SOEN – 6841
Software Project Management

Amin Ranj Bar

Winter, 2021

Agenda

- Productivity rate
- Functional size measurement
- COSMIC method
- Retrospective sessions
  in Agile Projects.

Background:
Analysis of the Productivity Rate

- Productivity rate is expressed in terms of output (functional size of the developed features) / input (team's hours worked in staff-hours)
- Why Functional Size and not User Story Points (USP)?

  When the size of the user stories is measured in USP during a software process assessment across several Agile teams, inconsistencies can be identified due to the subjective nature of the user story points.

MEASURES FOR THE REQUIREMENTS PHASE: Functional Size Measurement (FSM)

- What is functional size?

Functional size is defined in ISO 14143-1:2007(E) as:
"a size of the software derived by quantifying the Functional User Requirements (FUR)," where FUR is as "a sub-set of the User Requirements describing what the software shall do, in terms of tasks and services"

- Examples of FUR are:
  - input of students data in a registration system;
  - calculate the average mark of students in a course;
  - list the students that are above the average.

Why to measure Functional Size?

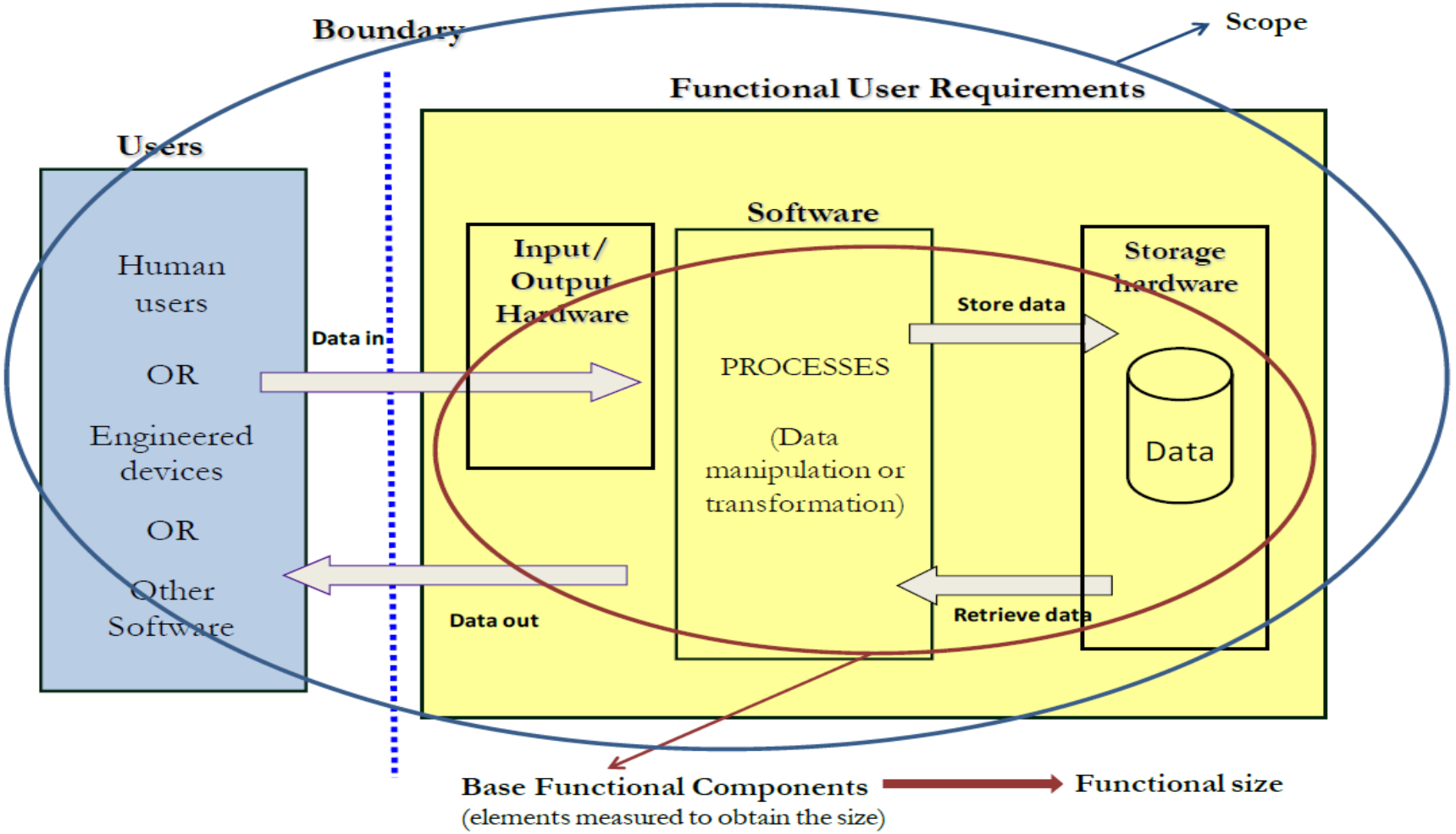
- FSM enables comparison of applications and projects based on their size.
- Productivity rates for applications of a similar attribute profile can be compared for benchmarking support ratios and improvement purposes.
- Productivity rates from past projects can also be used to predict effort, once a project's Functional Size has been determined.

MEASURES FOR THE REQUIREMENTS PHASE: Functional Size Measurement (FSM)

The current functional size measurement (FSM) methods:

- **ISO 19761:2011 COSMIC functional size measurement method**
- ISO 20926:2009 IFPUG 4.1 functional size measurement method
- ISO 24570:2005 NESMA functional size measurement method
- ISO 20968:2002 MKII function point analysis
- ISO 29881:2010 FiSMA 1.1 functional size measurement method

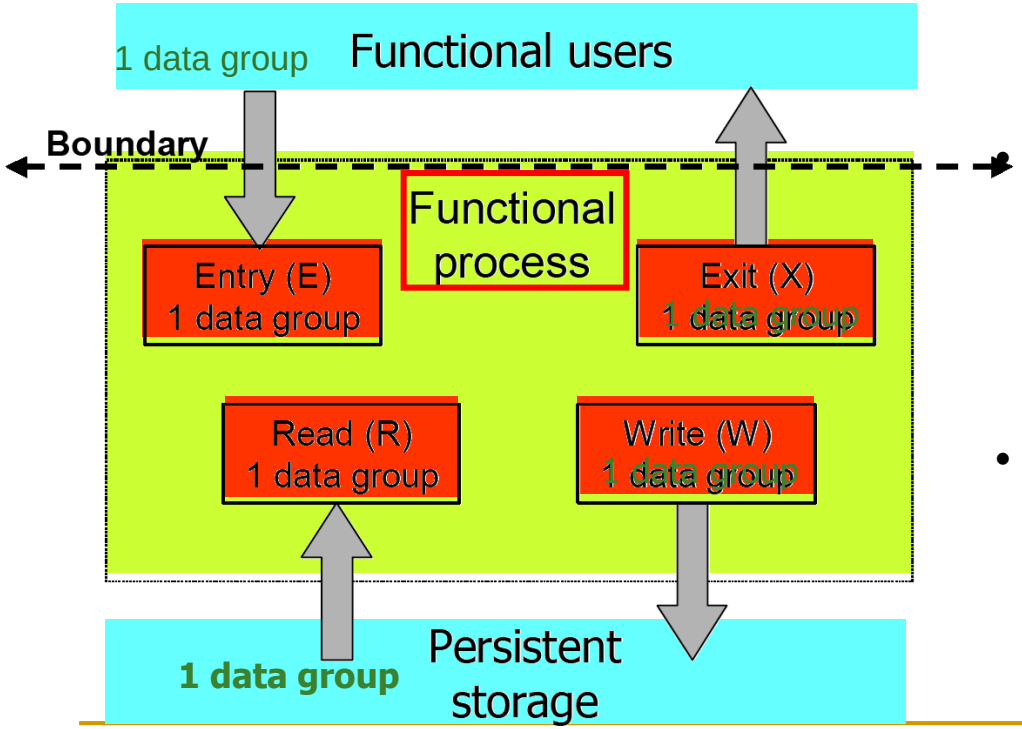
Functional size measurement methods: general representation [ISO 14143-1:2007]



The COSMIC method

1. The *Basic Functional Component (BFC) is the* ***data movement****.*
2. The measurement *unit is a COSMIC Function Point (CFP), which represents* ***one data movement****.*
3. A *functional user of a software application can be: a human, another software application, or a hardware device.*
4. A *boundary is a conceptual interface between the functional users and the software application.*
5. The functional users interact with the software application **via *data movements.***
6. There are *four types of data movements: Entry (**E**), eXit (**X**), Read (**R**), and Write (**W**).*

COSMIC FSM Standard (ISO/IEC 19761)



- **System is viewed as a black-box**
- ***Size*** =

  The number of movements of "***data-groups***" between the System and the Users / the Persistent Storage
- **Data-movements of 4 types:**
  1. ***Entry***
  2. ***Exit***
  3. ***Read***
  4. ***Write***
- **Unit of Size is COSMIC Function Points (CFP)**

Measurement Units

DEFINITION – COSMIC unit of measurement
1 CFP (Cosmic Function Point), which is the size of one data movement.

PRINCIPLE – The COSMIC measurement principle
a) The size of a functional process is equal to the number of its data movements.
b) The functional size of a piece of software of defined scope is equal to the sum of the sizes of its functional processes.

- **Each data movement**
  - Is Counted only once within a functional process
  - Does not involve another data movement type

Functional process

A functional process is an elementary component of a set of Functional User Requirements, comprising a unique, cohesive and independently executable set of data movements.

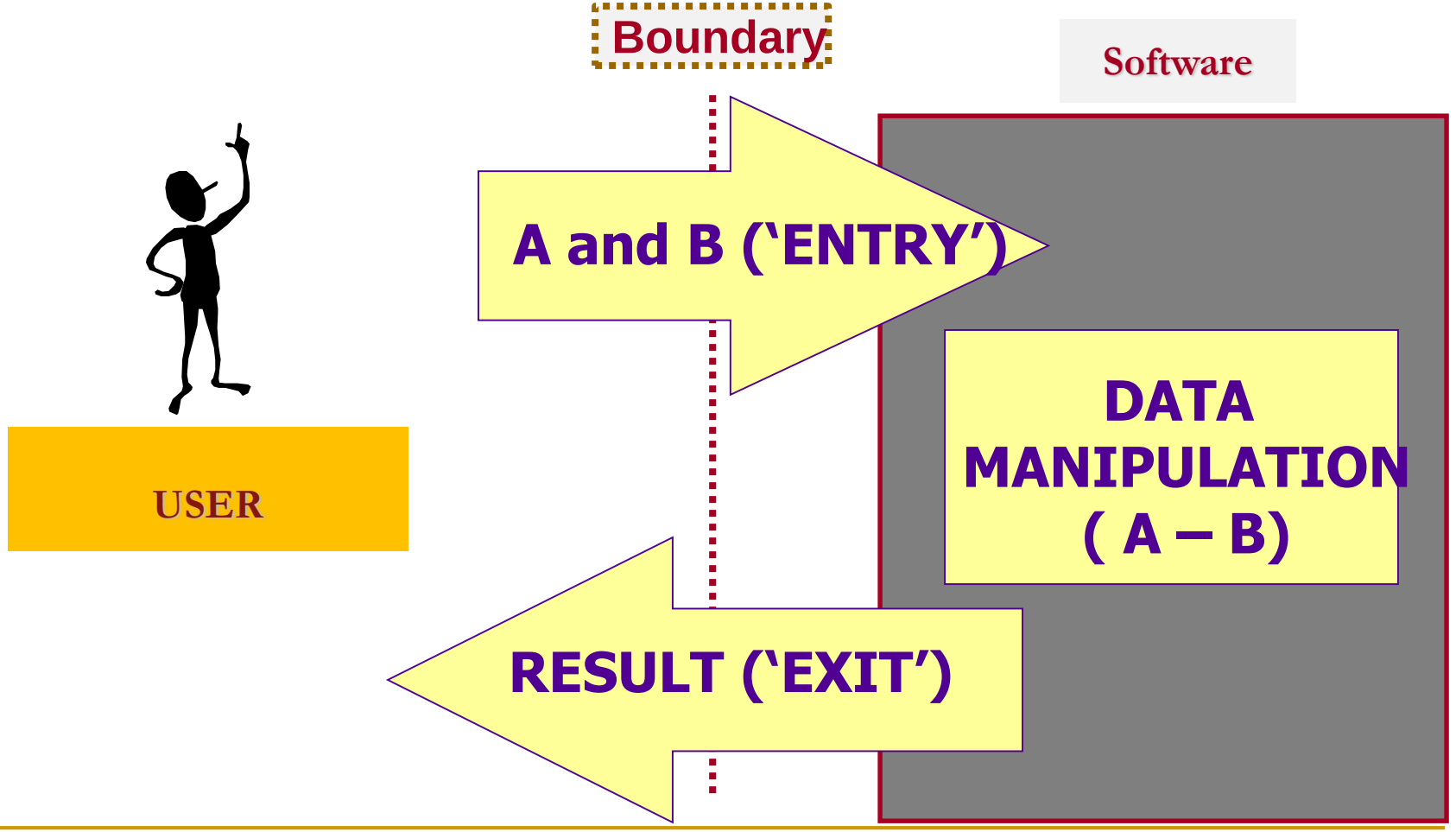
In soen6841 project: consider a user story

Example

Apply COSMIC measurement procedure to measure "**A – B**":

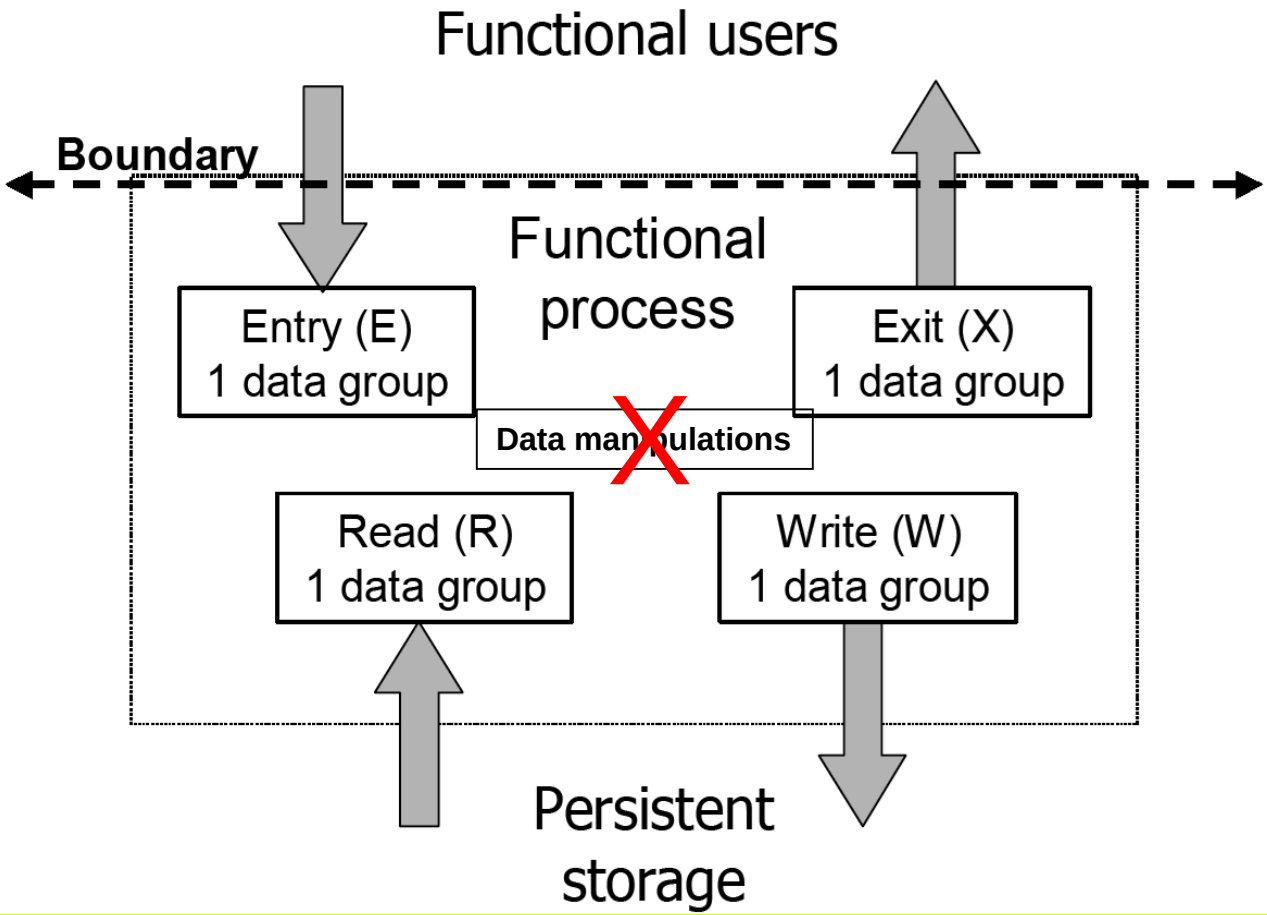
- The user enters the values A and B of the operands
- The system in return displays the result

COSMIC model for A-B:
1 Entry, 1 Exit. Total: 2CFP
(Data manipulation is excluded from the counting)



ISO/IEC 19761

COSMIC counting rules



*Each data-movement of one data-group accounts for **one CFP** !*

COSMIC counting rules

- Each functional process:
  - is independently executable
  - triggered by an event
  - Contains all that is required to be done in response to the triggering event
- Data movement types: E, X, R, W
- Entry type : considered to include certain associated data manipulations such as validation of the entered data
  - Triggering events are Entry type
  - Clock and timing events are Triggering events
- Each data movement moves data belonging to a single data group
  - data movement “de-duplication”

Aggregate the Measurement Results

FP Size = Sum(Ne) + Sum(Nx) +Sum(Nr) + Sum(Mw)

Where:

- Ne = number of Entries
- Nx = number of Exits
- Nr = number of Reads
- Mw = number of Writes

- there is no upper limit to the functional size of the functional requirements
- Min size: 2 CFP

Example of one Functional Process

“Create a list of sales territories"

1 -This use case starts when the user enters the sales territories’ codes and names.

2 - The system saves the sales territories.

3 - The system displays a confirmation message or, if it fails to save the territories, it issues an error message.

ID Process	Process description	Subprocess	Data Group	Data movement Type	CFP	Σ CFP
1.1	Create Sales Territories	User enters the sales territories’ codes and names	SalesTerritories	E	1	
		Save the territoires	SalesTerritories	W	1	
		displays confirmation or error message	Error message	X	1	
	Total functional size in CFP =					**3**

COSMIC measurement procedure

1. Identify the functional users
2. Identify the triggering events
3. Identify the functional processes enabled by those events
4. Identify the data groups
5. Identify the data movements from the interface (Entry, eXit, Read and Write)
6. Obtain the total number of Cosmic Function Points (CFP)

Example of COSMIC in Practice

Extract from a requirements document
...The following use case describes creating a new budget. First, the user navigates to the budget creation page. He then enters the budget attributes. All the mandatory attributes cannot be empty and the budget amounts cannot be negative. User saves the budget. ...

Requirement Sentence	Sentence Type	Data-group	Data-movement	CFP
(1) The following use case describes creating a new budget.	N/A			
(2) First, the user navigates to the budget creation page.	Functional	User Request	Entry	1
(3) He then enters the budget attributes.	Functional	Budget	Entry	1
(4) All the mandatory attributes cannot be empty and the budget amounts cannot be negative.	Non-Functional			
(5) User saves the budget.	Functional	Budget	Write	1

More Complex Exercise

Scope of measurement: Measure the size of the functionality "Create a new customer" as it is described in the flow of events.

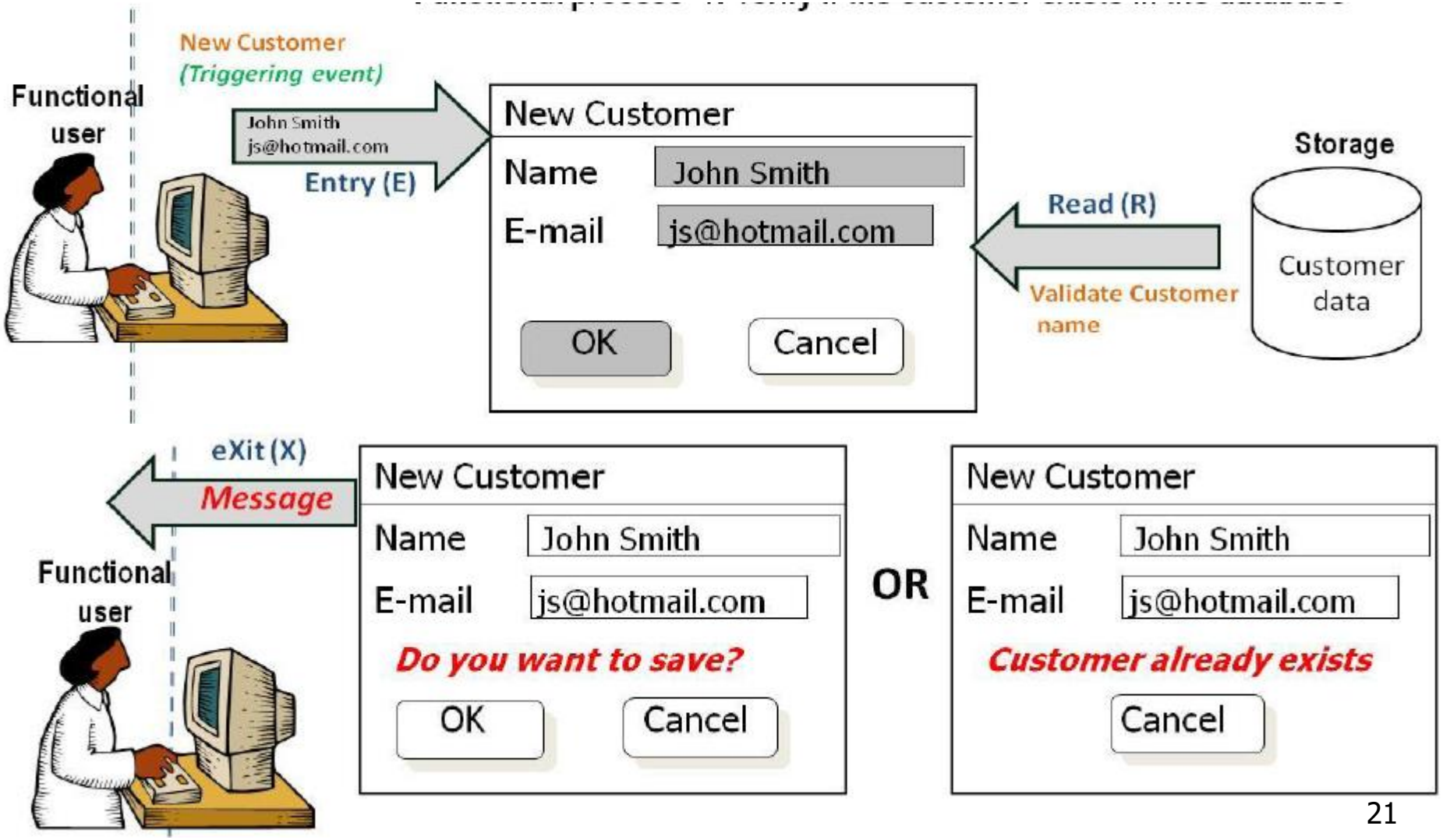
Functional user: Salesman

Pre-conditions: The salesman is already logged in the system and has selected the option "Create Customer"

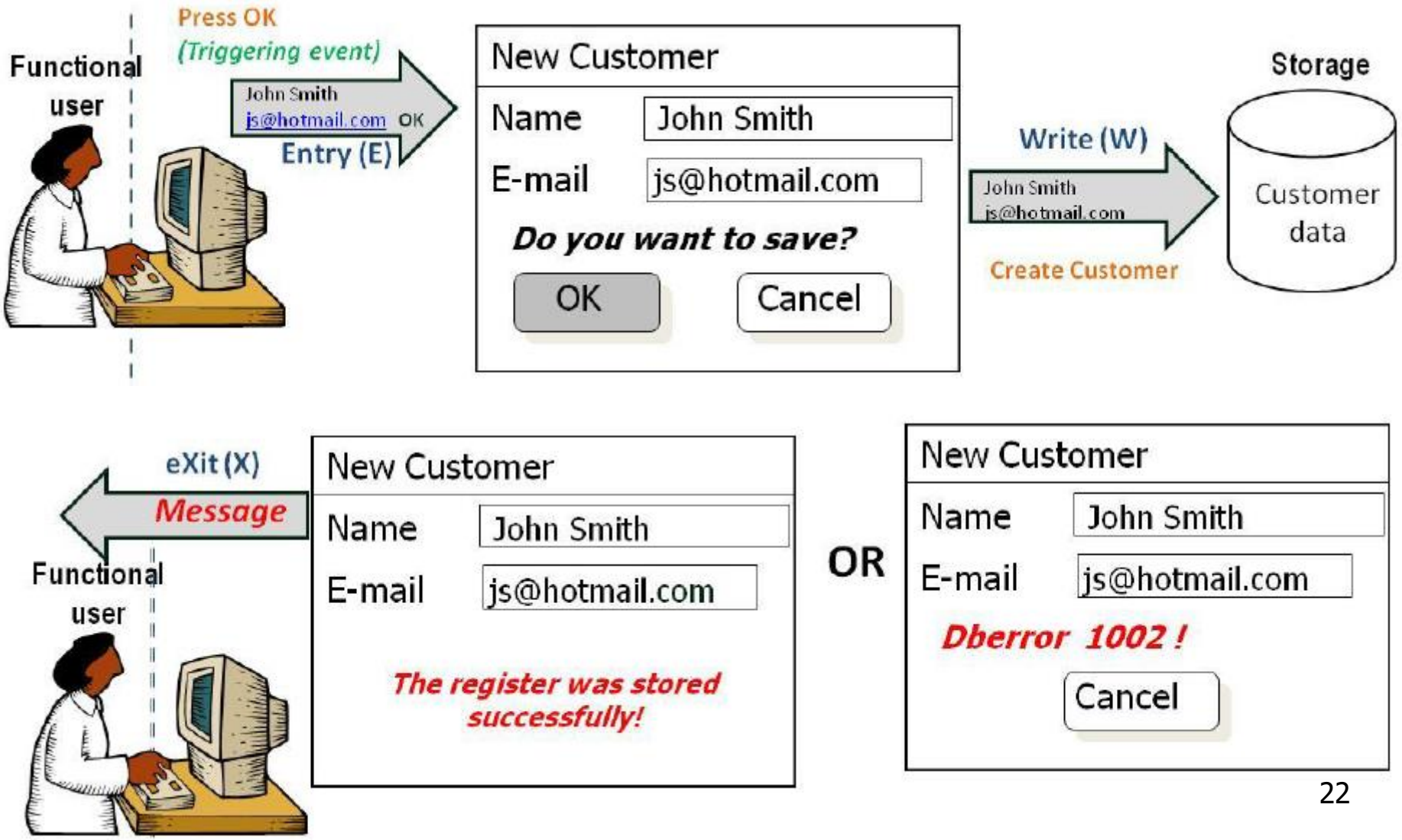
Flow of events:

1. The salesman enters the name and email of the new customer (John Smith, js@hotmail.com) and press OK.
2. The system verifies if the customer already exists in the database.
3. If the customer exists, an error message is displayed.
4. If the customer is new, the system asks the salesman to confirm the data that is going to be saved by pressing OK. If needed, the salesman can correct the customer data.
5. A new customer is created into the database.
6. A confirmation or error message is displayed.

Exercise: Verify if the customer exit in the database



Exercise: confirm customer data and save



Exercise: Hints

- **Functional processes:**
  1) Verify if the customer exists in the database,
  2) Create a new customer in the database**.**
- **Data group (DG):**
- Customer
  - **Data attributes:**
    - Name, email
- System messages

Exercise: Solution

Total functional size= 6 CFP

Process	Subprocess	Data Group (DG)	DM	CFP	Comment
Verify if the customer exists in the database	The salesman enters the Customer data and presses OK	Customer	E	1	
	The system retrieves the existing customers from the database to verify if John Smith already exists as customer	Customer	R	1	
	A confirmation message OR an error message (if the customer already exists)	Software Messages	X	1	
Create a new customer in the database	The salesman verifies/corrects the customer data and presses OK	Customer	E	1	This movement is considered as an Entry because the salesman can retype the customer data
	The customer data is stored in the database	Customer	W	1	
	Confirmation or error message	Customer	X	1	

More Principles of COSMIC FSM

Each functional process consists of sub-processes.

A sub-process may be either a **data movement** or a **data manipulation.**

A data movement moves a single **data group** .

There are four data movement types, **Entry**, **Exit**, **Write** and **Read**. An Entry moves a data group into a functional process from a functional user. An Exit moves a data group out of a functional process to a functional user. A Write moves a data group from a functional process to persistent storage. A Read moves a data group from persistent storage to a functional process.

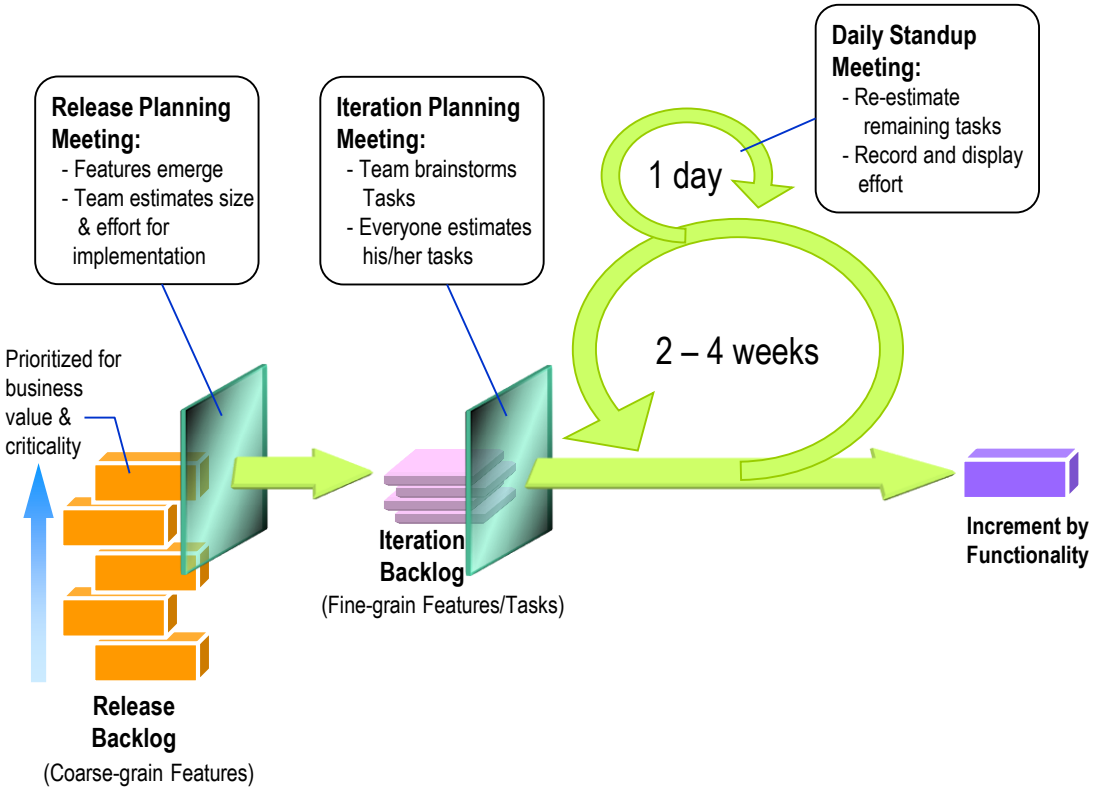
A data group consists of a unique set of **data attributes** that describe a single **object of interest.**

Each functional process is started by its **triggering Entry** data movement. The data group moved by the triggering Entry is generated by a functional user in response to a **triggering event.**

Planning & Estimation in Software Projects

"Estimates will never be anything other than approximate, however hard you try."
- Beck & Fowler (2000)

- Software Size Estimation
  - Effort is a function of size
  - Textual Requirements are the primary source of information
    - Problem Description
    - User Stories
    - Smart Use Cases
    - **All in plain natural language!**
- Current Practice of User Stories Size Estimation
  - Planning poker technique
  - Subjective evaluations
  - Inconsistent results across teams



COSMIC & QA

- The process of measuring a functional size of a statement of requirements provides a very good check on the quality (clarity, completeness and consistency) of the requirements.
- If the requirements are unclear such that a functional size cannot be measured properly, then the developers will also face difficulties; they will have to make assumptions, which may or not be correct.

RETROSPECTIVE

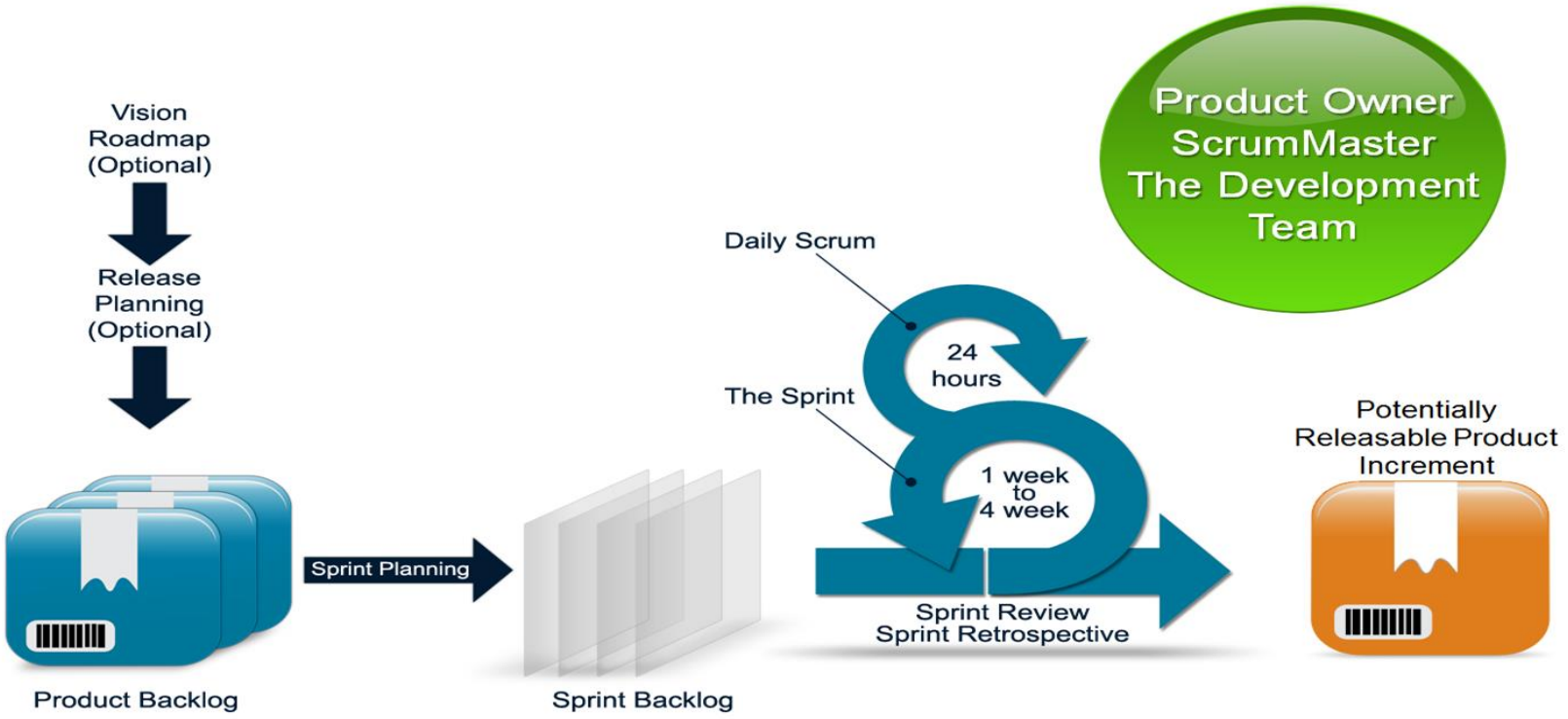
Objectives

Define what a retrospective ?

Understand why we need retrospective? And when?

Understand how to perform a Retrospective?

Scrum Process Review



What?: Definition of a Retrospective

Agile principle: At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

It’s a fundamental vehicle to discover, share, and pass along the learning from experience

A lesson isn’t “learned” until you’ve actually acted on it correctly. Until that point, it’s only a lesson identified.

It can be useful for a project management to communicate positive and negative practices.

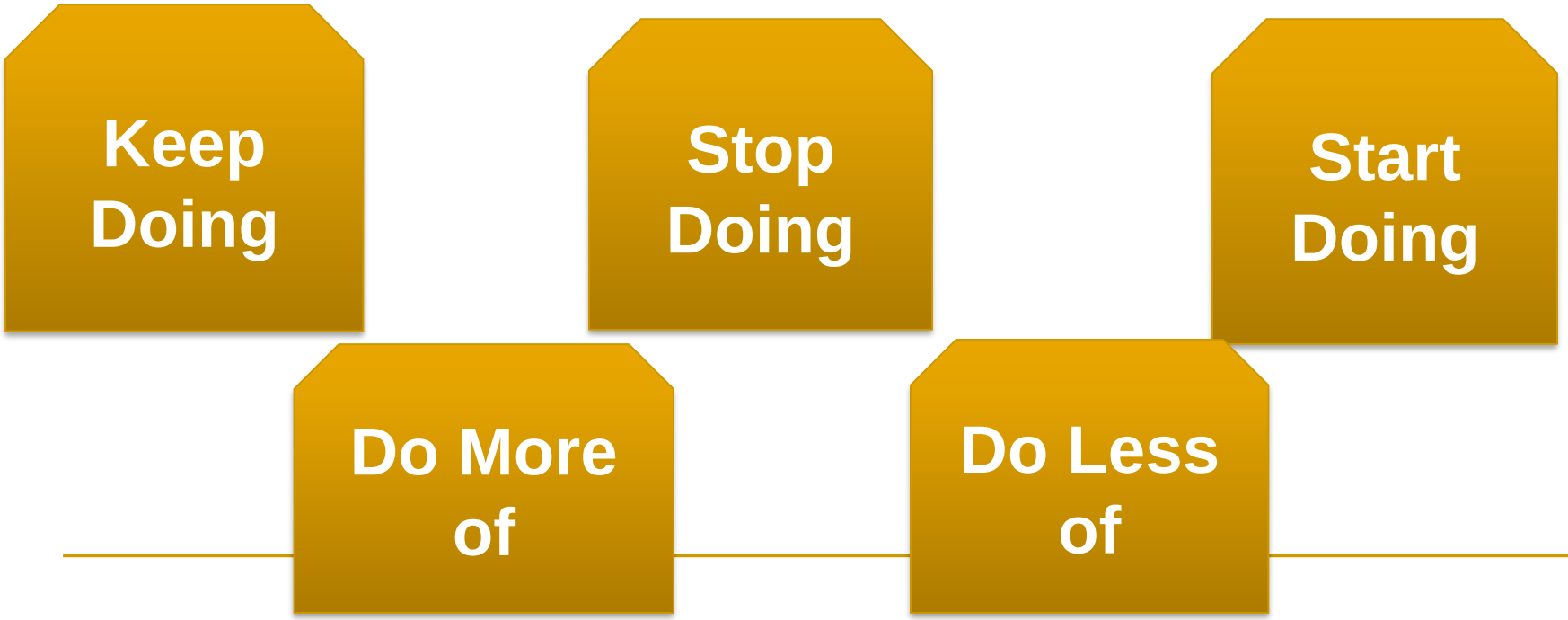
Retrospective

Retrospective is about looking “back” to move forward.

- Ask:
  - **What went well?**
  - **What could be changed to deliver better results?**
  - **What should be stop doing?**
  - What have we learned?
  - What still puzzles us?
- **Capture lessons learned:**
  - Target the top **three** issues.
  - Do not take on too much!
- **Next Sprint:**
  - Take 1 or 2 concrete actions:
    - For example, “We need to stop doing X.” “We need to get better at Y.”

How? Retrospective Model

There are many possible questions to address during the retrospectives:



To summarize: Sprint Retrospective

The Sprint Retrospective should focus on three questions.

What went well? (Continue)

- Each member can offer up a brief bullet point.
  - Example: I like the inner circle approach to the Daily Scrum which keeps non-team members from interrupting our meeting.

What do we want to change? (Start)

- **The team should try to agree on one or two things to try in the next Sprint to help resolve one problem**

What went poorly? (Stop)

- **Each member can offer up a brief bullet point: EX: The daily meeting at 7:30 A.M. is too hard for team member A to hit and we had to wait multiple times during the Sprint.**

How to run a Retrospective: Process

DOCUMENT the postmortem REPORT IN Sprint presentation

Step one:

Start with the “What went well” section, give everyone (in the team) about 5 minutes to come up with as many ideas as they could.

Step two:

Review each of the parts

Each team will go through each idea, discuss a different point, agree/disagree (in a healthy way). Most importantly, make sure everyone **on the team has a voice and is able to express their opinion**

Select **3** ideas from **each** part to put in the postmortem report.

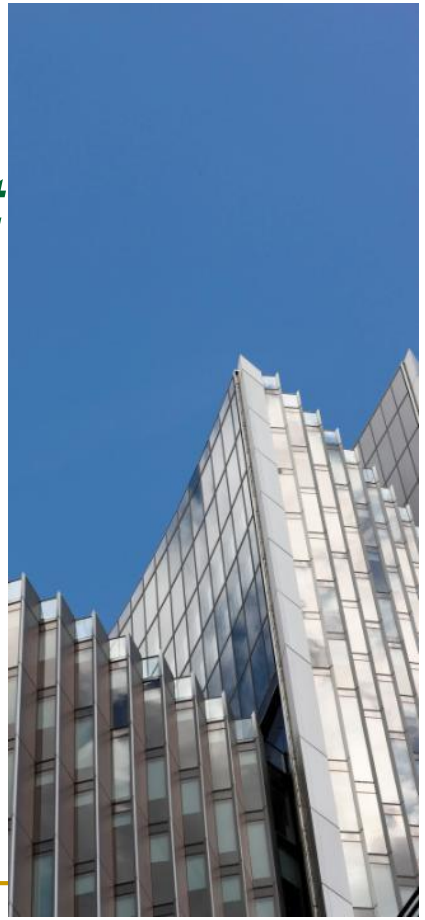
Retrospective

Take retrospectives to the next level:

- Consider tracking your results.
- Consider using a self-assessment tool (assess team's agile skills at end of each Sprint)

Why it is important?

The prime directive says: "Regardless of what we discover, we understand and truly believe that everyone did the best job they could, given what they knew at the time, their skills and abilities, the resources available, and the situation at hand"



Sprint 1 retrospective

- What you did, what you would do the same, and what you would do differently next time?
  - The focus is not on blame, but on improvement.
- Satisfied Customer
  - comment on how clients' requests were managed

Summary

- Productivity rate
- Functional size measurement
- COSMIC method
- Retrospective sessions
  in Agile Projects.