Req Gathering, Analysis & Specification

Dr Samit Bhattacharya

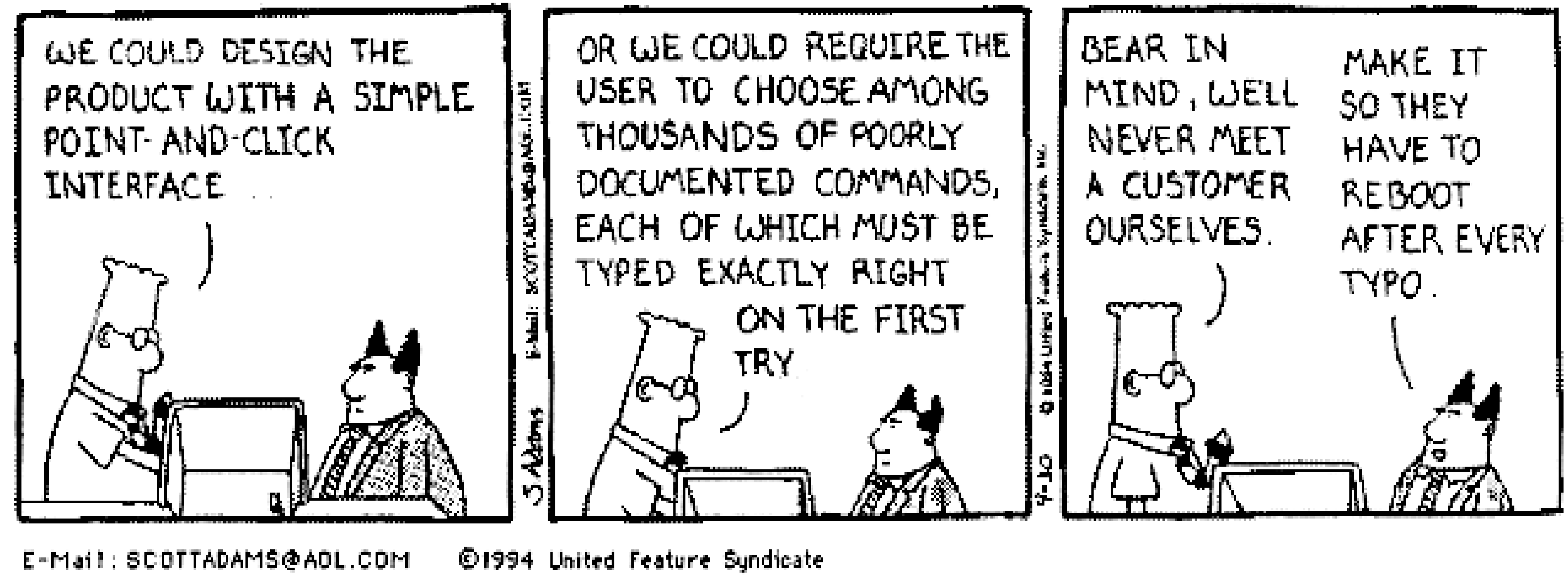
Computer Science and Engineering

IIT Guwahati

Recap

- SCD vs UCD

System Centered Design

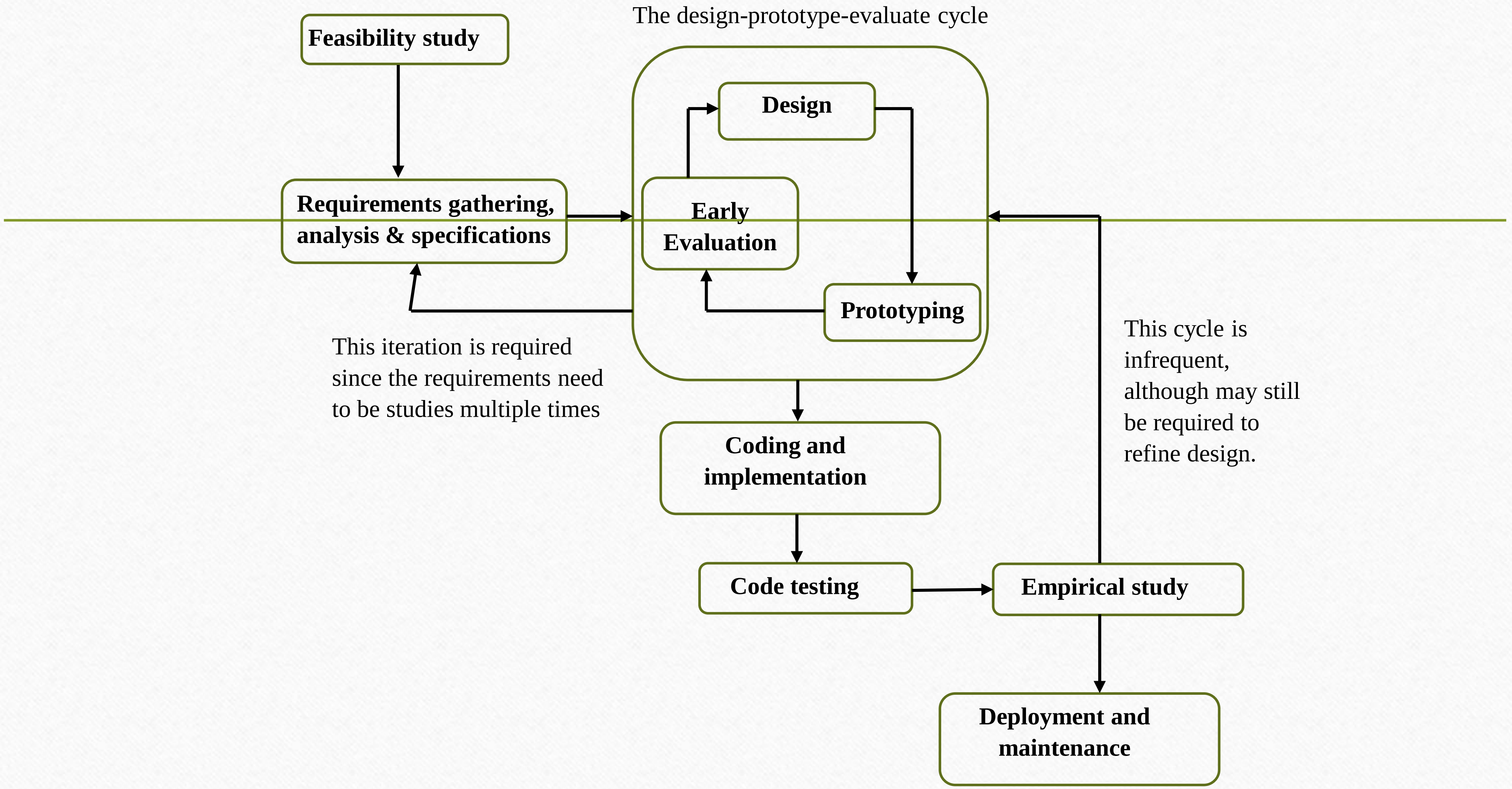


System Centered Design

- Main concerns
  - What can I easily build on this platform?
  - What can I create from the available tools?
  - What do I as a programmer find interesting?

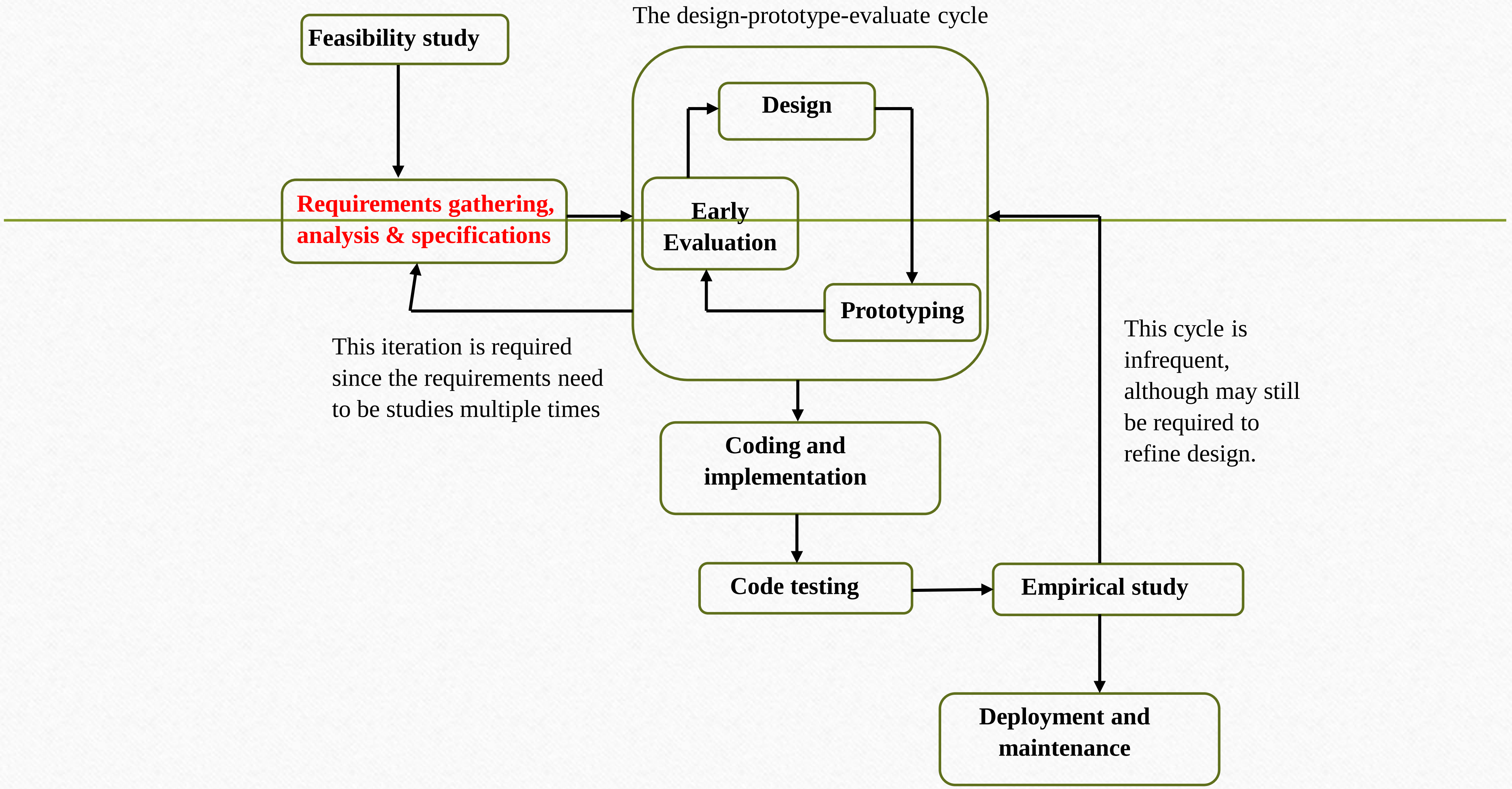
User Centered Design

- Design is based upon a user’s
  - Abilities
  - Context
  - Goals/needs
- Not only programmer’s convenience


The design-prototype-evaluate cycle
Feasibility study
Requirements gathering, analysis & specifications
Design
Early Evaluation
Prototyping
This iteration is required since the requirements need to be studies multiple times
This cycle is infrequent, although may still be required to refine design.
Coding and implementation
Code testing
Empirical study
Deployment and maintenance

Note

- First stage: feasibility study (we'll not spend much time on it)
- Instead, we'll discuss the second stage – requirement analysis and specifications


The design-prototype-evaluate cycle
Feasibility study
Requirements gathering, analysis & specifications
Design
Early Evaluation
Prototyping
This iteration is required since the requirements need to be studies multiple times
Coding and implementation
Code testing
Empirical study
This cycle is infrequent, although may still be required to refine design.
Deployment and maintenance

Example Scenario

- Recall the calendar app (mobile app)
- What we mean by requirement gathering, analysis & specification for this system
  - What are the requirements?

Example Scenario

- Many responses
  - Some related to feasibility study
  - Some related to design phase

- Conclusion: requirement analysis is different than what you think!

Basic Idea

- Two broad types of requirements
  - Functional
  - Non-functional

Functional Requirements

- Main idea
  - Abstraction (identify black box functions to be supported by the system, with clearly defined input and output)
  - Decomposition & hierarchy (to better manage the functional description)

Functional Requirements

- Example again!
  - What are the functions for the calendar app?
  - How to formulate in terms of functional specifications?
  - Formation of function hierarchy!

Functional Requirements

- Key points to remember while identifying functions
  - Avoid exploratory style – from implementation for function (it should be other way round)
  - Abstraction and hierarchy

SRS Format

- When you put the requirements in textual form, you get the SRS (software requirement specification) document
  - Hierarchical structure (R1, R1, R1.1 etc)
  - Each R (e.g., R1, R1.1 etc) represents a function with clearly defined input and output and some description (optional)

SRS Format

- Example - Withdraw Cash from ATM
  - R1: withdraw cash
  - Description: The withdraw cash function first determines the type of account that the user has and the account number from which the user wishes to withdraw cash. It checks the balance to determine whether the requested amount is available in the account. If enough balance is available, it outputs the required cash, otherwise it generates an error message.
    - R1.1 select withdraw amount option
      - Input: “withdraw amount” option
      - Output: user prompted to enter the account type

SRS Format

- Example - Withdraw Cash from ATM
  - R1: withdraw cash
  - Description: ….
    - R1.1 select withdraw amount option
    - R1.2: select account type
      - Input: user option
      - Output: prompt to enter amount

SRS Format

- Example - Withdraw Cash from ATM
  - R1: withdraw cash
  - Description: ….
    - R1.1 select withdraw amount option
    - R1.2: select account type
    - R1.3: get required amount
      - Input: amount to be withdrawn in integer values greater than 100 and less than 10,000 in multiples of 100.
      - Output: The requested cash and printed transaction statement.
      - Processing: the amount is debited from the user's account if sufficient balance is available, otherwise an error message displayed.

Non-Functional Requirements

- Usability
- Reliability
- Some others
- We’ll focus on usability

Motto

- Know thy user

Recap - Usability Definition

- ISO definition (ISO 9241-210:2009) - "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a **specified context of use**."

Usability Req Gathering - Contextual Inquiry

Contextual Inquiry

- A method to gather end-user requirement (**one of the many such methods**)
- Required for UCD

Contextual Inquiry

- Its importance in UCD highlighted as early as 1988, although the term used was "phenomenological research method"
- By the end of the 1990s, it became well developed and popular

Contextual Inquiry

- Often referred to as a “semi-structured” interview process
  - Consists of both **interview** and **observations**

Contextual Inquiry

- Primary objective - watch and observe users perform tasks in their “natural” work setting (the work context)
- During observations, users may be asked questions to clarify on certain behavior or activities

Contextual Inquiry

- Usually the process continues for **one** to **two hours**

Contextual Inquiry

- Observations (including interaction with users) enable designers learn about task domain, work culture and physical and social constraints at the work place
  - **The knowledge is important since the "specific work context" is an essential part of the definition of usability**

Contextual Inquiry

- **Ex - recollect the calendar app example from the last lecture**

Contextual Inquiry

- Suppose the target group of users are the managerial staff of any organization

Contextual Inquiry

- Without any knowledge about intended user behavior, we can come up with all sorts of possible designs

Contextual Inquiry

- However, suppose we now performed a CI and **observed two important activities performed by the users: set meeting dates and reminders**

Contextual Inquiry

- Clearly, a calendar app intended to be used by this group of users should have support for these activities

Contextual Inquiry – Two Ways

- The inquiry can be performed in *active* or *passive* mode

Contextual Inquiry – Two Ways

- Active mode - observer is physically present
- Passive mode - user activities are videotaped and reviewed later

Contextual Inquiry

- **During the inquiry, observer should act as "apprentice" and treat the users as "masters"**
- Purpose is to learn user behavior, **not to teach** them how to behave

Contextual Inquiry – Five Stages

- Plan
- Initiate
- Execute
- Close
- Reflect

Plan (Things to do)

- Identification of the goals of observation
- Knowledge gathering about task domain
- Arrangements for recording observations
- Identification of users and the date, time and place of observation
- Scripting of the procedure and rehearsals

Initiate Stage

- Contact the authorities for permission and open communication with the end users to remove their anxieties

Execute Stage

- Perform observations and record data

Close Stage

- Send "thank you" notes to the users (you may require them later)

Reflect Stage

- Analyze the data collected to identify design goals
- Not Easy since you may end up with lots of data
- Tools and techniques help - e.g, affinity diagram method

Creating an Affinity Diagram

Step 1 - Generate ideas

Step 2 - Display ideas

Step 3 - Sort ideas into groups

Step 4 - Create header cards

Step 5 - Draw finished diagram

Step 2 - Display the Ideas

Issues in Implementing Continuous Process Improvement

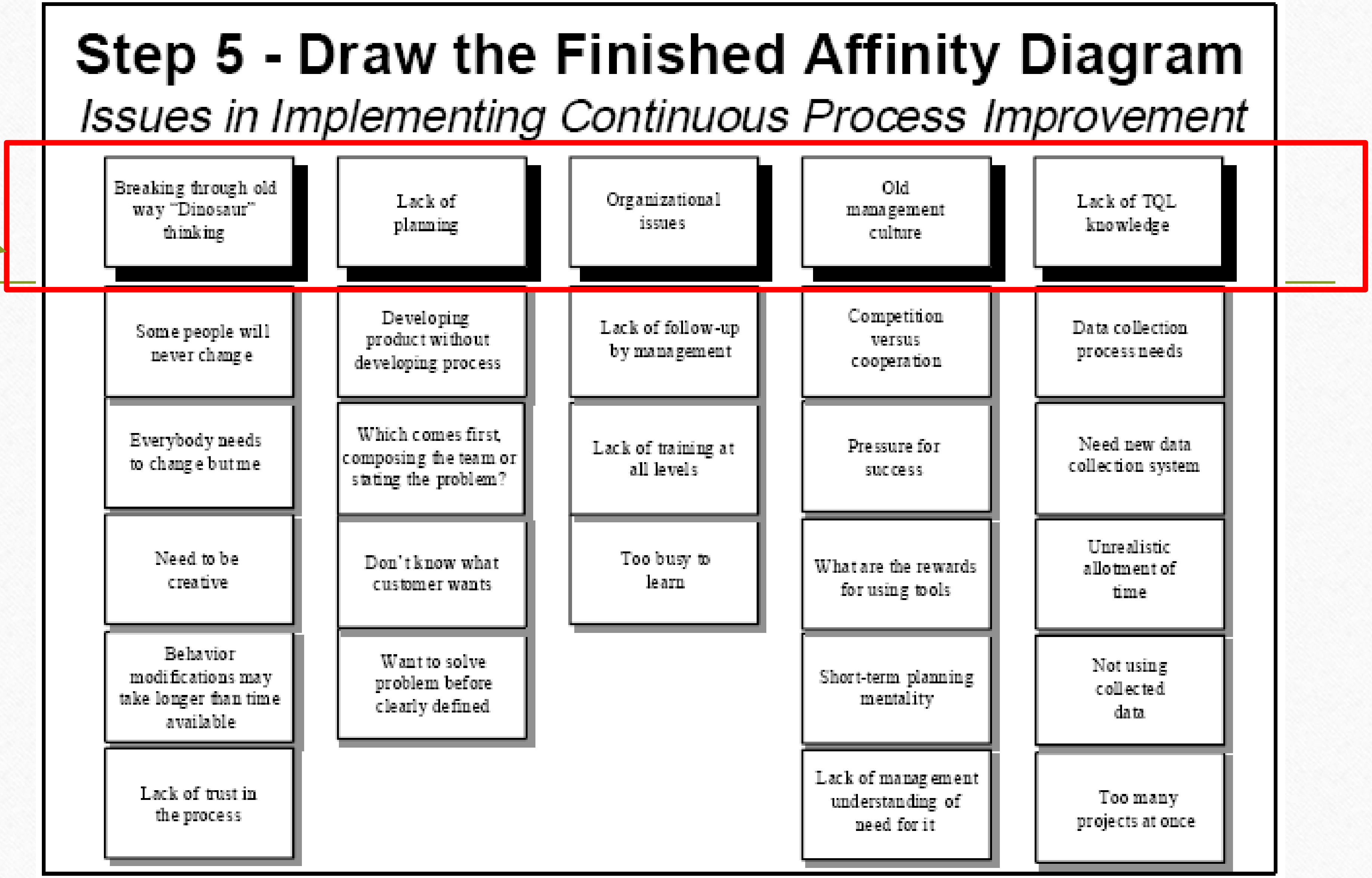
- Data collection process needs
- Pressure for success
- Lack of follow-up by management
- Which comes first, composing the team or stating the problem?
- Want to solve problem before clearly defined
- Lack of training at all levels
- What are the rewards for using tools
- Unrealistic allotment of time
- Everybody needs to change but me
- Need new data collection system
- Lack of management understanding of need for it
- Developing product without developing process
- Not using collected data
- Too busy to learn
- Don't know what customer wants
- Some people will never change
- Lack of trust in the process
- Behavior modifications may take longer than time available
- Short-term planning mentality
- Competition versus cooperation
- Need to be creative
- Too many projects at once

Step 3 - Sort Ideas into Related Groups

Issues in Implementing Continuous Process Improvement

Some people will never change	Developing product without developing process	Lack of follow-up by management	Competition versus cooperation	Data collection process needs
Everybody needs to change but me	Which comes first, composing the team or stating the problem?	Lack of training at all levels	Pressure for success	Need new data collection system
Need to be creative	Don't know what customer wants	Too busy to learn	What are the rewards for using tools	Unrealistic allotment of time
Behavior modifications may take longer than time available	Want to solve problem before clearly defined		Short-term planning mentality	Not using collected data
Lack of trust in the process			Lack of management understanding of need for it	Too many projects at once

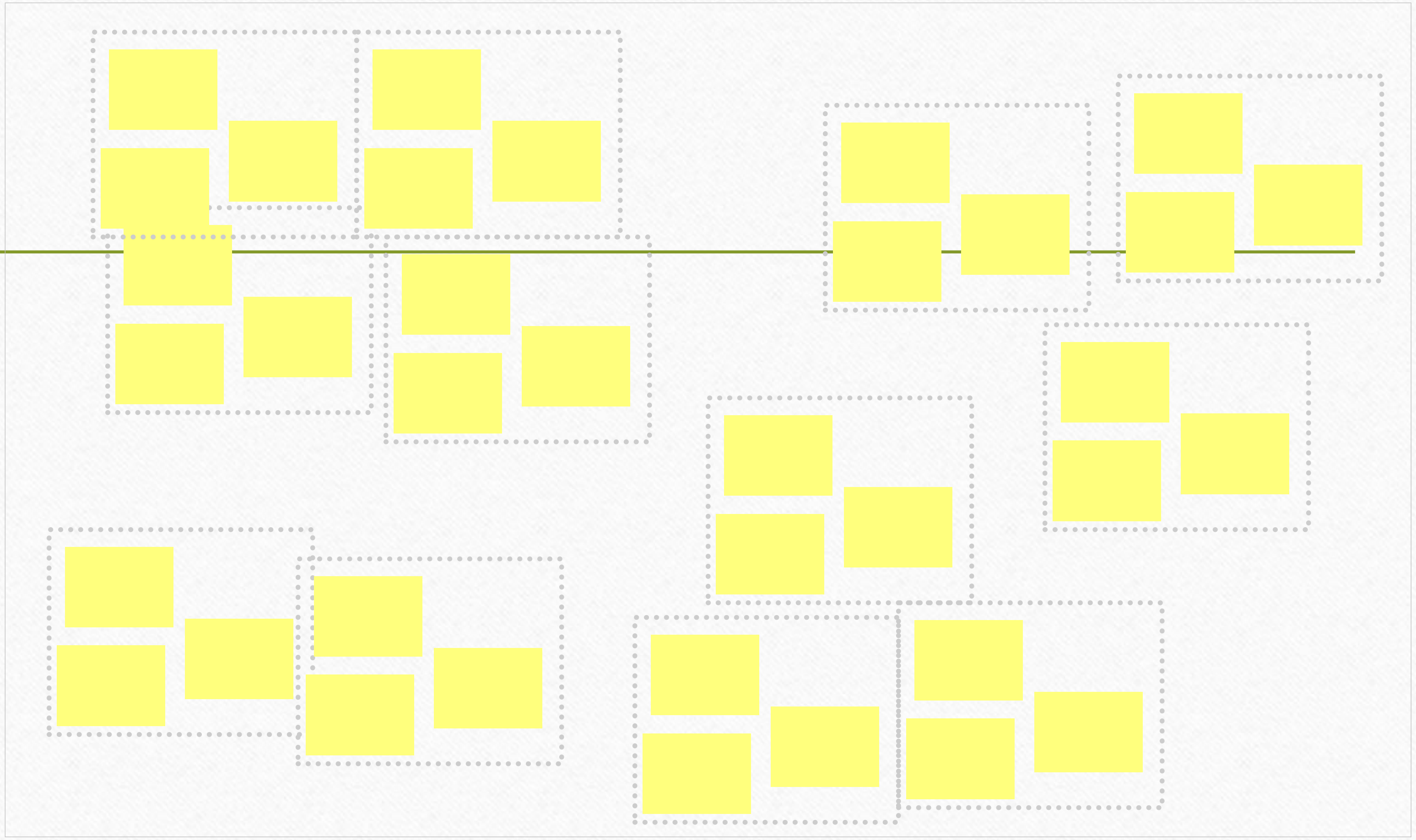
Header cards (step 4)



Affinity Diagram

- Represents your collective understanding of how the user works
- Paste large sheet of paper on the wall
- Write each task or subtask on a *post-it* note
- Organize and group post-its on the large sheet of paper until the changes stabilize
- Clarify questions and missing details with users

Affinity Diagram



Book

- Rajib Mall – Fundamentals of S/W Engineering
- Roger Pressman –S/W Engineering: A Practitioner's Approach
- NPTEL MOOCS course on user-centric computing for HCI, L5
- Samit Bhattacharya - Human-Computer Interaction: User-Centric Computing for Design, McGraw-Hill India (Chapter 2, Sec 2.4.3)