

# Measuring the functional size of real-time software

**Co-authored by:**

**A. Abran, J.-M. Desharnais, S. Oigny**

**Université du Québec à Montréal -**

**Software Engineering Management Research Laboratory,**

**Centre d'Intérêt sur les Métriques (C.I.M.), CANADA**

***April 1999***

# *Presenter profile*

---

## ⊙ **Serge Oigny, M.Sc.**

- ✓ **Director - Technological innovations, UQAM-Software Engineering Management Research Laboratory**
- ✓ **Member of CIM Executive Committee**
- ✓ **Editor of FFP measurement manual v. 2.0**
- ✓ **Formerly Corporate Manager of software development in the pulp & paper industry**
- ✓ **13 years experience in the IT consulting market**

# Agenda...

---

- ⊙ Introduction
- ⊙ Characteristics of real-time software
- ⊙ The measurement process model
- ⊙ Measurement Procedures
- ⊙ Overview of field tests results
- ⊙ Conclusion

# *Introduction...*

---

- ⊙ **Functional size measurement**
- ⊙ **Characteristics of Full Function Points**
- ⊙ **An analogy**

# ***Functional Size Measurement***

---

⊙ ISO/IEC/JTC1/SC7 Standard #14143 definition:

“ Functional Size : A size of software derived by quantifying the **functional user requirements**”

# *Characteristics of FFP...*

---

- FFP is a Functional Size Measure
- Focused on the '**User functional view**'
- Applied at **any time** during the software development life cycle
- Derived in terms **understood by users**
- Derived without reference to:
  - **effort**
  - **methods** used
  - **physical or technical** components .

# *Characteristics of FFP...*

---

- Version 1.0 of FFP released in 1997
- Version 2.0 currently under final review
- Major improvements will be outlined  
using this mark: **V 2.0**

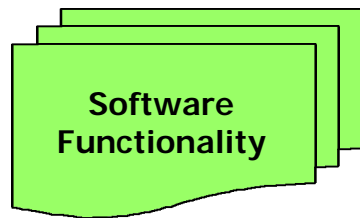
# *An analogy...*



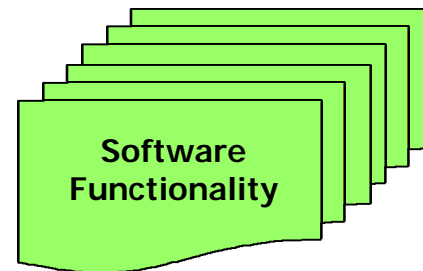
**2000 sq. ft.**



**4000 sq. ft.**



**500 FFP**



**1000 FFP**

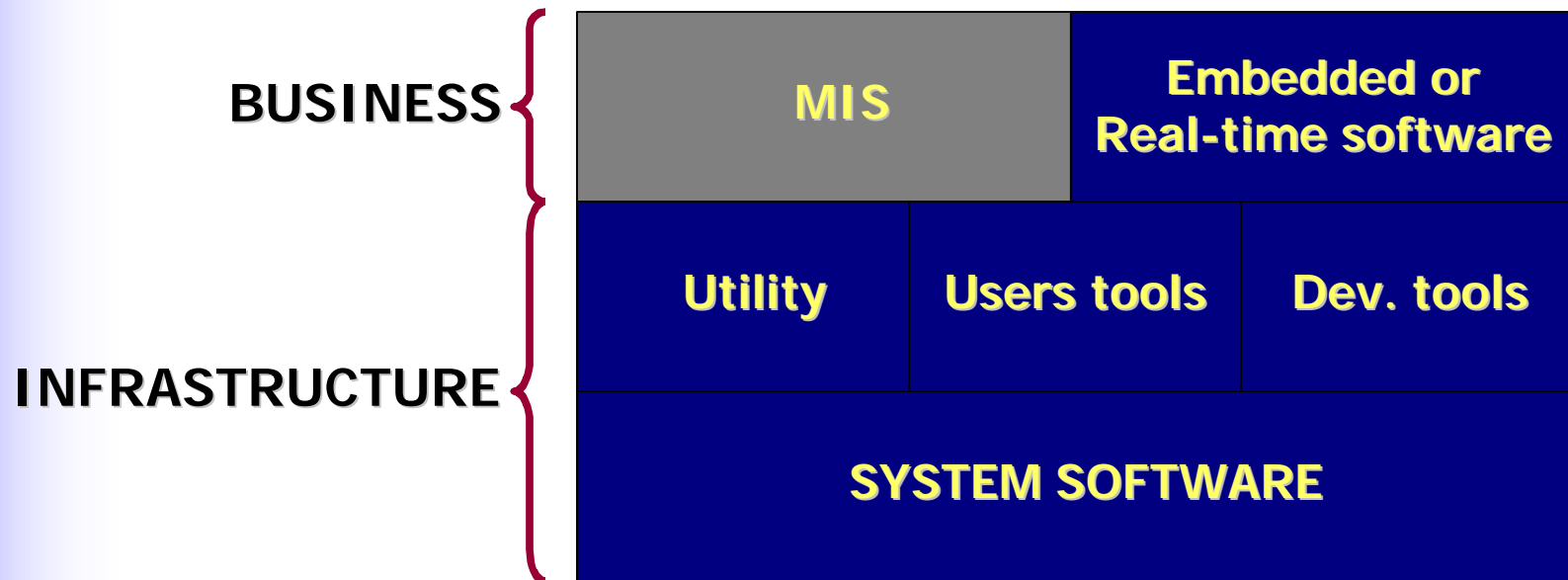


# ***Characteristics of real-time software***

---

- ⦿ **Different types of software**
- ⦿ **Real-time or embedded software**
- ⦿ **Limitations of IFPUG 4.0 Function Point**

# Different types of software



# ***Real-time or embedded software***

---

## **⊙ Timing**

- ✓ Tight constraints on the rate of execution and on the timing of tasks
- ✓ Explicit constraints on timing
- ✓ Dedicated components to manage timing
- ✓ Correctness of the result is linked to timing

## **⊙ Interaction with**

- ✓ Engineered devices
- ✓ People
- ✓ Other software applications

# ***Limitations of IFPUG 4.0 FP***

---

**Compared to MIS software...**

**USERS**

**People**  
**Other software**  
**Devices**

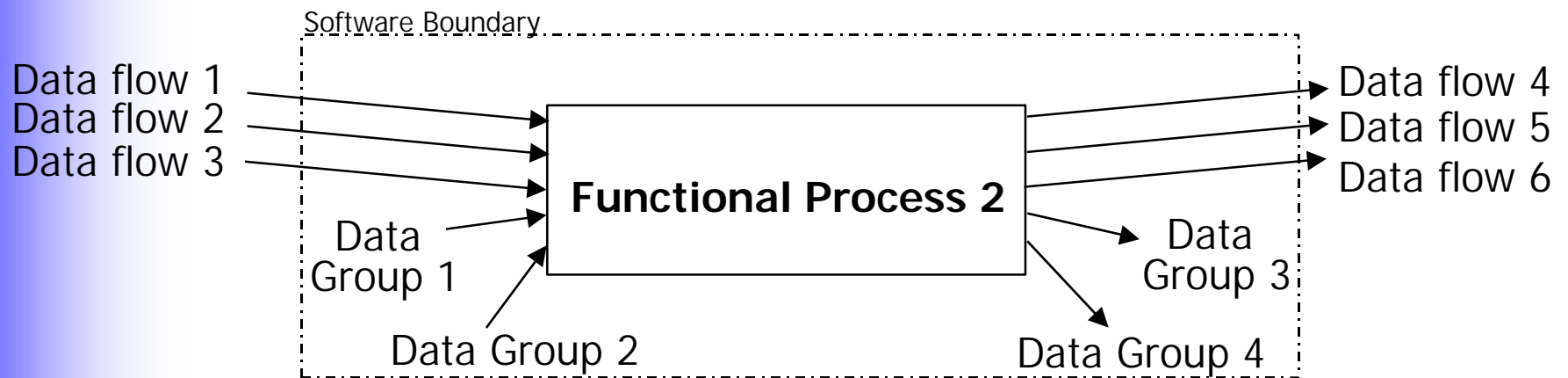
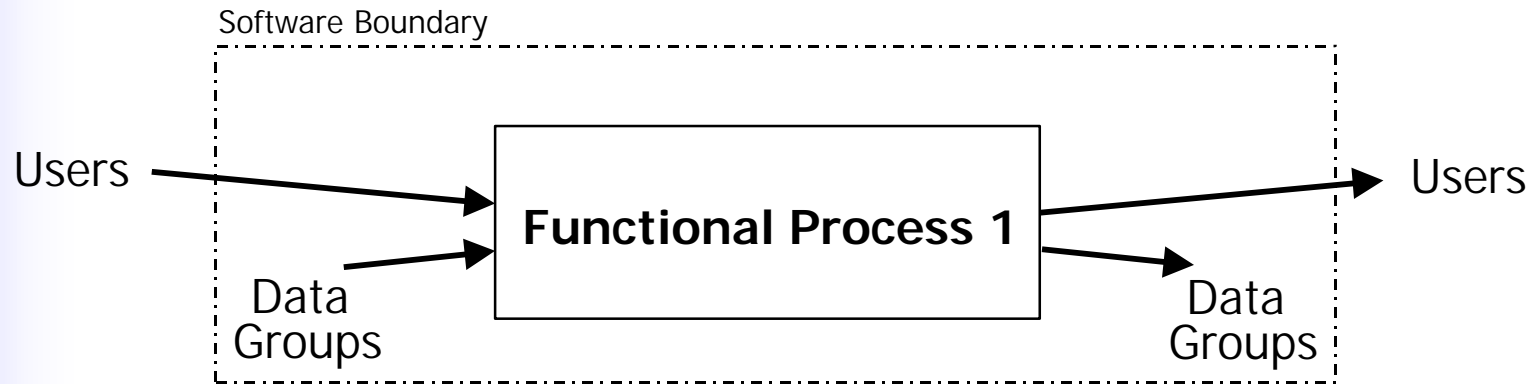
**DATA**

**Permanently stored (files, DB, ...)**  
**Not stored permanently (signals, ...)**

**PROCESSES**

**No. of sub-processes varies a lot**  
**Processes role is not easily classified as input, output or inquiry**

# Limitations of IFPUG 4.0 FP



**IFPUG Function Points (4.0), do not adequately measure the functional size of real-time software**

# *The measurement process model*

---

- ⊙ Overview of the model
- ⊙ Notes on measurement purpose...
- ⊙ Notes on measurement strategy...
- ⊙ Notes on documentation to be used...

# Overview of the model

Software to  
be measured

PHASE 1

- Identify software layers
- Identify boundary
- Identify data items
- Identify functional processes

FFP software  
model

- Identify sub-processes
- Assign points
- Aggregate results

PHASE 2

Software  
functional size

## ***Notes on measurement purpose***

---

- ⊙ Identify the **business issue** which needs to be addressed, for instance:
  - ✓ ...estimating the size of deliverables,
  - ✓ ...allocating supported functionality in maintenance,
  - ✓ ...measuring functionality required by business activities
  - ✓ ...establishing replacement costs of software portfolio,
  - ✓ ... assisting testing strategies layout,
  - ✓ ... assessing the size of development backlog,
  - ✓ ...establishing mandatory functionality for package evaluation.



# ***Notes on measurement purpose***

---

## ⊙ Determine:

- ✓ what questions need to be answered by the size measure,
- ✓ which software applications need to be sized
- ✓ what components of the software will be included or excluded

# ***Notes on measurement strategy***

---

## ⊙ Identify:

- ✓ **Which software** is to be sized,
- ✓ **How** the sizing will be performed,
- ✓ **Who** will do the sizing,
- ✓ **Who** will **assist** as the application expert,
- ✓ **Which Functional Size Measurement method** will be used e.g. Full Function Points (FFP) Version 1.0 or 2.0,
- ✓ **When** and **where** will the sizing take place,
- ✓ **Which software tools**, measurement forms, will be used.

# ***Notes on documentation to be used***

---

## ⊙ **Planned Applications** (New development)

- ✓ requirements specification
- ✓ logical design specification
- ✓ report layouts
- ✓ screen layouts
- ✓ logical data model

## ⊙ **Existing Applications** (Enhancements)

- ✓ all of the above plus
- ✓ user manual
- ✓ access to application online

# Measurement Procedures

---

## ◎ PHASE 1 - MAPPING

- a) Software layers, boundary and measurement scope
- b) Identifying data items
- c) Identifying functional processes

## ◎ PHASE 2 - MEASURING

- a) Identifying sub-processes
- b) Assigning points
- c) Aggregating results

# *Measurement Procedures*

---

## ⦿ PHASE 1 - **MAPPING**

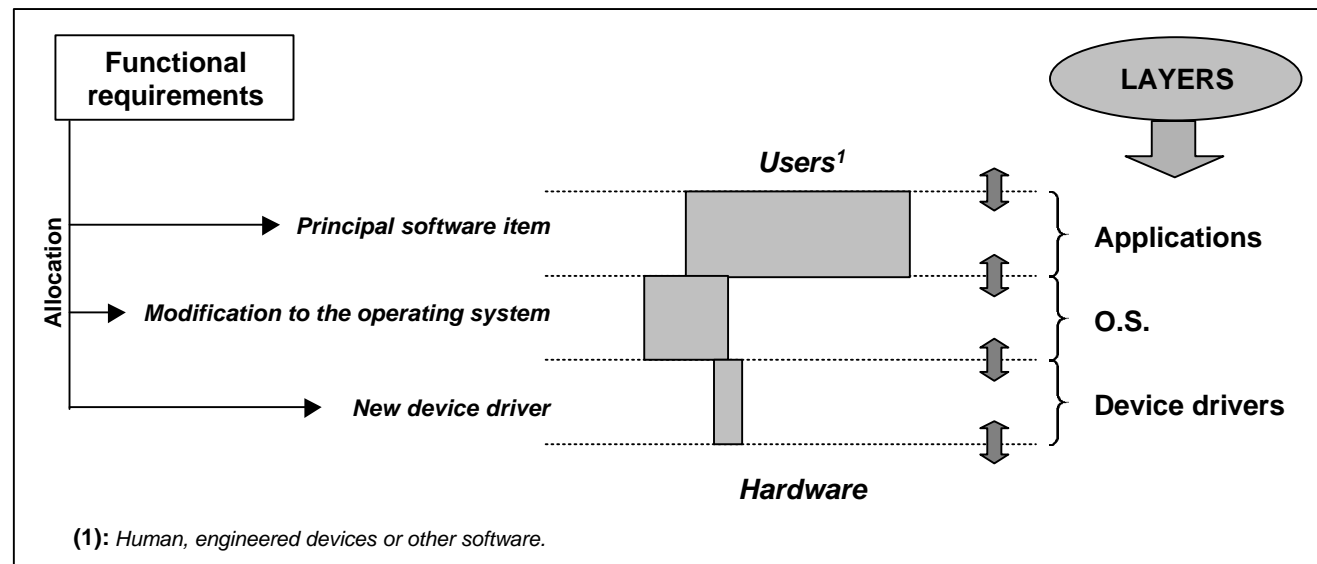
### **SOFTWARE LAYERS, BOUNDARY and MEASUREMENT SCOPE**

## 1 a) MAPPING...

# Software layers

V 2.0

## Concept of LAYER :



# Software boundary

---

V 2.0

## ⊙ Definition of **BOUNDARY** :

*'The boundary of a piece of software is the **conceptual frontier** between this piece and the environment into which it operates, as it is **perceived externally from the perspective of its users**.*

*The boundary allows the measurer to distinguish, without ambiguity, **what is included inside the measured software** from what is part of the measured software's operating environment.'*

**By convention, a boundary exists between adjacent layers.**

# Software boundary

---

**V 2.0**

## ⊙ Definition of **USER** :

*'Human beings, software or engineered devices which interact with the measured application.'*



# *Software boundary*

---

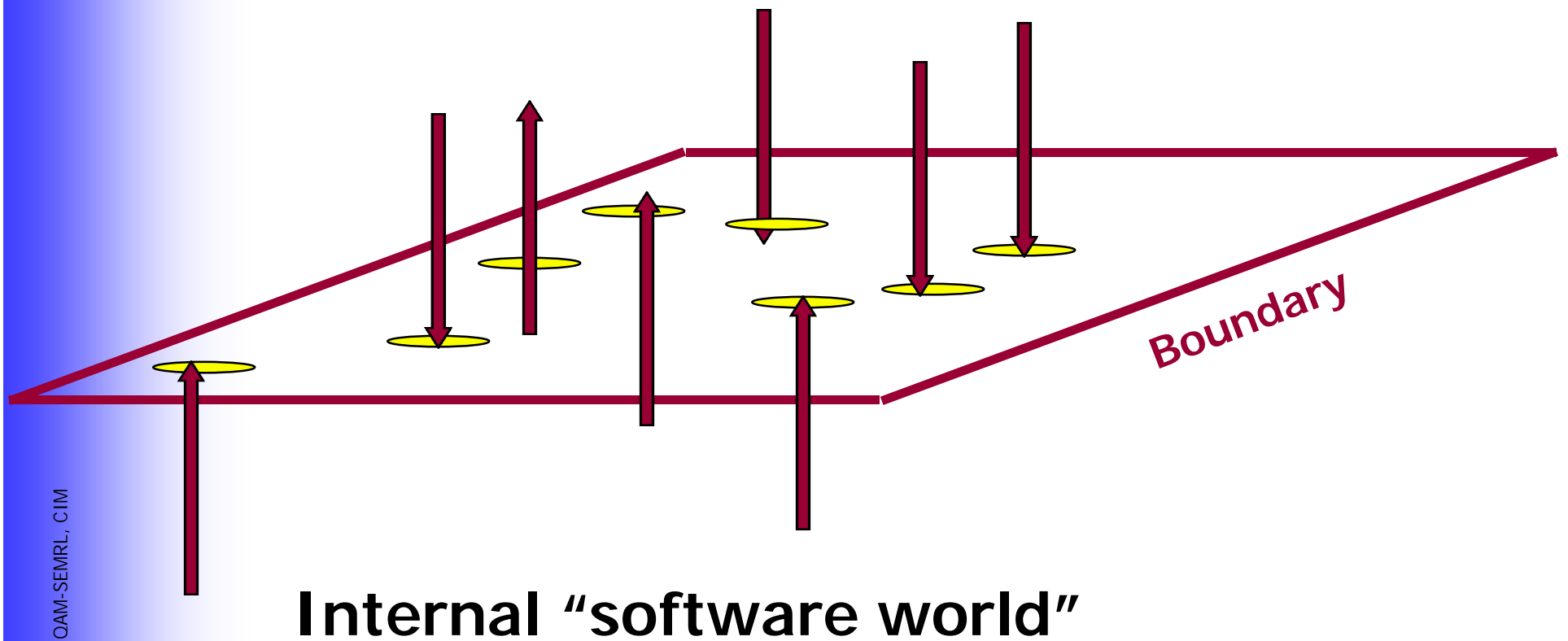
## ⊙ Boundary is:

- ✓ a conceptual 'membrane' through which data passes into and out of the software,
- ✓ external limits of the software,
- ✓ point where the software stops and the "external" users world starts.

1 a) MAPPING...

# Software boundary

External "users world"

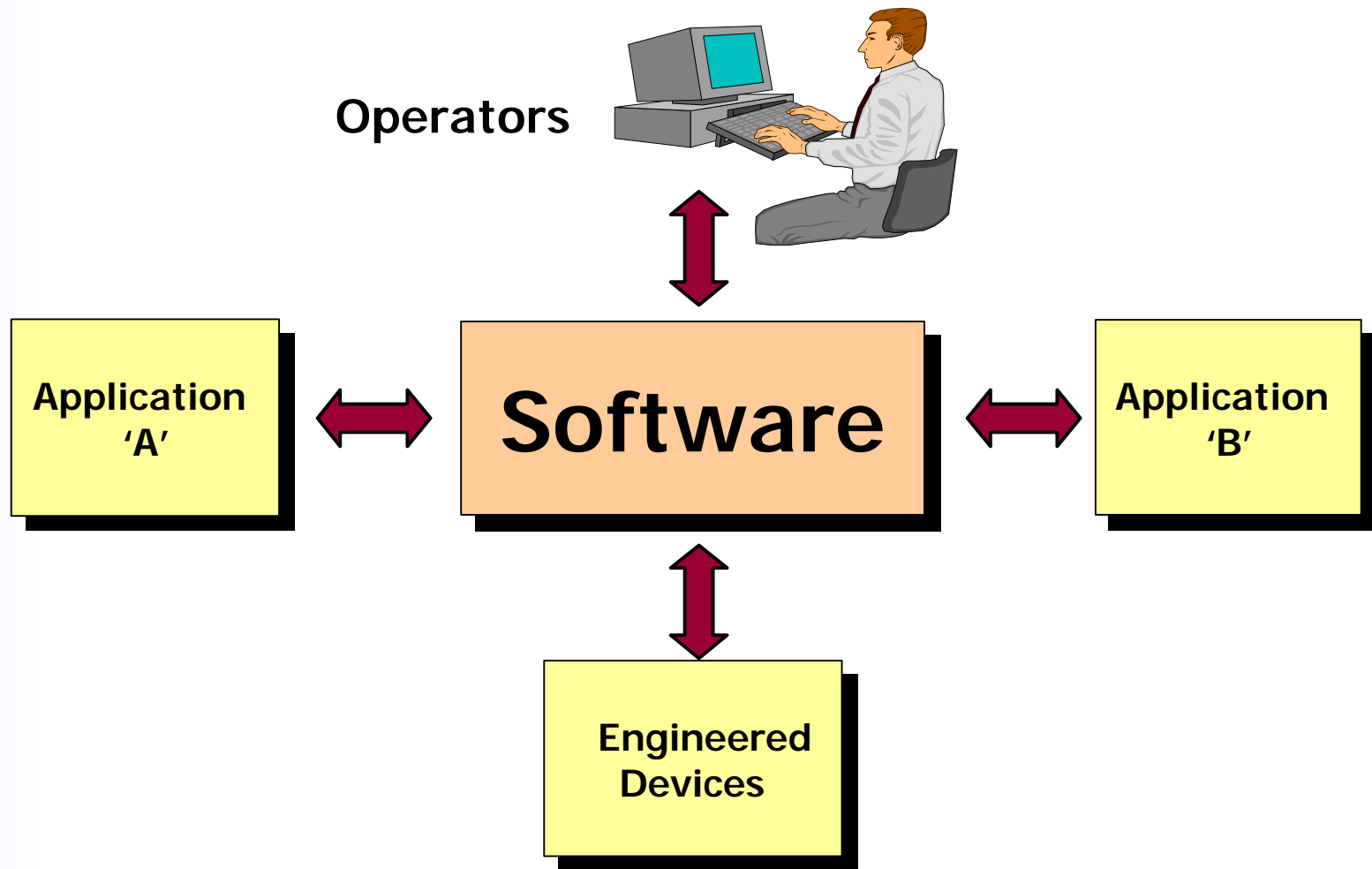


# Software boundary

---

- ⊙ Boundary may be illustrated on an application boundary diagram similar to a 'context diagram'
  
- ⊙ Identify all **major groups of data movements** between the boundary of the measured software and:
  - ✓ its human user operators,
  - ✓ the boundaries of other software
  - ✓ or engineered devices

# Software boundary



# 1 a) MAPPING...

# Software boundary

## Application Boundary



Lights  
Buzzers  
Usage Data  
Reports  
Alarms

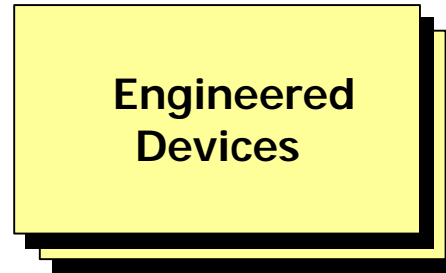


Configuration  
Parameters  
Status  
Control  
Actuators



Incoming Calls  
Sensors  
Alarms  
Status Parameters

Buttons Parameters  
Threshold Values  
Responses



# Measurement scope

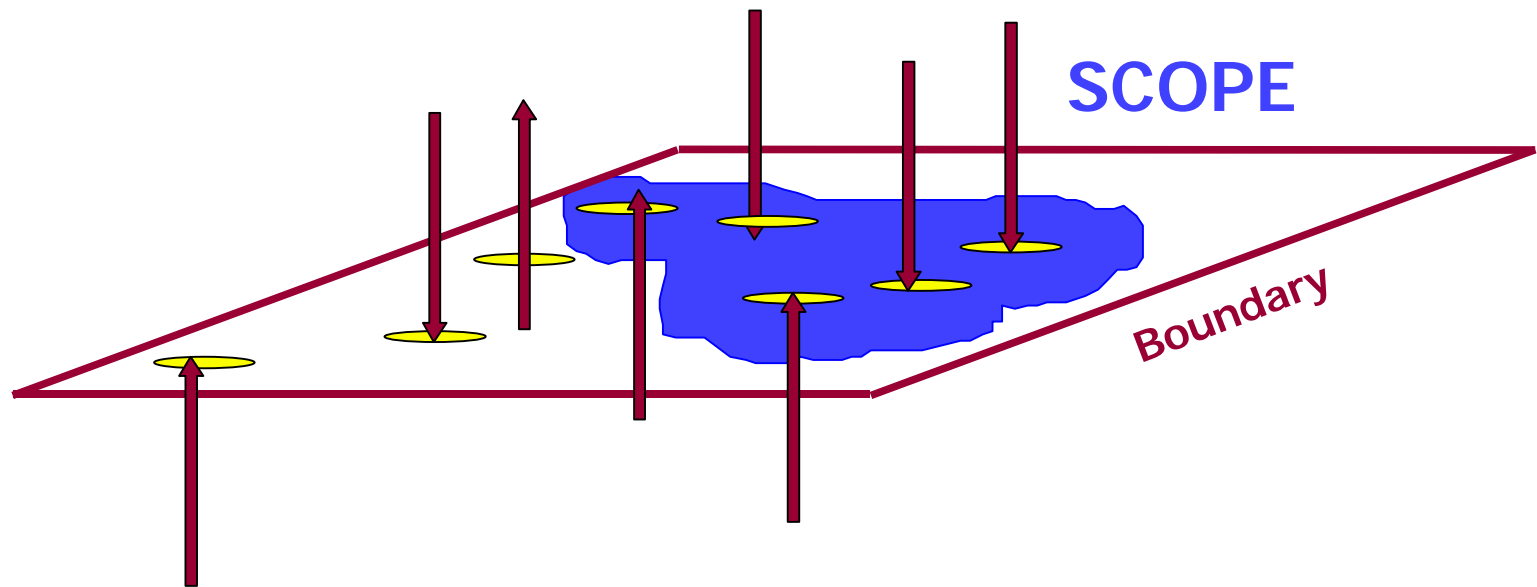
---

- ⊙ Definition of **SCOPE**:

*"The set of functional features, inside the application boundary, for which the size is to be measured"*

- ⊙ Measurement **SCOPE** is dictated by the **PURPOSE** of the measurement exercise.

# Measurement scope



**SCOPE defines a sub-set of the software to be sized**

# *Measurement Procedures*

---

## ◎ PHASE 1 - MAPPING

### IDENTIFYING DATA ITEMS



# *Identifying data items*

---

- ⊙ Key concepts
- ⊙ Identification
- ⊙ Summary

# *Identifying data items*

---

## Key concepts

- ⊙ **Data selection**

Which ones are mapped to the software model ?

- ⊙ **Data occurrences**

How are they organized ?

- ⊙ **Data activity**

How are data handled by the measured application ?

# Identifying data items

---

## Key concepts - Data selection

- ➔ If a piece of data is processed but not saved or reused, it is not permanent and it is not measured.
- ➔ If a piece of data is saved or reused , it is measured.
- ➔ A piece of data must exists for more than one **transaction**<sup>1</sup> to be measured.

(1) Note: a transaction correspond to ONE operation cycle of a functional process (more on this later)

# Identifying data items

---

## Key concepts - Data occurrence

- ⊙ **Multiple occurrences** are groups of data which can have more than one instance of the same type of record. In real-time, multiple occurrences have the same structure than the one found in MIS System.
  - ✓ Example: Flight record (black box)

# *Identifying data items*

---

## Key concepts - Data occurrence

- ⊙ **Single occurrence** are groups of data which have one and only one instance of the record.
  - ✓ Example: Data related to a time clock for a specific time.

# *Identifying data items*

---

## Key concepts - Data activity

### ⊙ Updated data (UCG)

e.g.: add, change, delete, populate, revise, update, assign, create ...

A data may be updated by more than one software application.

### ⊙ Read only data (RCG)

The data is consulted by the software being mapped without being updated.

The data may be updated by other software.

# *Identifying data items*

---

## Identification

- 1- Select all logically related groups of data that exists for more than one transaction.**
  - ✓ From a normalization point of view our practice suggests that a logically related group of data could be in second or third normal form.
  
- 2- Group data according to their structure**
  - ✓ Each multiple occurrences group is identified
  - ✓ Merge all single occurrence together into one group

# *Identifying data items*

---

## Identification

### 3- Determine the nature of data activity for each identified group

- ✓ A **UCG** is a group of data updated by the application being measured.
- ✓ An **RCG** is a group of data used, but not updated, by the application being measured.



# Identifying data items

---

## Identification

- 4- Verify that Updated Control Group (UCG) and Read-only Control Group (RCG)

### ARE

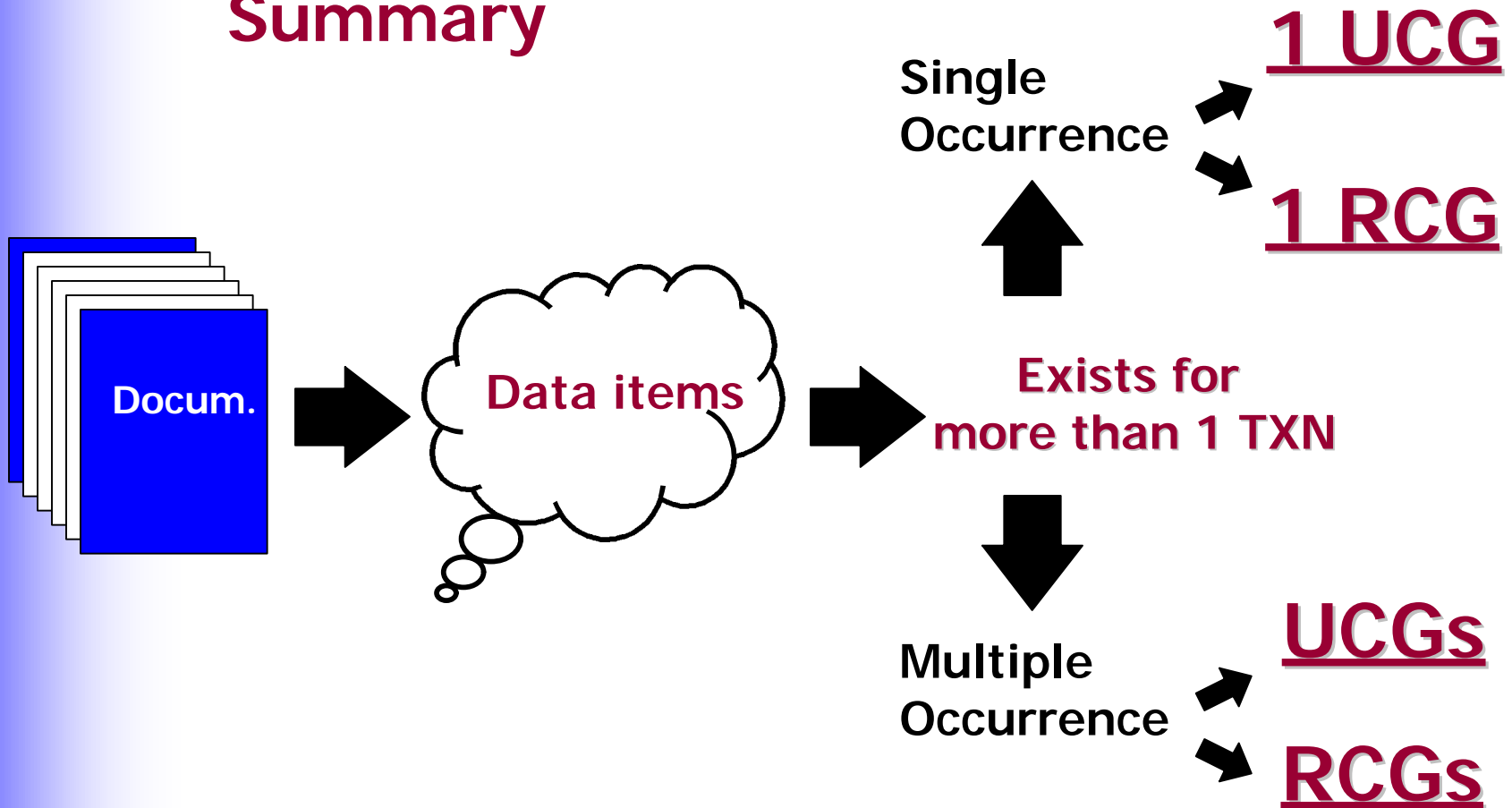
- ✓ Files maintained by the users

### BUT ARE NOT

- ✓ Sorting files
- ✓ Index files or secondary index
- ✓ Generated files sent to another application

# Identifying data items

## Summary



# *Measurement Procedures*

---

## ◎ PHASE 1 - MAPPING

### IDENTIFYING FUNCTIONAL PROCESSES

# *Identifying functional processes*

---

- ⊙ **Key concepts**
- ⊙ **Identification**
- ⊙ **Summary**

# *Identifying functional processes*

---

## Key concepts

- ⊙ Trigger
- ⊙ Functional process
- ⊙ Transaction

## *Identifying functional processes*

---

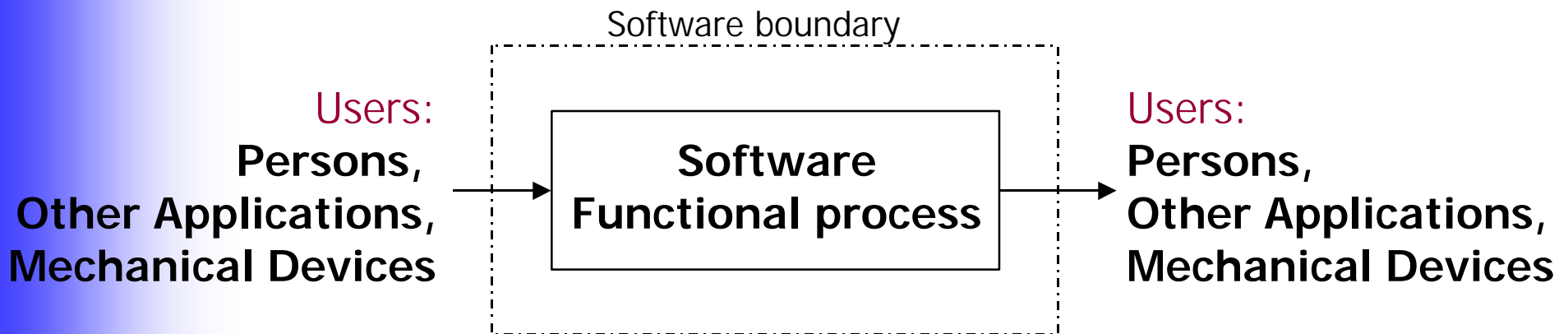
### Key concepts - Triggers

- ⊙ An event **initiating** a functional process from a the perspective of the software users,
- ⊙ An event occurring outside the software boundary,
- ⊙ When an event occurs, data usually enters the software boundary,
- ⊙ Clocks and timing events can be triggers.

# Identifying functional processes

## Key concepts - Functional process

“A set of operations or activities which acts on input data to produce a result.”



## *Identifying functional processes*

---

### Key concepts - Transactions

- ⊙ A **transaction** is an instance of a functional process,
- ⊙ A transaction includes all processing associated with the occurrence of an external trigger.

**Example:** in a watch, each tick of the timing crystal is a trigger. All processing associated with each new tick is a separate transaction.



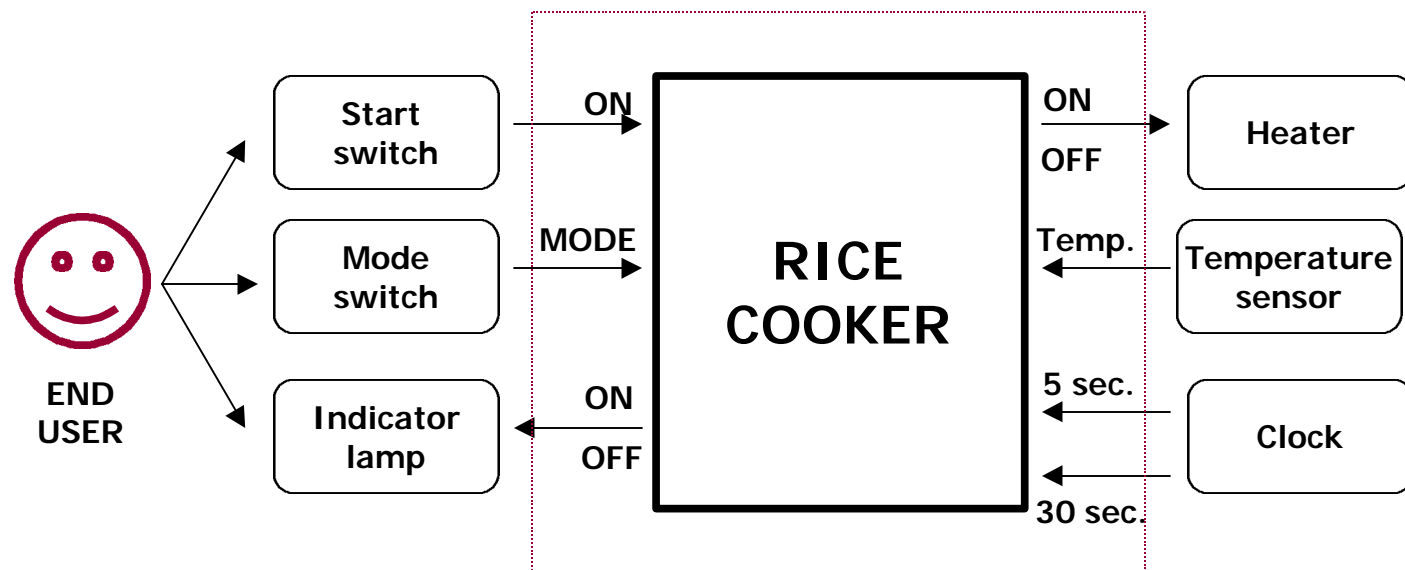
# Exercise

## Using the Case Study document:

- What is the purpose of the measurement exercise ?
- Identify the boundary of the application
- Identify the data items
- Identify the functional processes

# Exercise

- **PURPOSE:** Practice FFP measurement
- **BOUNDARY:**



# Exercise

	(1)	(2)	(3)
<b>TRIGGER</b>	Temperature data	Selected cooking mode Elapsed time Target temperature Indicator lamp (ON/OFF) Heater (ON/OFF)	Mode switch Start switch Temperature sensor 30 sec. clock signal 5 sec. clock signal
<b>FUNCTIONAL PROCESSES</b>			
<b>MODE SWITCH PRESSED</b>			
MODE SELECTION CONTROL			
<b>START SWITCH PRESSED</b>			
ELAPSED TIME CONTROL			
<b>30 sec. CLOCK SIGNAL</b>			
TARGET TEMPERATURE CONTROL			
<b>5 sec. CLOCK SIGNAL</b>			
COOKING TEMPERATURE CONTROL			

- (1) Multiple occurrence RCG
- (2) Single occurrence UCG
- (3) Single occurrence RCG

# *Measurement Procedures*

---

## ◎ PHASE 2 - MEASURING

### IDENTIFYING SUB-PROCESSES

## *Identifying sub-processes*

---

### Key concepts - Sub-processes

- ⊙ An FFP sub-process is a functional **elementary data movement** occurring during the execution of a functional process.
- ⊙ There are **four types** of FFP sub-process: entry, exit, read and write.
- ⊙ The object of an elementary data movement is either a multiple occurrence data group or a single occurrence data attribute.
- ⊙ An FFP sub-process is equivalent to **ISO Basic Functional Component types (BFC)**.

## *Identifying sub-processes*

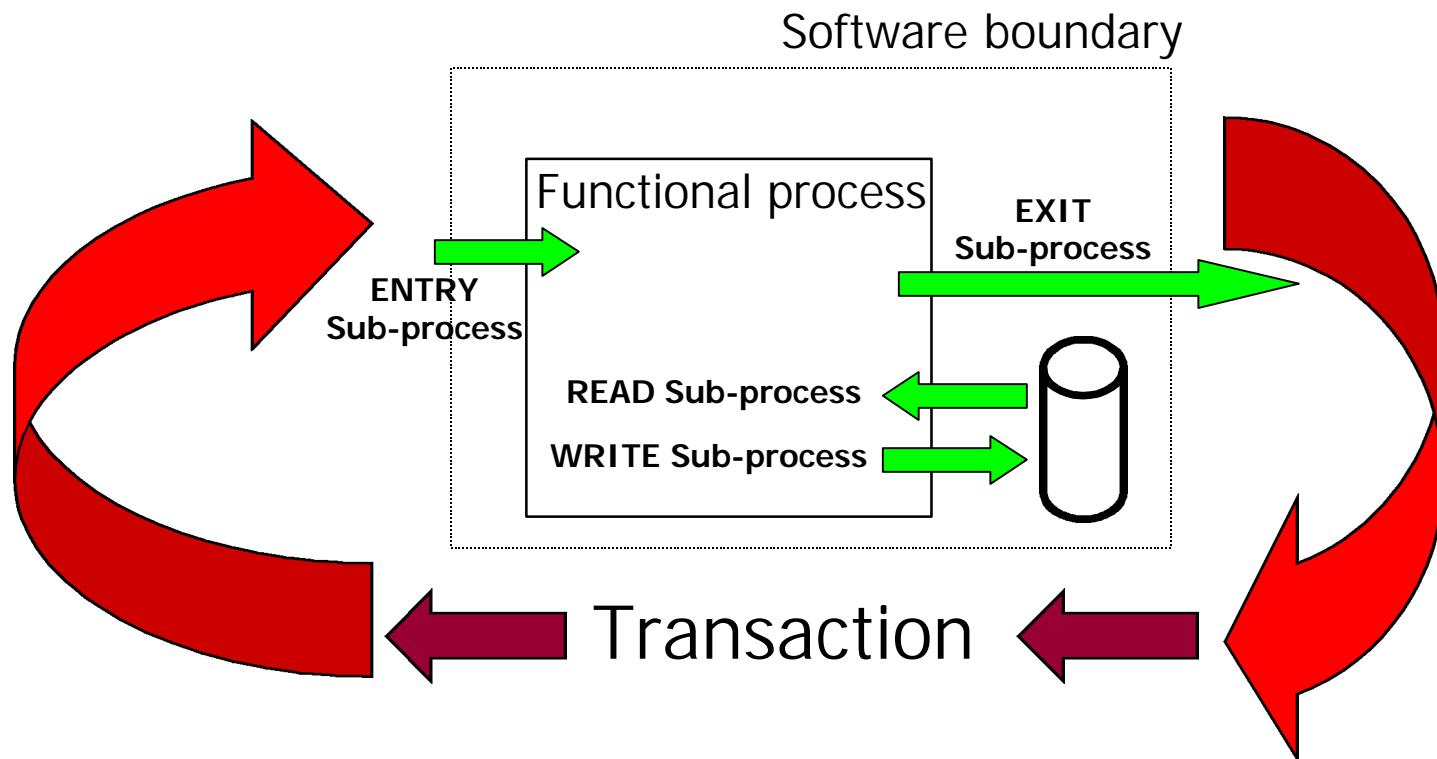
---

### Key concepts - Sub-processes

- ⊙ Identified from a functional perspective,
- ⊙ Single sub-processes; duplicates removed,
- ⊙ A sub-process moves only one group of data.

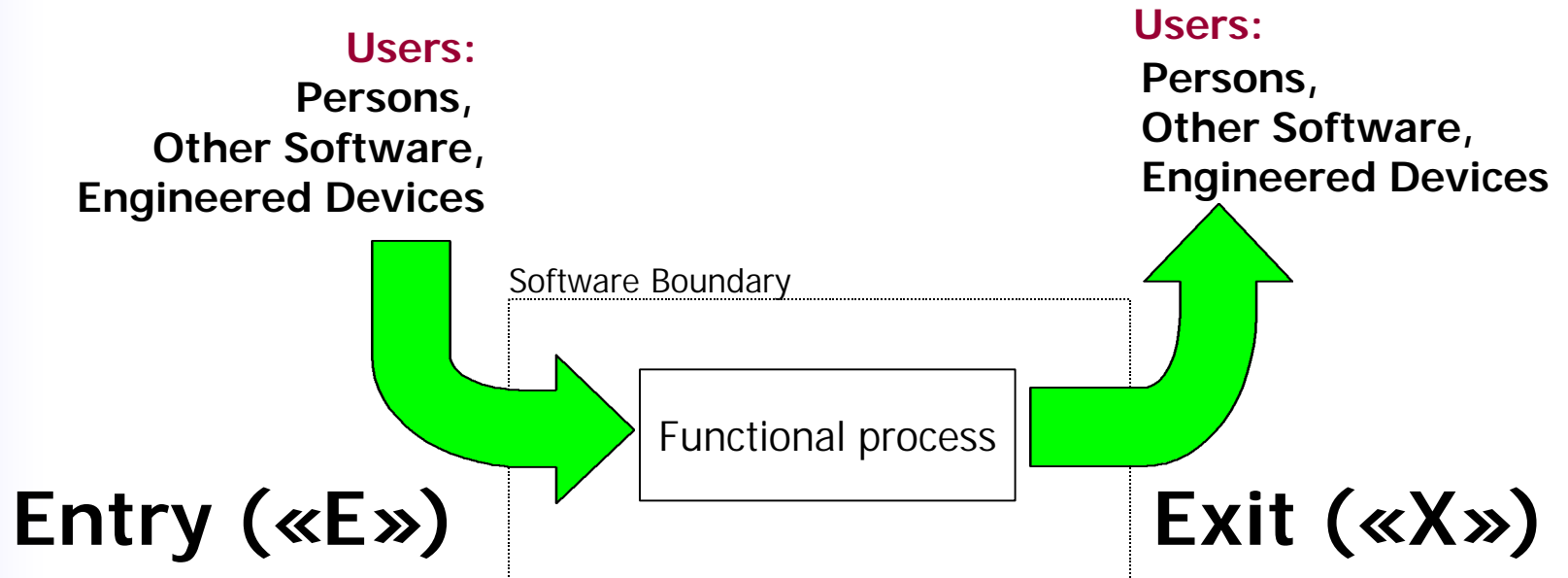
# Identifying sub-processes

## Key concepts - 4 classes of sub-processes



# Identifying sub-processes

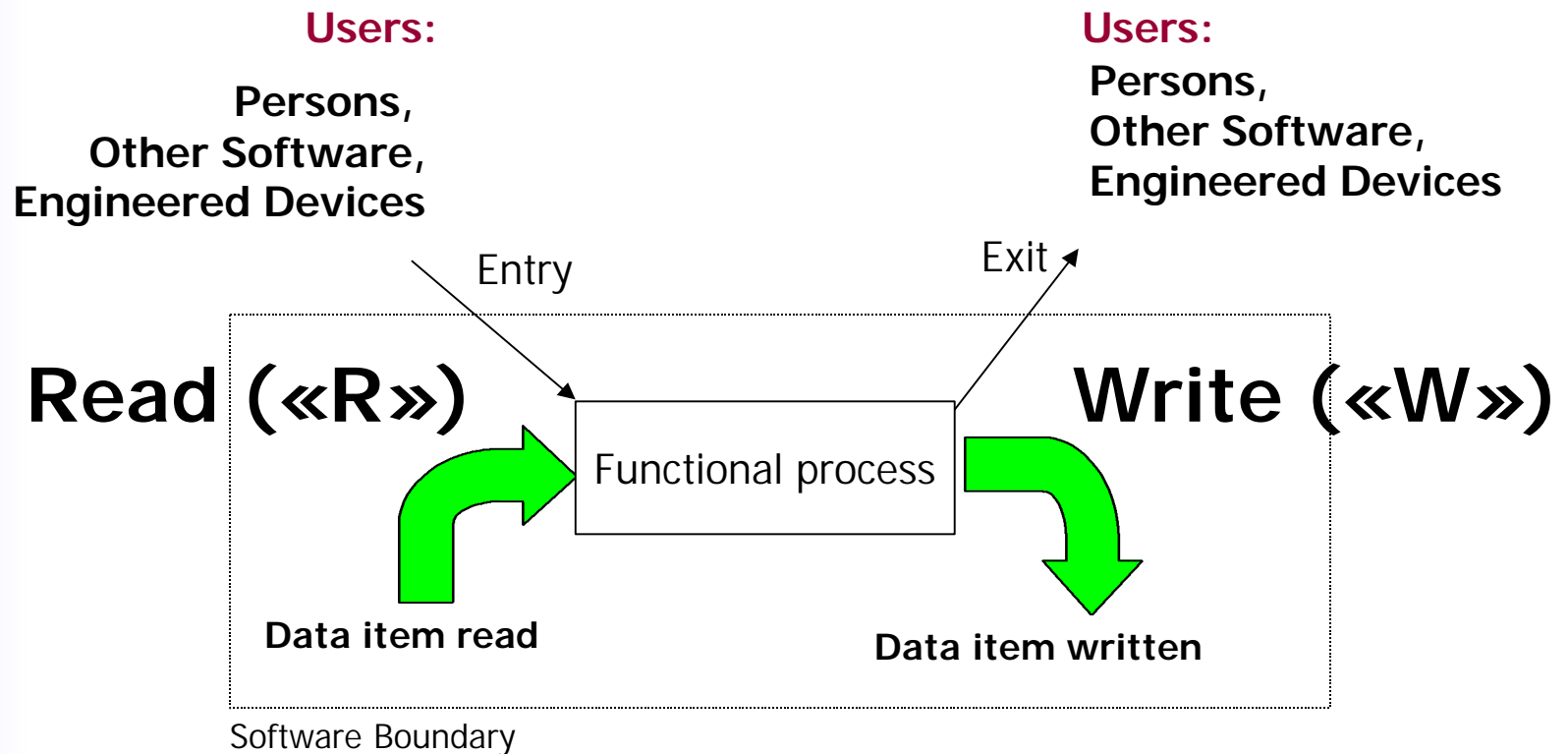
## Key concepts - 4 classes of sub-processes





# Identifying sub-processes

## Key concepts - 4 classes of sub-processes



## *Identifying sub-processes*

---

### Identification rules: Entry

- ⊙ The sub-process **receives** a data item from outside the software boundary,
- ⊙ The sub-process is associated with **only one** data item,
- ⊙ The sub-process does not **exit, read, or write** data items,
- ⊙ The sub-process is unique: processing and data items identified are different from other Entries within the same functional process.

## *Identifying sub-processes*

---

### Identification rules: Exit

- ⊙ The sub-process **sends** data outside of the software boundary.
- ⊙ The sub-process sends **only one** data item.
- ⊙ The sub-process does not **receive, read, or write** data item.
- ⊙ The sub-process is unique: processing and data items identified are different from other Exits in the same functional process.

## *Identifying sub-processes*

---

### Identification rules: Read

- ⊙ The sub-process **reads** a data item.
- ⊙ The sub-process reads **only one** data item.
- ⊙ The sub-process does not receive, exit, or **write** data items.
- ⊙ The sub-process is unique: processing and data items identified are different from other Reads in the same functional process.

## *Identifying sub-processes*

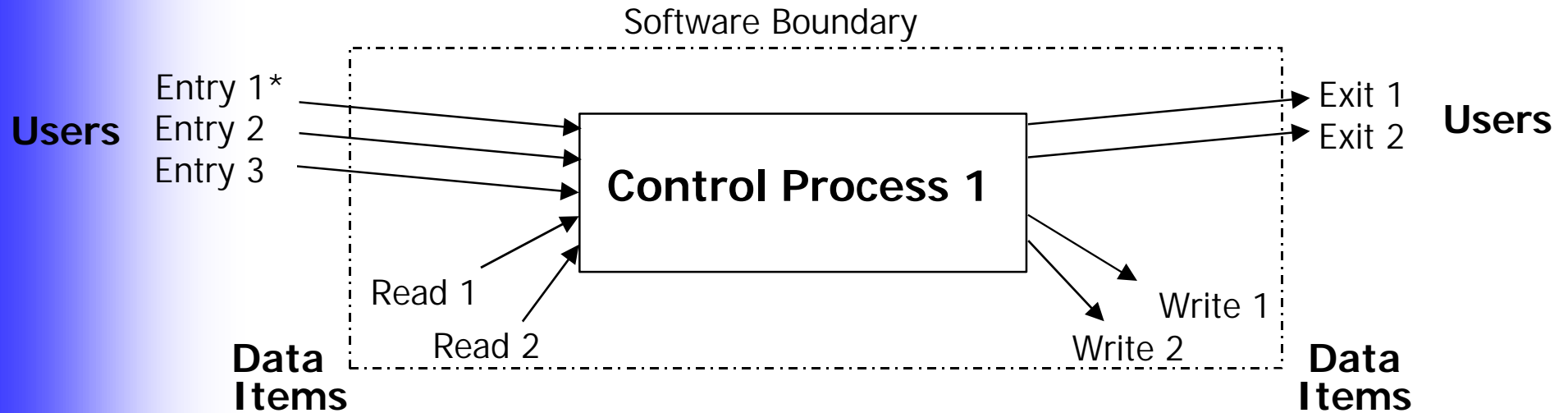
---

### Identification rules: Write

- ⊙ The sub-process **writes** to a data item.
- ⊙ The sub-process writes to **only one** data item.
- ⊙ The sub-process does not **receive**, **exit**, or **read** data items.
- ⊙ The sub-process is unique: processing and data items identified are different from other Writes in the same functional process.

# Identifying sub-processes

## Summary

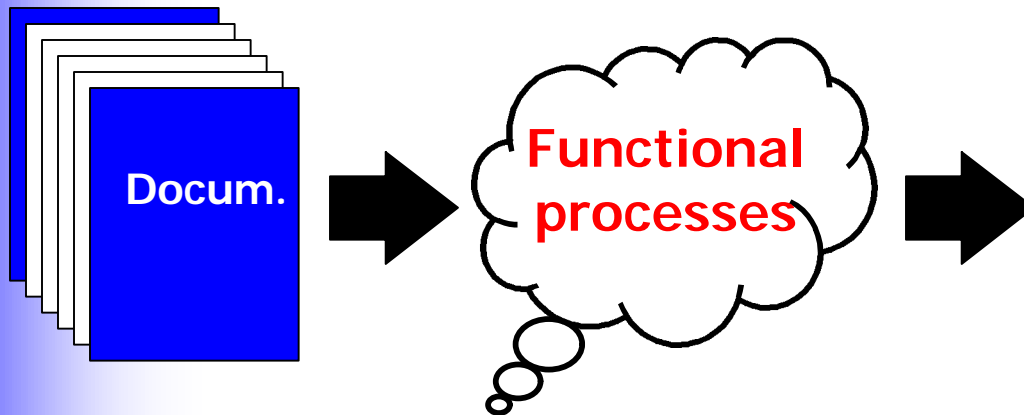


**Each arrow is a sub-process.**

\* Entry 1 is the trigger

# Identifying sub-processes

## Summary



- **Trigger 1**
  - Functional process 1
    - Sub process 1.1
    - Sub process 1.2
    - ...
  - Functional process 2
    - Sub process 2.1
    - Sub process 2.2
- **Trigger 2**
  - Functional process 1
    - Sub process 1.1
    - ...

# Exercise

## Using the Case Study document:

- Identify the sub-processes



# Exercise

TRIGGER	FUNCTIONAL PROCESSES	(1)	(2)	(3)
		Temperature data	Selected cooking mode Elapsed time Target temperature Indicator lamp (ON/OFF) Heater (ON/OFF)	Mode switch Start switch Temperature sensor 30 sec. clock signal 5 sec. clock signal
<b>MODE SWITCH PRESSED</b>	MODE SELECTION CONTROL	<input type="checkbox"/>	W <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	E <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
<b>START SWITCH PRESSED</b>	ELAPSED TIME CONTROL	R <input type="checkbox"/>	R W <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	<input type="checkbox"/> E E <input type="checkbox"/> <input type="checkbox"/>
<b>30 sec. CLOCK SIGNAL</b>	TARGET TEMPERATURE CONTROL	R <input type="checkbox"/>	R R W X <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> E <input type="checkbox"/>
<b>5 sec. CLOCK SIGNAL</b>	COOKING TEMPERATURE CONTROL	<input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> R <input type="checkbox"/> X <input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> E <input type="checkbox"/> E <input type="checkbox"/>

- (1) Multiple occurrence RCG
- (2) Single occurrence UCG
- (3) Single occurrence RCG

# *Measurement Procedures*

---

## ◎ PHASE 2 - MEASURING

### ASSIGNING POINTS

## ***Assigning points***

---

	<b>V. 1.0</b>	<b>V 2.0</b>
<b>Data items</b>	<b>Yes</b>	<b>To be determined</b>
<b>Functional processes</b>	<b>Yes</b>	<b>Yes</b>

## ***Assigning points***

---

- ⊙ **Data items (v. 1.0 only):**
  - ⊙ key concepts
  - ⊙ measurement functions
  - ⊙ Quick validation tips
- ⊙ **Functional processes:**
  - ⊙ measurement functions
  - ⊙ Quick validation tips

## *Data items: key concepts*

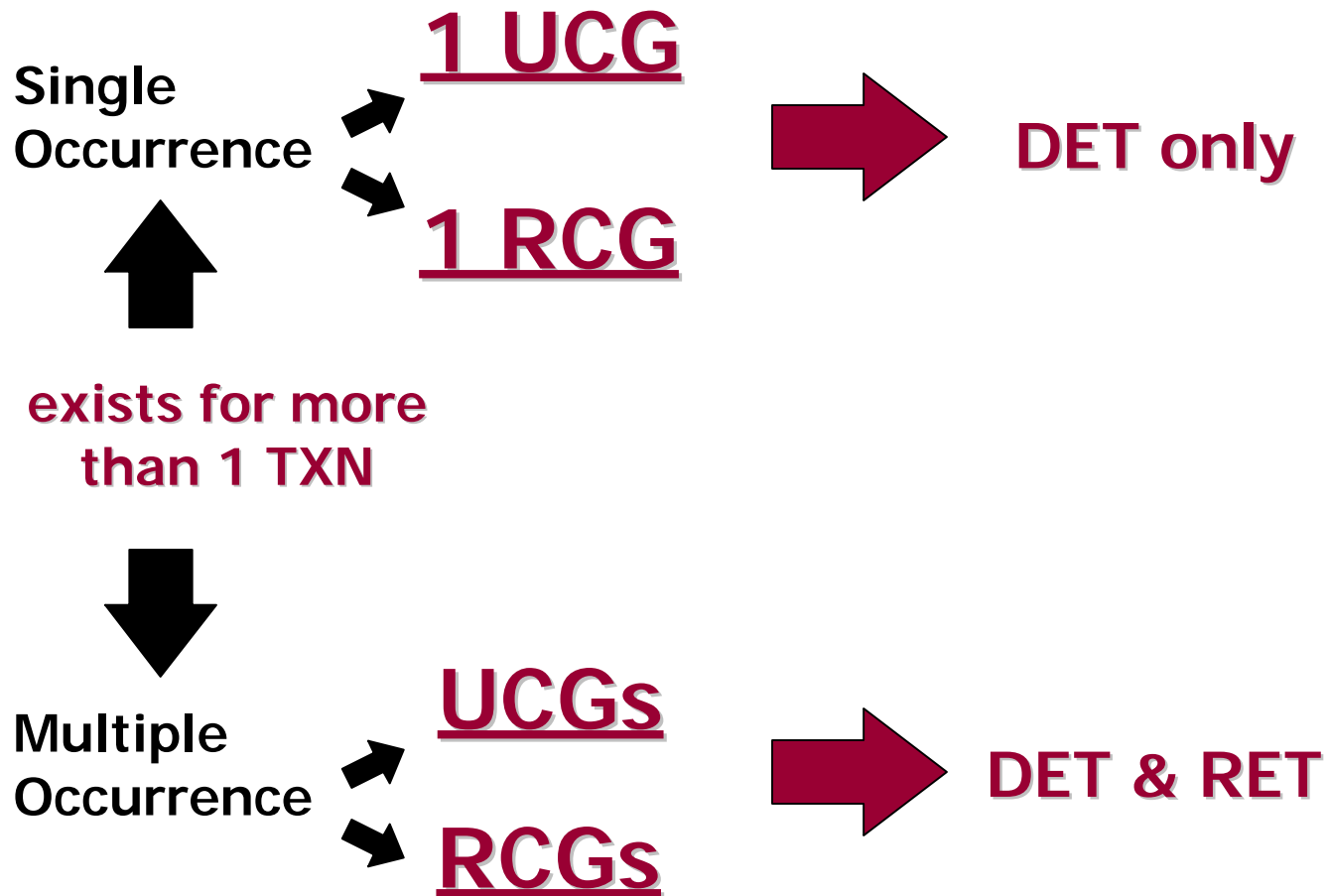
---

Points are assigned to data as a function of two characteristics:

**DET:** The number of data elements

**RET:** The number of user recognizable subgroup of data elements

# Data items: key concepts



## *Measurement function*

---

### Single occurrence Updated data (UCG):

- ⊙ Point assignment is based on the number of data element types (DET)
- ⊙ Points = (number of DET / 5) + 5

**Note:** There is only one single occurrence UCG within a piece of software. It includes all the single occurrence updated values within the software being measured.

## *Measurement function*

---

### Single occurrence Read-Only Data (RCG):

- ⊙ Point assignment is based on the number of data element types (DET)
- ⊙ Points = number of DET / 5

**Note:** There is only one single occurrence RCG within a piece of software. It includes all the single occurrence read-only values within the software being measured.



# Measurement function

---

## Multiple occurrence RCG and UCG:

DETs \ RETs	1 - 19	20 - 50	51 +
1	L	L	A
2 - 5	L	A	H
6 +	A	H	H

## Measurement function

---

### Multiple occurrence UCG and RCG:

	UCG	RCG
L = Low	7	5
A = Average	10	7
H = High	15	10

## *Quick validation tips*

---

### Check if:

- ⊙ All data exist for more than one transaction,
- ⊙ Repeated fields have been measured only once,
- ⊙ Data updated in more than one software has been measured in each software

## ***Functional processes: measurement function***

---

### **V. 1.0**

- ⊙ Based on the number of DET moved by the sub-process:
  - ⊙ 1 to 19 DET moved: 1 point,
  - ⊙ 20 to 50 DET moved: 2 points,
  - ⊙ 51 DET + moved: 3 points.

### **V 2.0**

- ⊙ Yardstick: **1 FFP = 1 elementary data movement**,
- ⊙ Therefore all identified sub-process received 1 point.

## Quick validation tips

---

- ⊙ Check that each **functional process** :
  - ✓ has at least one Entry (**E**),
  - ✓ has at least one Exit (**X**) or one Write (**W**),
  - ✓ does not have duplicate sub-processes.

# *Measurement Procedures*

---

## ⊙ PHASE 2 - MEASURING

### AGGREGATING RESULTS

## Aggregating results

---

- ⊙ FFP results can be aggregated at the desired level of detail by arithmetically adding the points assigned to sub-processes.
- ⊙ There is no upper limit to the functional size of a functional process.
- ⊙ The aggregation function is fully scalable when using **V 2.0**

# Exercise

Using the Case Study document:

- Calculate the functional size of the Rice Cooker



# Exercise

## Functional processes

### TRIGGER FUNCTIONAL PROCESSES

#### MODE SWITCH PRESSED

MODE SELECTION CONTROL

	W					E					1			1
--	---	--	--	--	--	---	--	--	--	--	---	--	--	---

#### START SWITCH PRESSED

ELAPSED TIME CONTROL

R	R	W					E	E			2		2	1
---	---	---	--	--	--	--	---	---	--	--	---	--	---	---

#### 30 sec. CLOCK SIGNAL

TARGET TEMPERATURE CONTROL

R	R	R	W	X					E		1	1	3	1
---	---	---	---	---	--	--	--	--	---	--	---	---	---	---

#### 5 sec. CLOCK SIGNAL

COOKING TEMPERATURE CONTROL

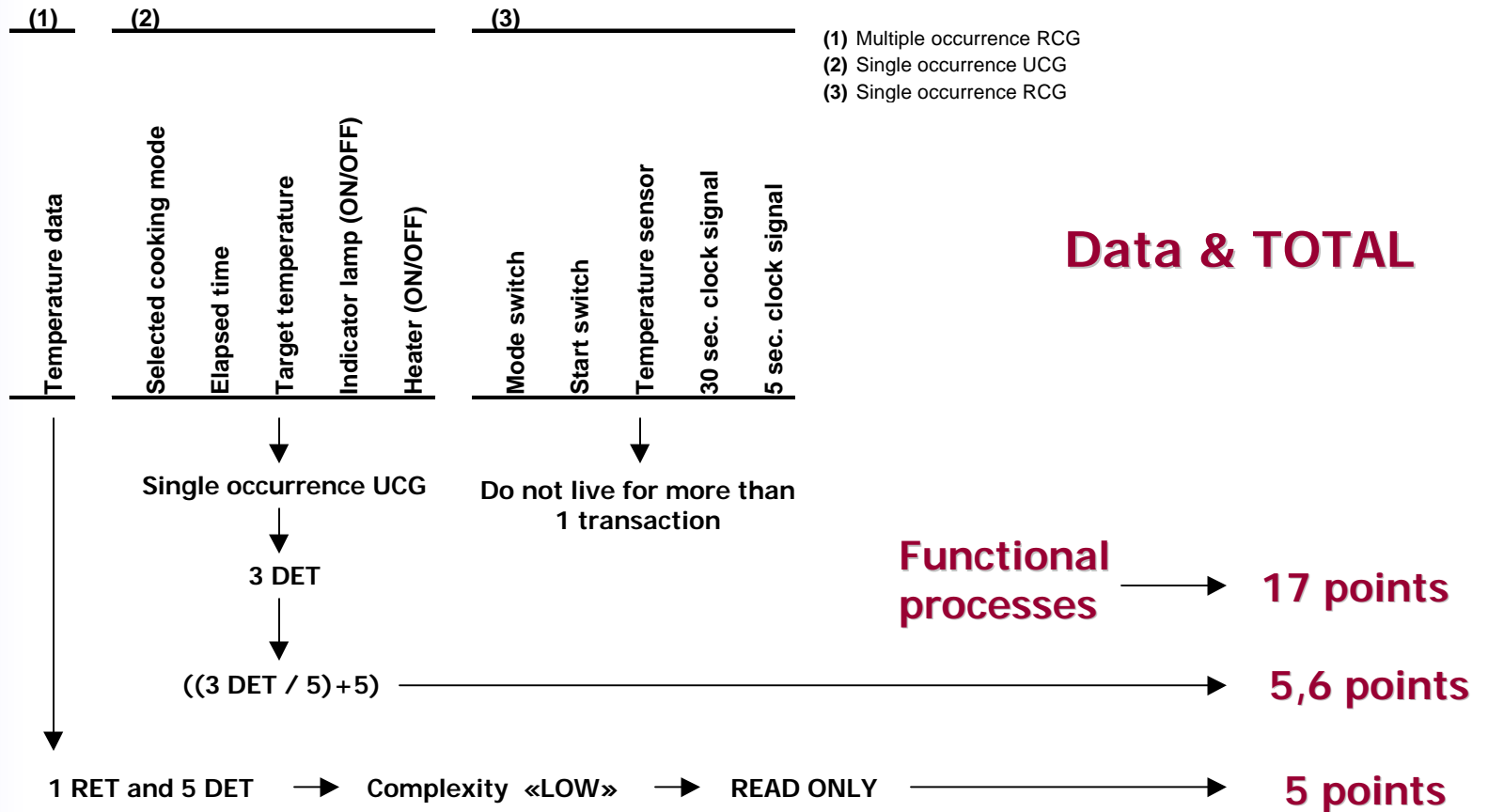
			R		X			E		E	2	1	1	
--	--	--	---	--	---	--	--	---	--	---	---	---	---	--

6	2	6	3
---	---	---	---

- (1) Multiple occurrence RCG
- (2) Single occurrence UCG
- (3) Single occurrence RCG

17 FFP

# Exercise



**Functional size: 27,6 FFP (v 1.0)**

# *Overview of field tests results*

# *Overview of field tests results*

---

- ⊙ Sources of data
- ⊙ First set: comparing FPA and FFP
- ⊙ Second set: relevance and usability
- ⊙ Third set: further comparisons FPA/FFP

# *First set*

---

- ⊙ Conducted by the research team in 1997,
- ⊙ 3 RT or embedded products measured,
- ⊙ 2 industrial partners participated,
- ⊙ **GOAL: Compare FFP with FPA (IFPUG 4.0)**

## First set

### Results...

	PRODUCT 1		PRODUCT 2		PRODUCT 3	
	TXN <sup>3</sup>	Points	TXN <sup>3</sup>	Points	TXN <sup>3</sup>	Points
<b>FPA<sup>1</sup></b>	54	256	9	38	32	123
<b>FFP<sup>2</sup></b>	753	777	40	46	468	479

Note 1: Using IFPUG 4.0 CPM, processes only

Note 2: Using FFP 1.0 CPM, processes only

Note 3: Number of processing transactions for which points are assigned

# *First set*

---

## Observations:

- ⊙ FFP results close to FPA when processes contain small number of sub processes,
- ⊙ FFP yield larger size measures when processes contain large number of sub processes,
- ⊙ Both methods require similar measurement effort

# *Second set*

---

- ⊙ Conducted without assistance from the research team in 1997,
- ⊙ Operational real-time products measured,
- ⊙ 1 industrial partner,
- ⊙ GOAL: Evaluate FFP for relevance and usability



# Second set

---

## Observations:

- ⊙ **Functional coverage established at 97%, based on expected number of functions to be measured.**
  
- ⊙ **Concepts and procedures are:**
  - ✓ Clear,
  - ✓ Easy to understand,
  - ✓ Usable without assistance of specialists

# *Third set*

---

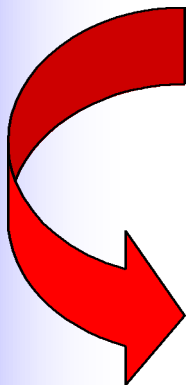
- ⊙ **4 industrial partners in North-America and Australia participated,**
  
- ⊙ **10 software products measured:**
  - ✓ 8 products related to the telecom business
  - ✓ 1 product related to power utility
  - ✓ 1 product related to the military sector
  
- ⊙ **All products measured by the same individual (CFPS, 12 years exp. in FSM)**

# Third set

**1<sup>st</sup> GOAL:** Compare IFPUG 4.0 and FFP

## RESULTS

Product	Type	FPA size	FFP size
A	Real-Time	210	794
B	Real-Time	115	183
C	Real-Time	N / A	2 604
D	Real-Time	43	318
<b>E</b>	<b>Mostly MIS</b>	<b>764</b>	<b>791</b>
F	MIS (batch)	272	676
<b>G</b>	<b>MIS</b>	<b>878</b>	<b>896</b>



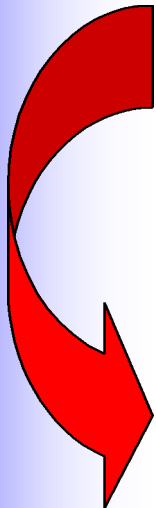
**Size is similar when measuring typical MIS software products**

## Third set

**1<sup>st</sup> GOAL:** Compare IFPUG 4.0 and FFP

### RESULTS

Product	Type	FPA size	FFP size
A	Real-Time	210	794
B	Real-Time	115	183
<b>C</b>	<b>Real-Time</b>	<b>N/A</b>	<b>2 604</b>
D	Real-Time	43	318
E	Mostly MIS	764	791
F	MIS (batch)	272	676
G	MIS	878	896

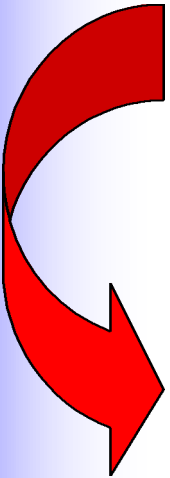


**One real-time software could only be sized with FFP**

# Third set

**1<sup>st</sup> GOAL:** Compare IFPUG 4.0 FPA and FFP

## RESULTS



Product	Type	FPA size	FFP size
<b>A</b>	<b>Real-Time</b>	<b>210</b>	<b>794</b>
<b>B</b>	<b>Real-Time</b>	<b>115</b>	<b>183</b>
C	Real-Time	N / A	2 604
<b>D</b>	<b>Real-Time</b>	<b>43</b>	<b>318</b>
E	Mostly MIS	764	791
<b>F</b>	<b>MIS (batch)</b>	<b>272</b>	<b>676</b>
G	MIS	878	896

Larger functional size for software products with numerous R-T processes (A, B and D); even for MIS with fewer direct user interactions (F).

# Third set

---

**1<sup>st</sup> GOAL:** Compare IFPUG 4.0 and FFP

What does it mean ?

	MIS product	RT product
FPA	200	200
FFP	~ 200	>> 200

Obviously, when considering RT products, FFP is measuring functionality that is not measured by IFPUG 4.0.

# Third set

## 2<sup>nd</sup> GOAL: Explore key economic ratios

### RESULTS

These 3 software products are all R-T software

Product	Size (FFP)	Effort (ph)	Duration (mth)	Unit effort (ph/FFP)	Sched. del. Rate (FFP/mth)
H	205	3 913	26	19	8
I	138	6 580	16	48	9
J	198	7 448	14	38	14

Until further data is available to allow statistically significant analysis, these should be interpreted as “order of magnitude” figures.

# *Conclusion*



# Conclusion

---

- ⊙ International recognition
- ⊙ Benchmarking your results
- ⊙ The future of Full Function Points
- ⊙ Available resources
- ⊙ Final remarks
- ⊙ Acknowledgements

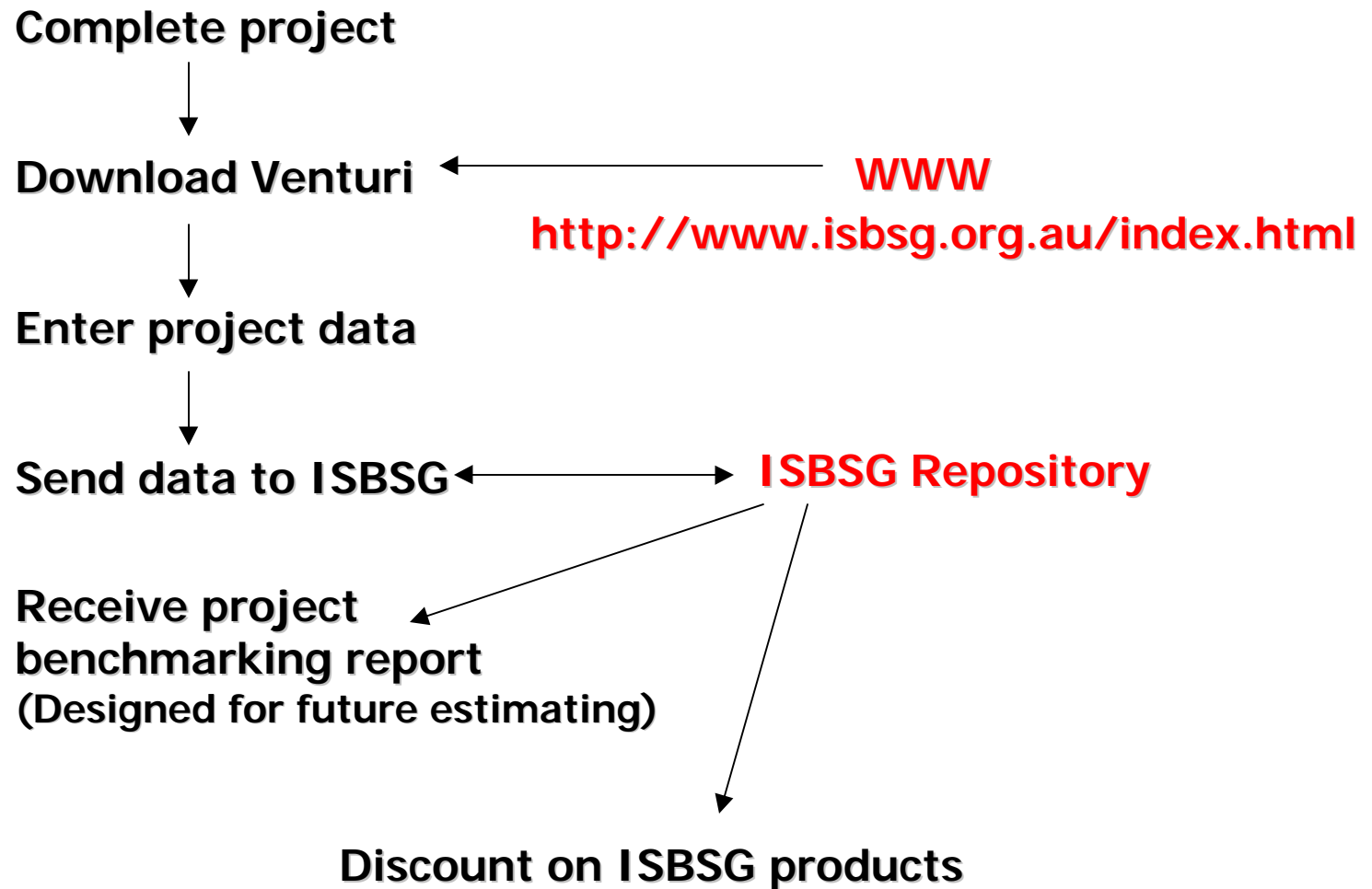
# ***International recognition***

---

**In the Spring of 1998, FFP was accepted as a valid functional size measure by ISBSG\*, an international benchmarking organization.**

**ISBSG: International Software Benchmarking Standards Group**

# *Benchmarking your results*



# ***The future of Full Function Points***

---

- ◉ Looking for more industrial partners for field testing,
- ◉ Looking for more industrial partners for data collection,
- ◉ International Measurement Standards Committee,
- ◉ ISO 14143 certification to start in 1999.

# *Available resources*

---

- ◎ **Complete documentation on the Web**
  - ✓ Concepts and definitions,
  - ✓ Measurement Manual,
  - ✓ Publications,
  - ✓ <http://www.lrgl.uqam.ca/ffp.html>
  
- ◎ **Support available**
  - ✓ Case Study
  - ✓ On site custom training
  - ✓ Consulting support

## *Final remarks...*

---

- ◉ FFP addresses a problem identified since 1986,
- ◉ FFP was designed for ISO compliance,
- ◉ FFP has been designed FOR the industry, WITH the industry,
- ◉ FFP is an open and transparent initiative, fully documented and easily available,
- ◉ FFP is already helping organizations manage their non-MIS software.

# Sources of Funding

---

Developing FOR the industry, WITH the industry, FFP industrial partners...



Bell Canada, CANADA



Northern Telecom, Canada & USA



JECS Systems Research, JAPAN



Hydro-Québec, CANADA

# *Acknowledgements...*

---

- ① **The Software Engineering Management Research Laboratory of the Université du Québec à Montréal is supported through a partnership with Bell Canada.**
- ① **Additional funding is provided by the National Science and Engineering Research Council of Canada.**