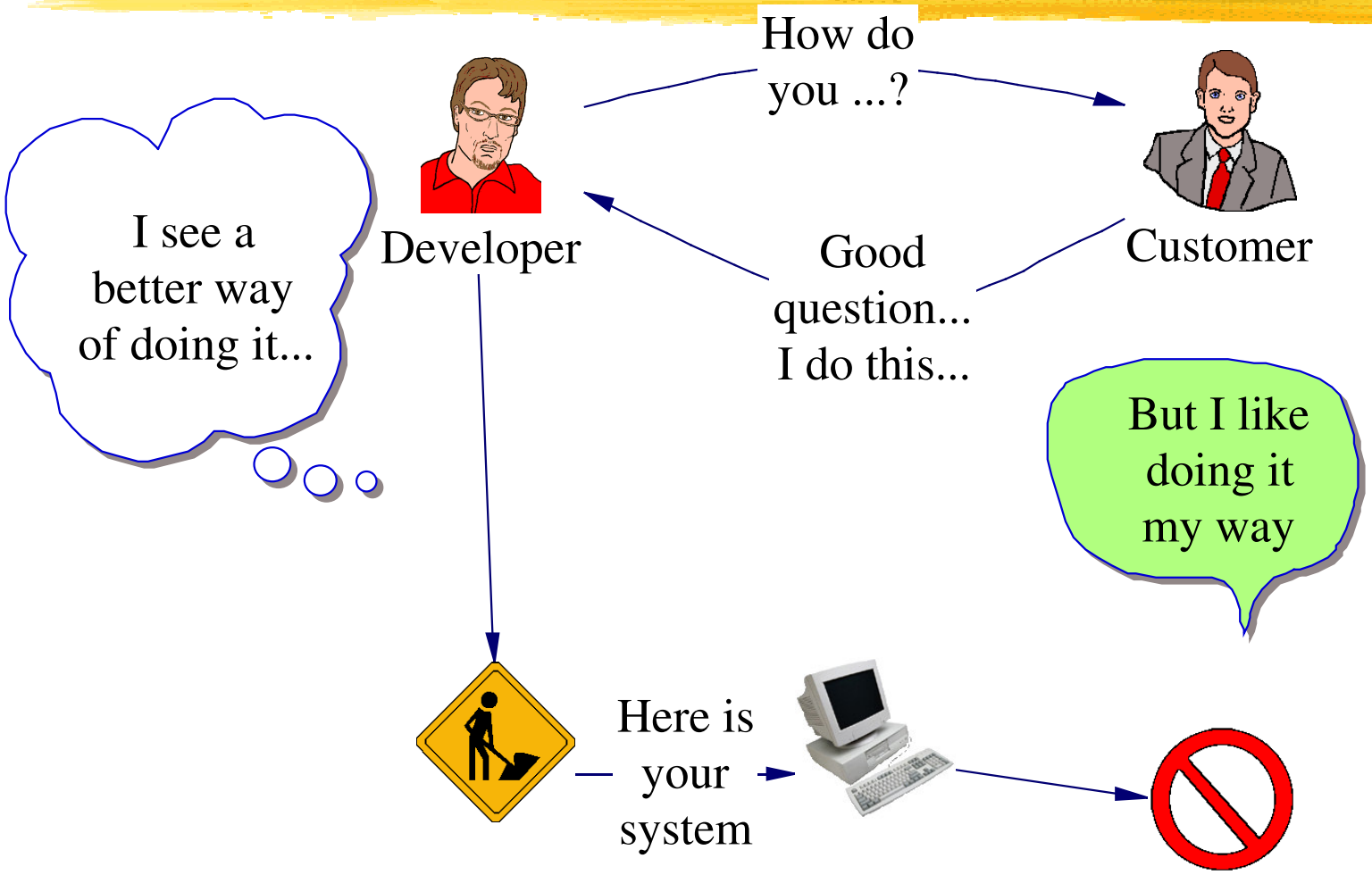
A few theories

Learning & Change

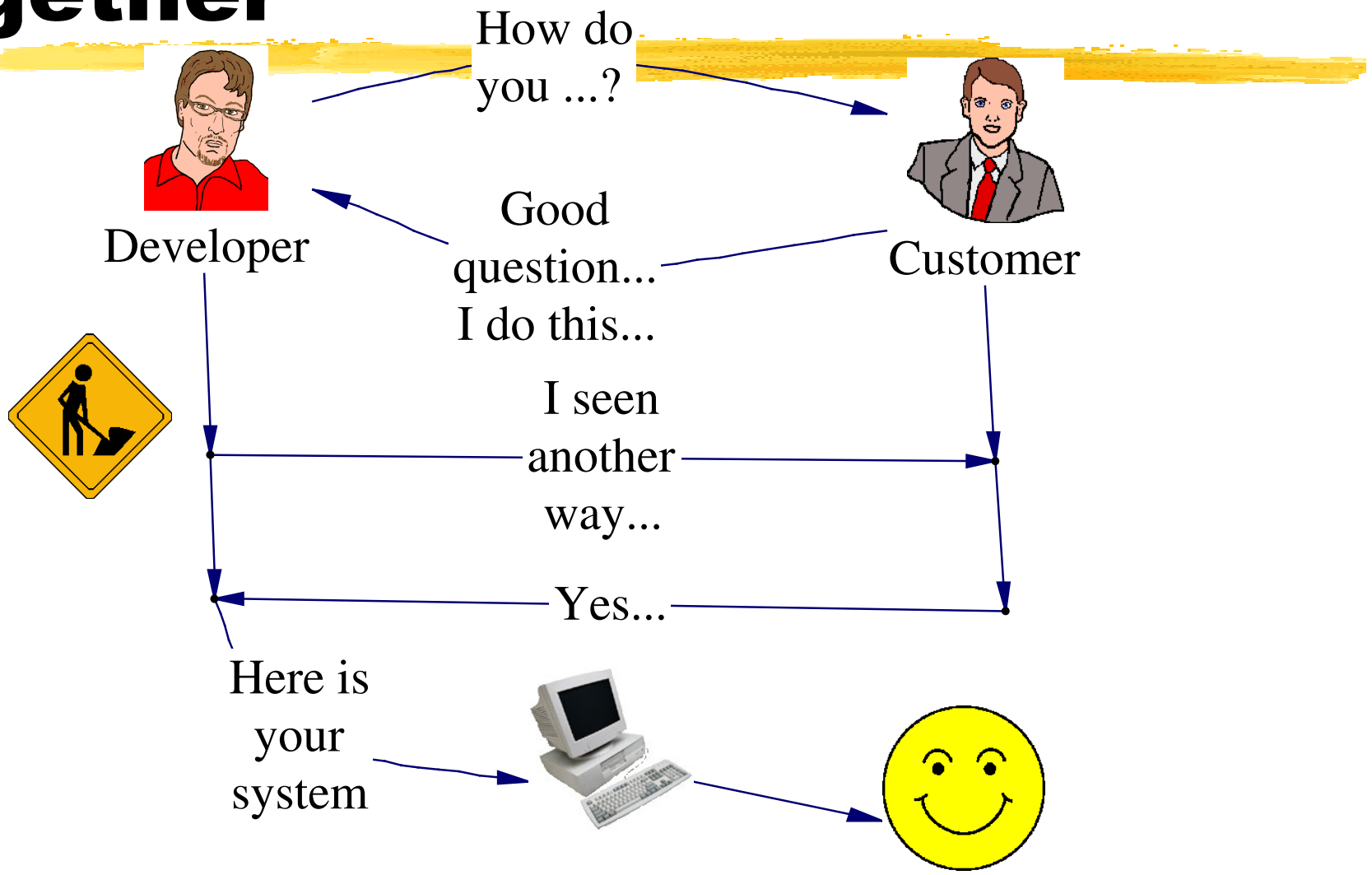
- Learning creates change
  - Without change there is no learning
- Change creates learning
  - Without learning there is no change



Don’t separate Learning & Change

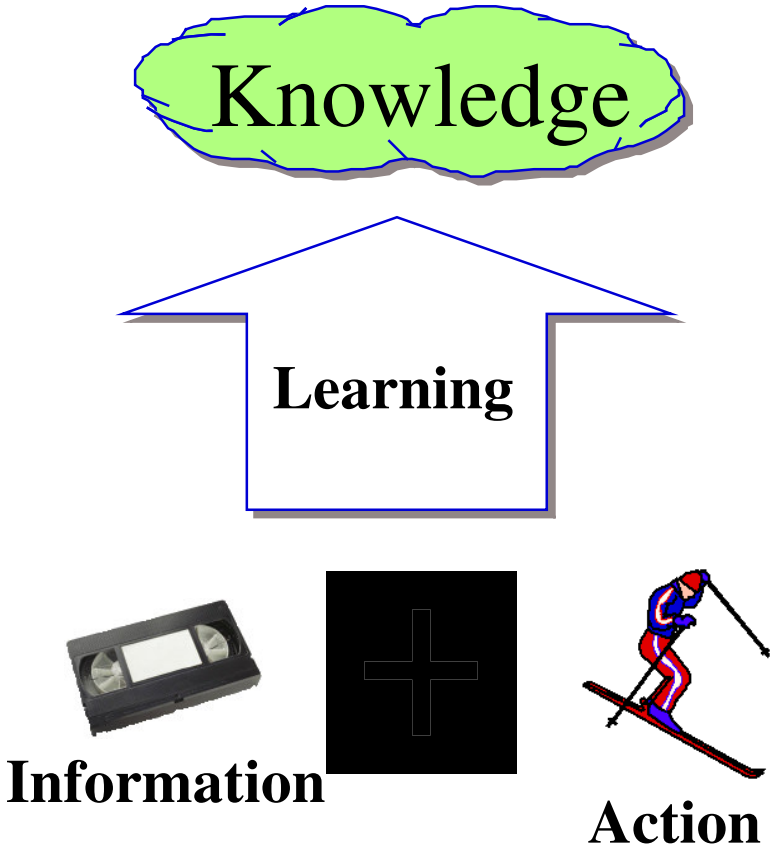


Learning & Change together



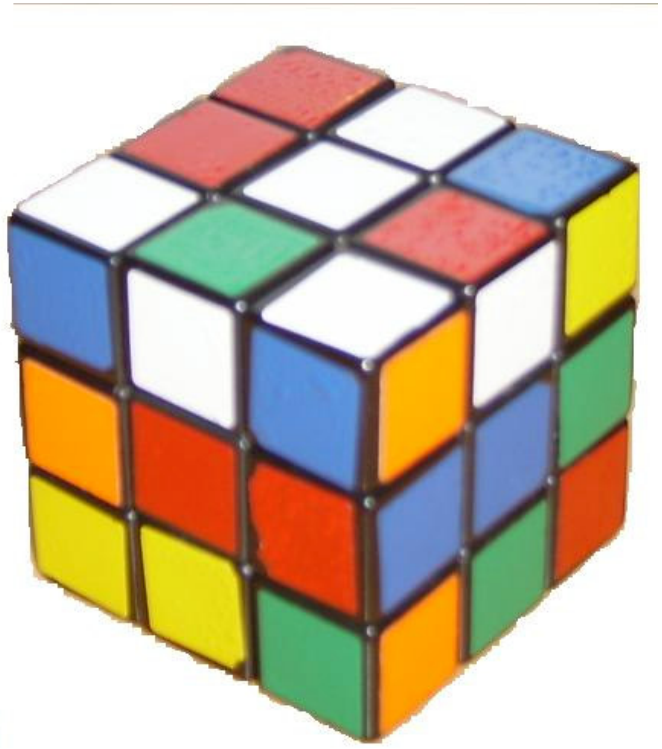
Knowledge & Learning

- Information is not knowledge
- Knowledge requires action
- Learning is knowledge creation



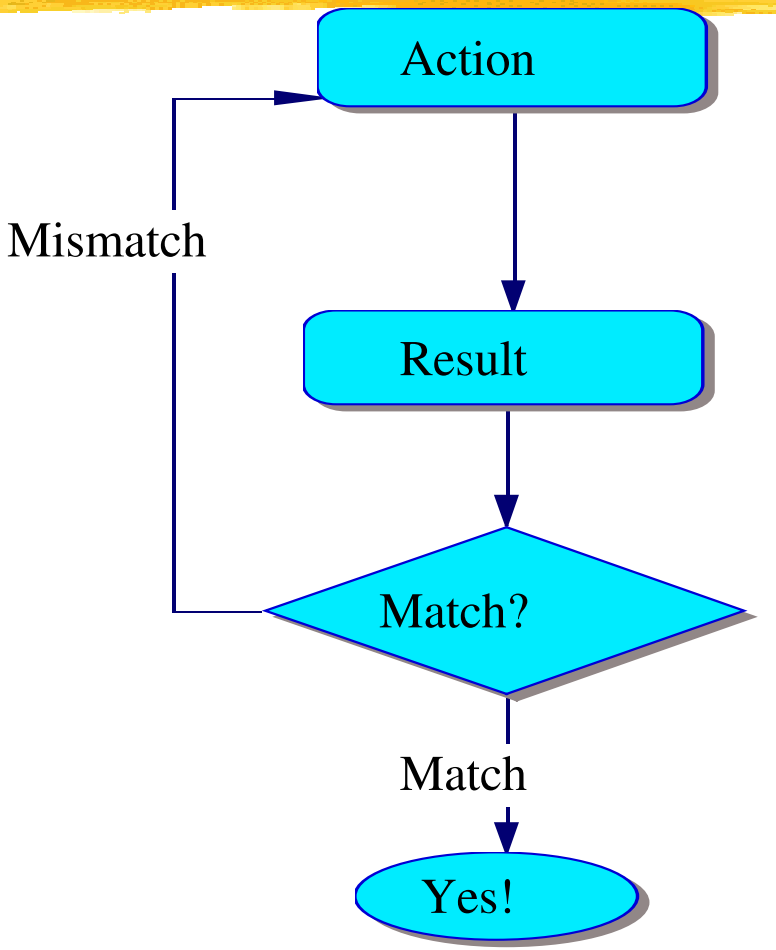
Problem solving is Learning

- How do we solve a problem?
  - ...
- Software Development is problem solving



Single loop learning

1. Do something
2. Something happens
3. Did it work?
   - No! - Do something, goto 1
   - Yes! - Great, Game over

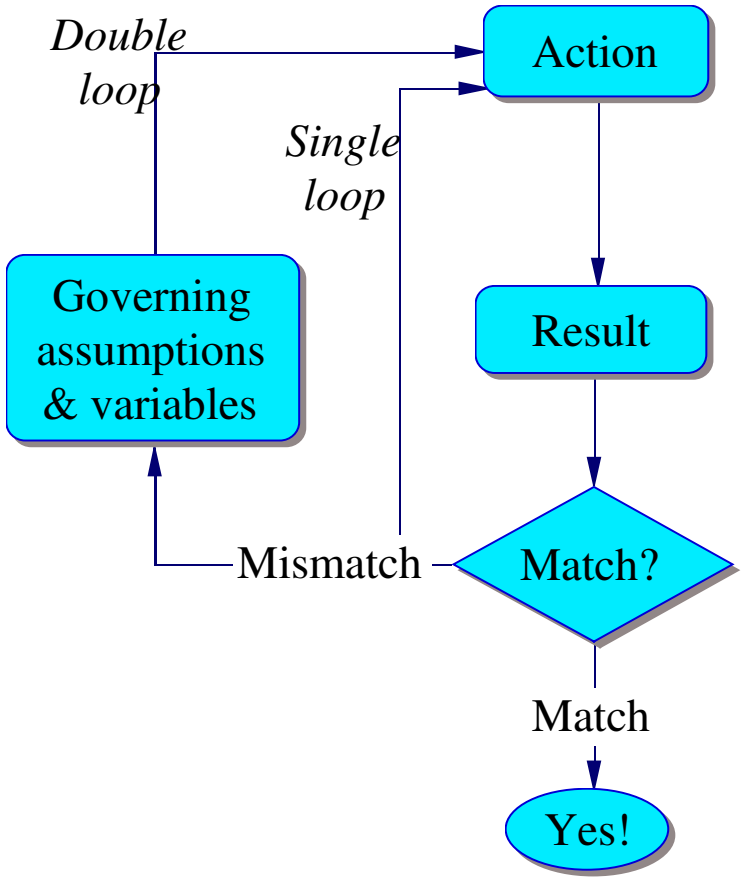


Problem with Single Loop

- Can lead to local solutions
  - "My boss gets upset if I take longer than my estimates... so I pad my estimates"
  - "My workers always pad their estimates so I cut 30% off"
- Deals with the immediate problem
  - Not the underlying problem

Double loop learning

- Look deeper
- What assumptions are we making?
- What conditions are we operating under?
- Why is this happening?



Learning together

- Learning with others has benefits
  - Discussion is creative
  - Saying it forces us to understand it
  - People bring new knowledge
  - Others point out our assumptions
  - Others see the flaws in our arguments

Communities of Practice

- People doing similar work share experiences
- Share knowledge
- Generates new knowledge
- The ACCU is a community of practice

Enhancing learning

- If we can enhance learning, we can improve our software development
- People are learning creatures
  - Our ability to learn marks us as different from other animals
  - Greater learning ability
  - Can ask questions

Can we enhance learning?

- Ideas please

Remove the barriers

- We are learning creatures
  - So why do we not learn?
- There are barriers to learning
  - Some are blocks on you learning
  - Some are blocks you create on others

Remove barriers

- Awareness
  - Is there awareness of the problem?
  - Does someone protect you?
  - Does someone always step in to fix it?
    - Is this person you?
    - Learned dependency
  - Is there quality discussion about the problem?

Remove barriers to learning

- Remove Fear
  - Do people run for cover?
  - Do people play safe?
  - Scared to take risks
- Don’t Jump to solutions (Single loop)
  - Discuss the problem in detail
  - Generate several solutions

Remove barriers to learning

- Individuals
  - Only so much an individual can do
  - Need to work as a team
  - Raise awareness in other

Share the learning

- Not enough for you to learn
  - Your development group needs to learn
  - Your customers needs to learn
- Limits to how much you can change if others don’t
- Others can help you learn
  - You can help others learn

Increase the learning: Feedback

- Need feedback to learn
- Increase the feedback
  - How good is this?
  - What could be better?
  - Feedback from others

Unlearning

- Sometimes we have to stop doing something
- Recognise an assumption we're making
  - Relax the assumption
- Forget a lesson we learnt
- Break out of single-loop behaviour

Dialogue

- Talking is a forum for learning
- Listen to others
  - Understand their point
  - Understand their feedback
  - Understand their suggested action
- Ask questions
  - Help them understand, help you understand

Dialogue

- Find a shared understanding
- Find an agreed action
- Can you measure the solution?

Learn with other people

- Team is the building block
- Modern software is too big for one person
  - Learn & work as a team
- Groups of like minded people
  - e.g. Form a book group; lunch party

Individual Reflection

- Take time to reflect yourself
  - Start a diary, keep a blog, tape recorder
  - Long baths
  - Write for CVu/Overload
- Think about what you are doing
  - How you are doing it
  - Why you are doing it
  - Your feelings

Team Reflection

- Teams can reflect too
  - Project retrospectives
  - In project reviews
- Give the team time to talk about *how* the are doing not just *what*

Not just the software team

- Your software will bring change to others
- Your customers will change
- Engage them in dialogue
- Understand they will change
  - Good reason for incremental delivery
- Include them in discussions
  - Helps everyone learn & is fair

The End

- Thank you for listening

Resources

- Argyris, *Organisational Learning II*, 1996
- Nonaka, *The Knowledge Creating Company,* 1995
- Pfeffer, *The Knowing Doing Gap*, 2000
- Seely-Brown, *Social Life of Information*, 2000
- Senge, *The Fifth Discipline,* 1990
- Senge, *The Dance of Change, 1999*
- Whitmore, *Coaching for Performance*, 2002