Software Engineering

Function points

Sw metrics

- Direct metrics
  - On the code
    - LOC (Line Of Code)
    - McCabe index
    - ...
  - On the requirements
    - Transactional (FP)
    - OO (research...)
    - ...
- Indirect
  - Service levels
  - Users' opinions
  - ....

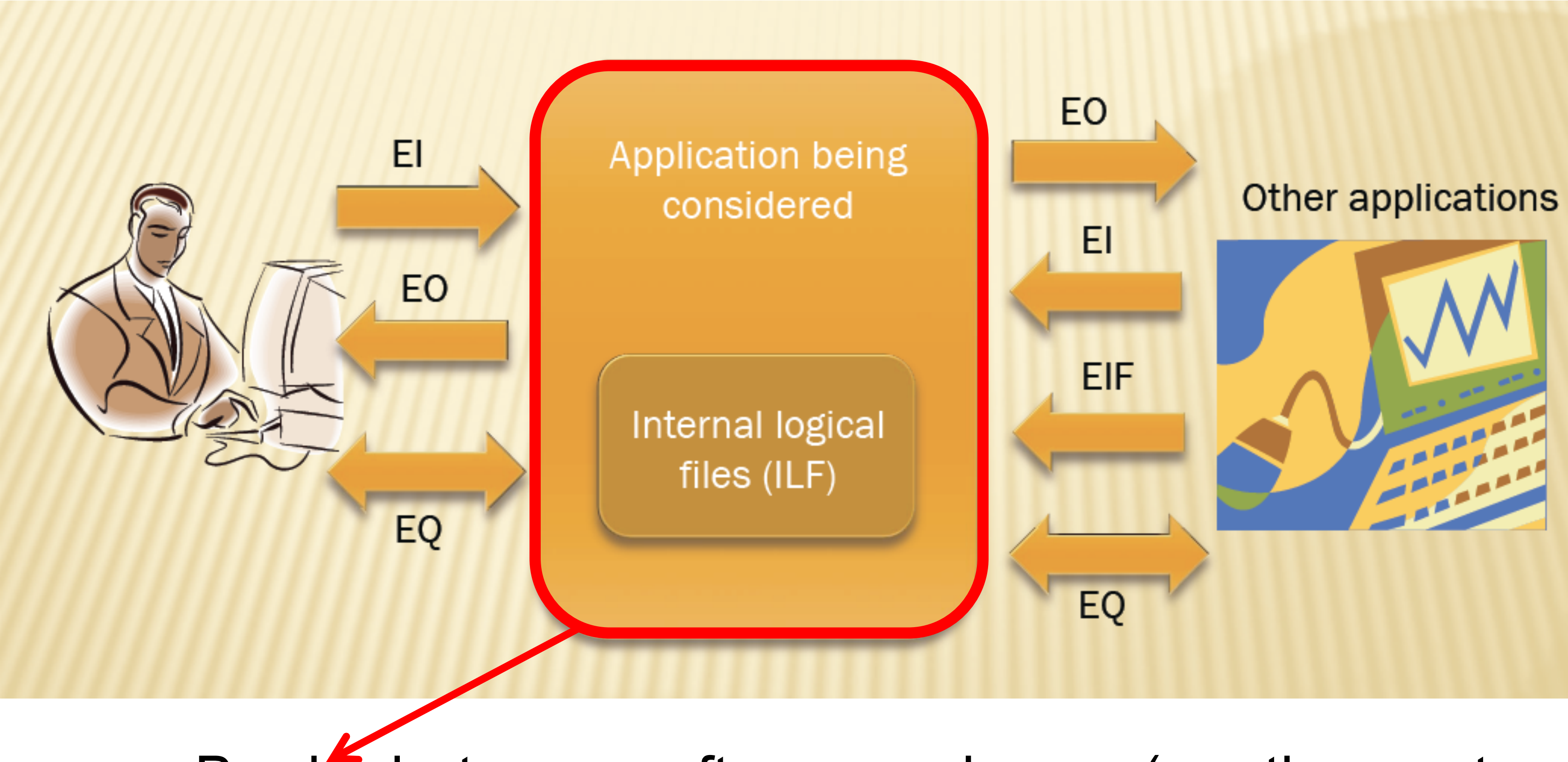
Dimension metrics

- LOC
  - Internal errors / KLOC
  - External errors / KLOC / year
  - Documentation pages / KLOC
  - LOC /person/month
  - Errors /person/month
  - cost per LOC
  - cost per documentation page
- LOC is (seems) easy to compute but is strictly associated with the actual programming language and programming style

Dimension metrics

- Program functionalities
- First proposal 79 (Albrecht) : Function Point (FP)
  - empirical formula based on basic (weighted) functionalities
- Several extensions to the original proposal

Functionalities



Border between software and user (or other systems)
What is going to be measured?
Inside = being measured
Outside = not being measured

FP : 1984 proposal

Functionality		Count	Weight			
EI		X	3-4-6		=	
EO		X	4-5-7		=	
EQ		X	3-4-6		=	
ILF		X	7-10-15		=	
EIF		X	5-7-10		=	
				Total		

$$FP = \quad \text{Total} \quad \times \left(0.65 + 0.01 \times \sum_{i=1}^{14} F_i\right)$$

The 5 functionalities

- **Internal logical file (ILF)** An ILF is a user-identifiable group of logically related data or control information maintained within the boundary of the application. The primary intent of an ILF is to hold data maintained through one or more elementary processes of the application being counted.
- **External interface file (EIF)** An external interface file (EIF) is a user identifiable group of logically related data or control information referenced by the application, but maintained within the boundary of another application. The primary intent of an EIF is to hold data referenced through one or more elementary processes within the boundary of the application counted. This means an EIF counted for an application must be in an ILF in another application.
- **External input (EI)** An external input (EI) is an elementary process that processes data or control information that comes from outside the application boundary. The primary intent of an EI is to maintain one or more ILFs and/or to alter the behavior of the system.
- **External output (EO)** An external output (EO) is an elementary process that sends data or control information outside the application boundary. The primary intent of an external output is to present information to a user through processing logic other than, or in addition to, the retrieval of data or control information . The processing logic **must** contain at least one mathematical formula or calculation, create derived data maintain one or more ILFs or alter the behavior of the system.

External inquiry (EQ) An external inquiry (EQ) is an elementary process that sends data or control information outside the application boundary. The primary intent of an external inquiry is to present information to a user through the retrieval of data or control information from an ILF of EIF. The processing logic contains no mathematical formulas or calculations, and **does not** create derived data. No ILF is maintained during the processing, nor is the behavior of the system altered

Complexity / Weights

- Each functionality has three weights (**L**ow, **M**edium, **H**igh)
- Weights are different for the 5 functionalities

Adjusted FP

- The produced value (non-weighted FP) can be changed (+/-35%) through a corrective formula that “captures" general characteristics of the system through 14 indicators such as:
  - does the system require procedures for backup/recovery?
  - does it require a data transfer ? ....

- Each indicator takes a value between 0 e 5
  - 0) Not relevant... 5) Essential

$$\text{AFP} = \text{Total} \times \left(0.65 + 0.01 \times \sum_{i=1}^{14} F_i \right)$$

- We will not deal with this issue anymore (reasons will be discussed in the next classes)

FP limits

- Pros
  - Widely used and accepted (standards, active organizations)
  - Certified personnel available
  - Objective calculation
  - UFP independent of technology
  - Can be used early in development process
  - Equally accurate as SLOC
- Cons
  - Semantic difficulty - “legacy” terminology difficult for teaching, and FP are in themselves hard to grasp and compare
  - Incompleteness – internal functionality? Stored data size vs. complex processing?
  - Lack of automatic count
  - Different versions

LOC vs FP

Programming Language	LOC/FP
Assembler	320
C	128
Cobol	105
Fortran	105
Pascal	90
Ada	70
OO (C++ / JAVA)	30
4GL	20
Generatore di codice	15
Foglio elettronico	6
Linguaggio grafico/visuale	4

Data functionalities

- **Internal logical file (ILF)** An ILF is a user-identifiable group of logically related data or control information maintained within the boundary of the application. The primary intent of an ILF is to hold data maintained through one or more elementary processes of the application being counted.
- **External interface file (EIF)** An external interface file (EIF) is a user identifiable group of logically related data or control information referenced by the application, but maintained within the boundary of another application. The primary intent of an EIF is to hold data referenced through one or more elementary processes within the boundary of the application counted. This means an EIF counted for an application must be in an ILF in another application.

Component of data functionalities

- **DET D**ata - **E**lement **T**ype – User identifiable single field within a ILF / EIF

- **RET** - **R**ecord **E**lement **T**ype - User identifiable group of fields within a ILF / EIF

- The complexity of a ILF or EIF is associated with the number of RET / DET

ILF/EIF complexity

Ret/Det	1-19 Det	20-50 Det	51+ Det
1 Ret	Low (7/5)	Low (7/5)	Medium (10/7)
2-5 Ret	Low (7/5)	Medium (10/7)	High (15/10)
6+ Ret	Medium (10/7)	High (15/10)	High (15/10)

ILF

COMPLESSITÀ	PUNTI
LOW	7
MEDIUM	10
HIGH	15

EIF

COMPLESSITÀ	PUNTI
LOW	5
MEDIUM	7
HIGH	10

Transactions

- **External input (EI)** An external input (EI) is an elementary process that processes data or control information that comes from outside the application boundary. The primary intent of an EI is to maintain one or more ILFs and/or to alter the behavior of the system.
- **External output (EO)** An external output (EO) is an elementary process that sends data or control information outside the application boundary. The primary intent of an external output is to present information to a user through processing logic other than, or in addition to, the retrieval of data or control information . The processing logic must contain at least one mathematical formula or calculation, create derived data maintain one or more ILFs or alter the behavior of the system.
- **External inquiry (EQ)** An external inquiry (EQ) is an elementary process that sends data or control information outside the application boundary. The primary intent of an external inquiry is to present information to a user through the retrieval of data or control information from an ILF of EIF. The processing logic contains no mathematical formulas or calculations, and creates no derived data. No ILF is maintained during the processing, nor is the behavior of the system altered

Transaction identification

Actions	EI	EO	EQ
1) Validate	can	can	can
2) Math calculations	can	must*	cannot
3) Select data using specific criteria	can	can	can
4) Evaluate Boolean conditions	can	can	can
5) Update one or more ILF	must* / p.goal	must*	cannot
6) Read one or more ILF /EIF	can	can	must
7) Get some control data	can	can	must
8) Compute new data	can	must*	cannot
9) Change the system behavior	must*/p.goal	must*	cannot
10) Find data and present them outside the application boundary	can	must/p.goal	must/p.goal
11) Acquire (control) data produced outside the application boundary	must	can	can
12) Manipulate data (e.g., sort)	can	can	can

must* means that the transaction must execute at least one of them

Transaction components

FTR changed / read ILF or read EIF (File Type Referenced)

DET User identifiable single field within a ILF / EIF

Transaction complexity

EI

FTR / DET	1-4 DET	5-15 DET	16+ DET
0 – 1 FTR	Low (3)	Low (3)	Medium (4)
2 FTR	Low (3)	Medium (4)	High (6)
3+ FTR	Medium (4)	High (6)	High (6)

EO/EQ

FTR / DET	1-5 DET	6/19 DET	20+ DET
0 – 1 FTR	Low (4/3)	Low (4/3)	Medium (5/4)
2-3 FTR	Low (4/3)	Medium (5/4)	High (7/6)
4+ FTR	Medium (5/4)	High (7/6)	High (7/6)

Example

- Invoice database

Requirements

- The sales division wants to handle the following customers' information
  - Name
  - CF
  - IVA (VAT)
  - Legal address
  - Corresponding address
  - Phone
  - Invoices

Requirements 2

- for each **invoice** :
  - Invoice number
  - Issue date
  - Payment date
  - and,

Specifiche dati 3

- for each **item** :
  - Description
  - Number of items
  - IVA percentage
  - Unit cost
- N.B. Description, IVA percentage, and unit cost come from an external application, Warehouse

Functional requirements 1

Estimate the FP of an application that allows for

- inserting
- deleting
- editing
- viewing

customers and invoices

The customer search is carried out using to the CF, while that of invoices by invoice number

Functional requirements 2
Printing an invoice from the screen visualizing it

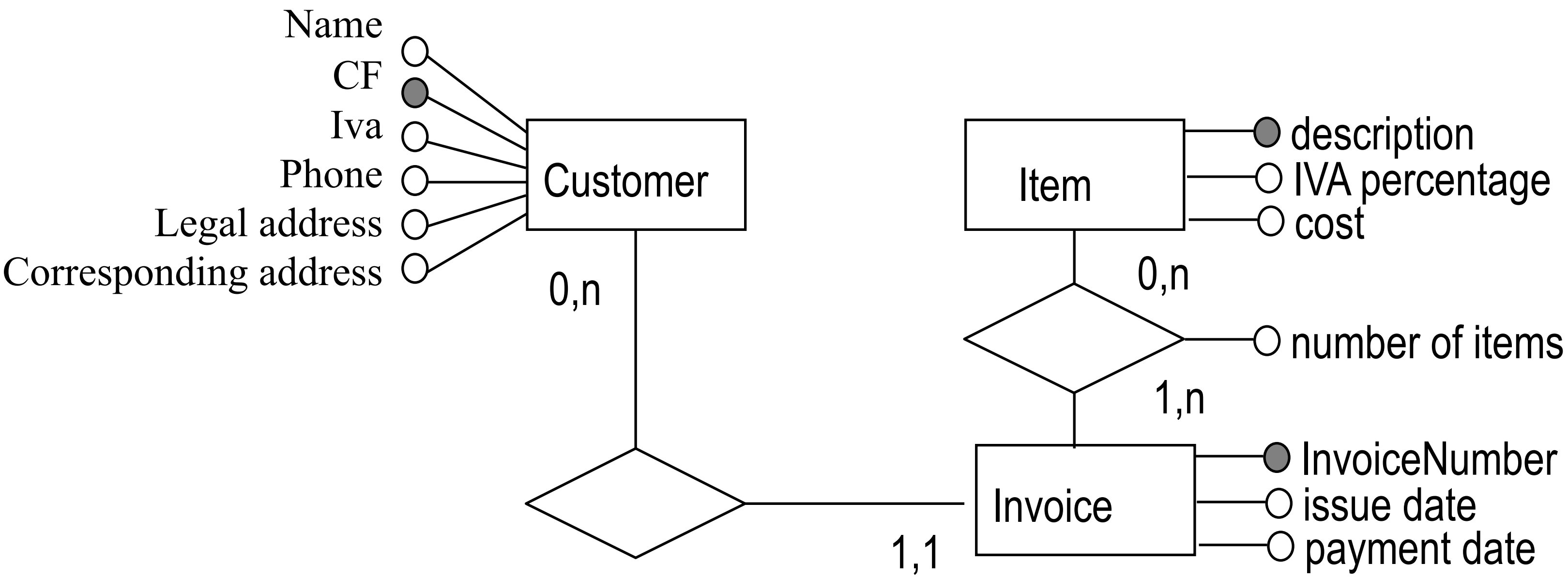
- Customer name
- IVA
- CF
- Legal address
- Phone
- Invoice number
- Issue date

- Item description
- Number of items
- IVA percentage
- Unit cost
- Total
- IVA Total

Functional requirements 2 Printing a customer from the screen visualizing him

- Customer name
- CF
- IVA
- Legal address
- Corr. address
- Phone

ER Schema
(not mandatory but useful)



ILF and EIF

- ILF Customer
- ILF Invoice
- EIF Item

ILF e EIF complexity

ILF	Customer	RET	1	DET	6
ILF	Invoice	RET	1	DET	4 (includes #items)
EIF	Item	RET	1	DET	3

ILF/EIF complexity table

Ret/Det	**1-19 Det**	**20-50 Det**	**51+ Det**
1 Ret	Low (7/5)	Low (7/5)	Medium (10/7)
2-5 Ret	Low (7/5)	Medium (10/7)	High (15/10)
6+ Ret	Medium (10/7)	High (15/10)	High (15/10)

2 ILF LOW -> 14 FP
1 EIF LOW -> 5 FP

Transaction: EI

- 6 EI:
  - insert
  - delete
  - update
- customer and invoices
- primary goal: update an ILF

EI complexity

Ins	Customer	FTR	1 (Customer)	DET	6
Del	Customer	FTR	1 (Customer)	DET	1
Upd	Customer	FTR	1 (Customer)	DET	5 (6-1:CF is the key)
Ins	Invoice	FTR	3 (Invoice, Customer, Item)	DET	8 (Invoice, Item, CF)
Del	Invoice	FTR	1 (Invoice)	DET	1 (#Invoice Number)
Upd	Invoice	FTR	3 (Invoice, Customer, Item)	DET	7 (8-1: InvoiceNumber is the key)

EI Transaction complexity

- Ins. and update of invoice ---> HIGH
- Others LOW

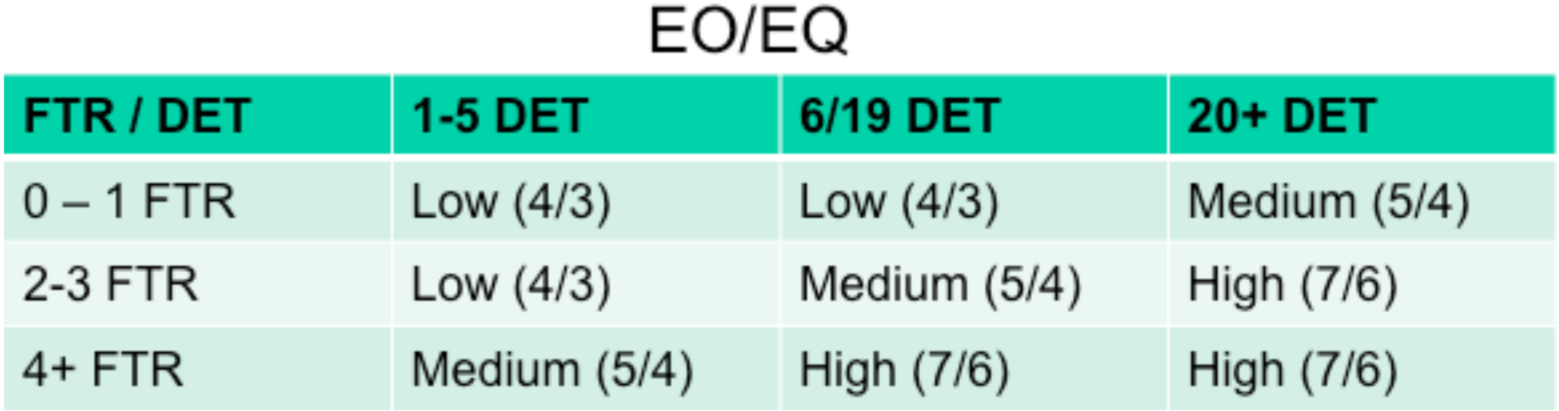
FTR / DET	1-4 DET	5-15 DET	16+ DET
0 – 1 FTR	Low (3)	Low (3)	Medium (4)
2 FTR	Low (3)	Medium (4)	High (6)
3+ FTR	Medium (4)	High (6)	High (6)

2 EI HIGH	->12 FP
4 EI LOW	->12 FP

Ins	Customer	FTR	1 (Customer)	DET 6
Del	Customer	FTR	1 (Customer)	DET 1
Upd	Customer	FTR	1 (Customer)	DET 5 (6-1:CF is the key)
Ins	Invoice	FTR	3 (Invoice, Customer, Item)	DET 8 (Invoice, Item, CF)
Del	Invoice	FTR	1 (Invoice)	DET1 (#Invoice Number)
Upd	Invoice	FTR	3 (Invoice, Customer, Item)	DET7 (8-1: #InvoiceNumber is the key)

Transaction: EO

- 1 EO printing an invoice
- Input : nothing, it is triggered from the invoice view
- Output: FTR 3(Invoice, Customer, Item)
- DET 10(Name,CF,IVA, Legal Address, Issue date, InvoiceNumber, Description, IVA percentage, cost,#item)



EO/EQ

FTR / DET	1-5 DET	6/19 DET	20+ DET
0 – 1 FTR	Low (4/3)	Low (4/3)	Medium (5/4)
2-3 FTR	Low (4/3)	Medium (5/4)	High (7/6)
4+ FTR	Medium (5/4)	High (7/6)	High (7/6)

1 EO MEDIUM -> 5 FP

EO because it computes derived data (e.g., tot invoice)

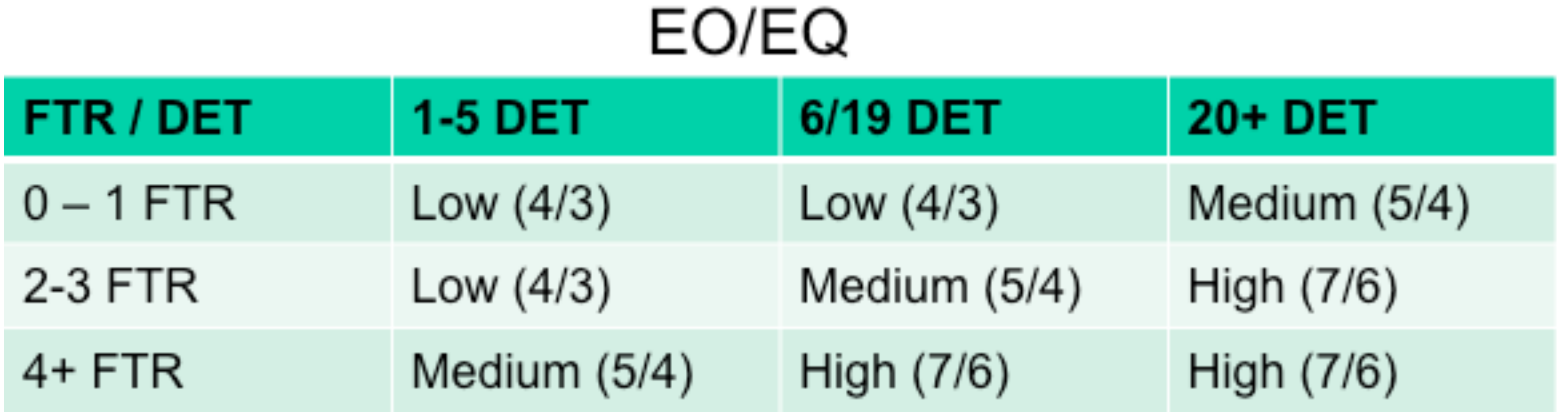
N.B. Derived data are **NOT** DET

Transaction EQ

1 EQ Printing a customer

- input :nothing, it is triggered from the customer view
- output FTR 1 (Customer) DET 6
  - 1 FTR 6 DET
- EQ because it only finds data and presents it outside the application boundary

EQ complexity



EO/EQ

FTR / DET	1-5 DET	6/19 DET	20+ DET
0 – 1 FTR	Low (4/3)	Low (4/3)	Medium (5/4)
2-3 FTR	Low (4/3)	Medium (5/4)	High (7/6)
4+ FTR	Medium (5/4)	High (7/6)	High (7/6)

1 FTR 6 DET

1 EQ LOW -> 3 FP

UFP - TOTAL

• ILF ELF	->	19
• EI	->	24
• EO	->	5
• EQ	->	3
• TOTAL		51 UFP

51 FP

Programming Language	LOC/FP	
Assembler	320	16320
C	128	
Cobol	105	
Fortran	105	
Pascal	90	
Ada	70	
OO (C++ / JAVA)	30	1530
4GL	20	
Generatore di codice	15	
Foglio elettronico	6	
Linguaggio grafico/visuale	4	