SWEN90016
Software Processes & Project Management

Function points (in depth example)

Marion Zalk
Department of Computing and Information Systems
The University of Melbourne
mzalk@unimelb.edu.au

2021– Semester 1

Lecture 7

Function Points (FP)

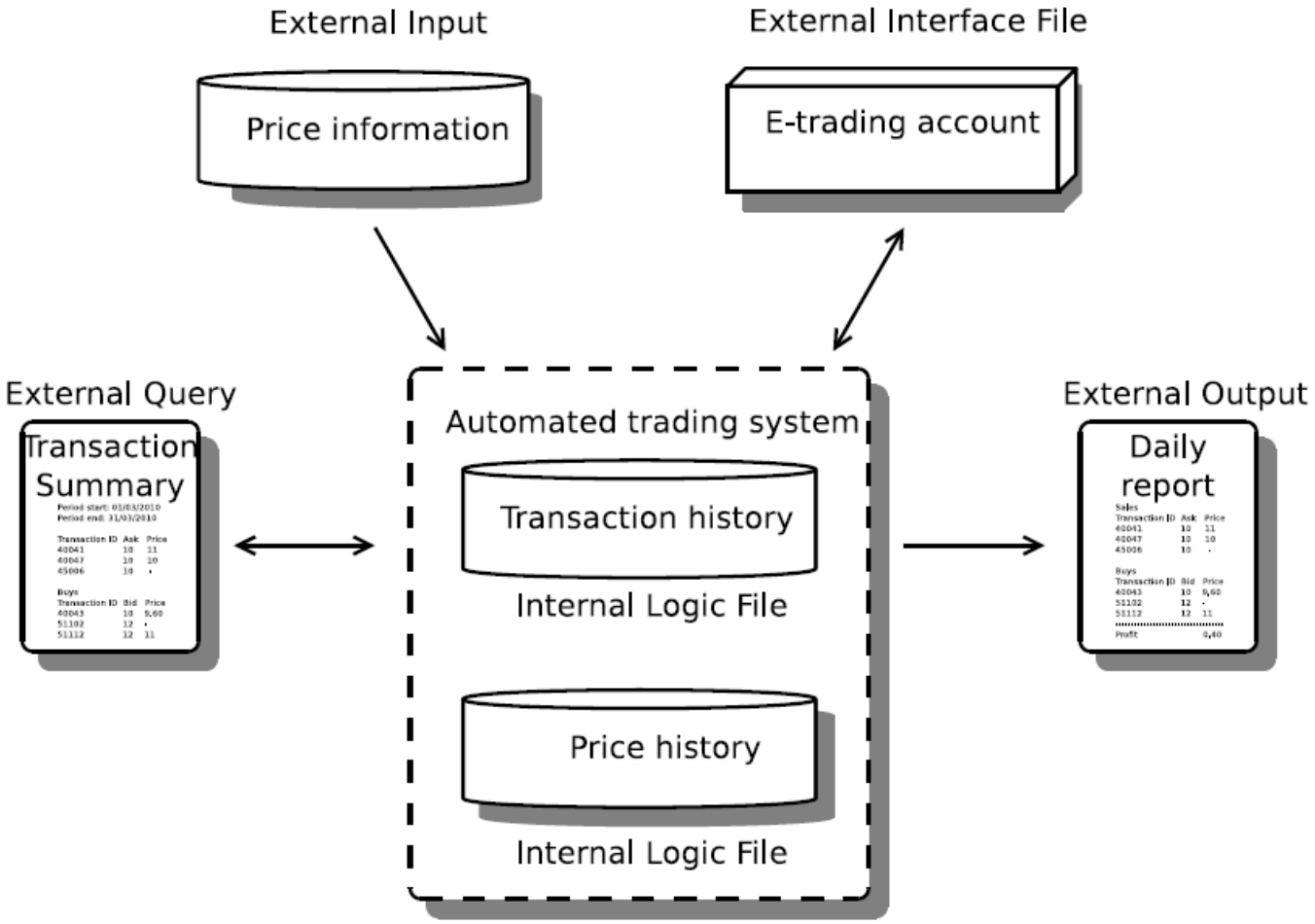
- Is used to express the *amount of functionality* in a software system, as seen by the user
- A *higher number of function points* indicates *more functionality*
  - Empirical evidence demonstrates that there is a *positive correlation between function points and the complexity* of the system
- Typically used to:
  - Estimate the cost and effort required to design, code and test a software system
  - Predict the number of errors
  - Predict the number of components
  - Measure productivity
- Function points are computed from the *Software Requirements Specification (SRS)*

Software Requirements

- **Software Requirements Specification (SRS)**
  - A document that specifies what is expected of a software system; referred to as the requirements of the system
  - It contains:
    - Functional Requirements:
      - Specify the functions that are required in the system
    - Non-functional Requirements:
      - Specify requirements that are not directly functions, such as performance, reliability, scalability etc. (quality requirements)

Example: Functional Requirements

Automated Trading System



Example: Automated Trading System

Automated Trading System: Functional Requirements

R.1	Read the previous day's trading information (high price, low price, opening price, closing price) from a third-party-server.
R.2	Save a complete "price history" in a database.
R.3	Based on the trends in prices for commodities, decide which commodities to bid for, and which to try to sell.
R.4	Send information to an external e-trading account which places the bid/ask for each commodity.
R.5	When the bid/ask is either accepted or expires, record the result in a "transaction history" database.
R.6	At the end of the day, produce a report that summaries the transactions of the day along with the following information: profit/loss for the day; number of trades of each commodity; average market price of all commodities; account summary.
R.7	At anytime the user can request a transaction history for a period which gives: transaction IDs; commodity type; bid/ask; price.

FP Computation Steps

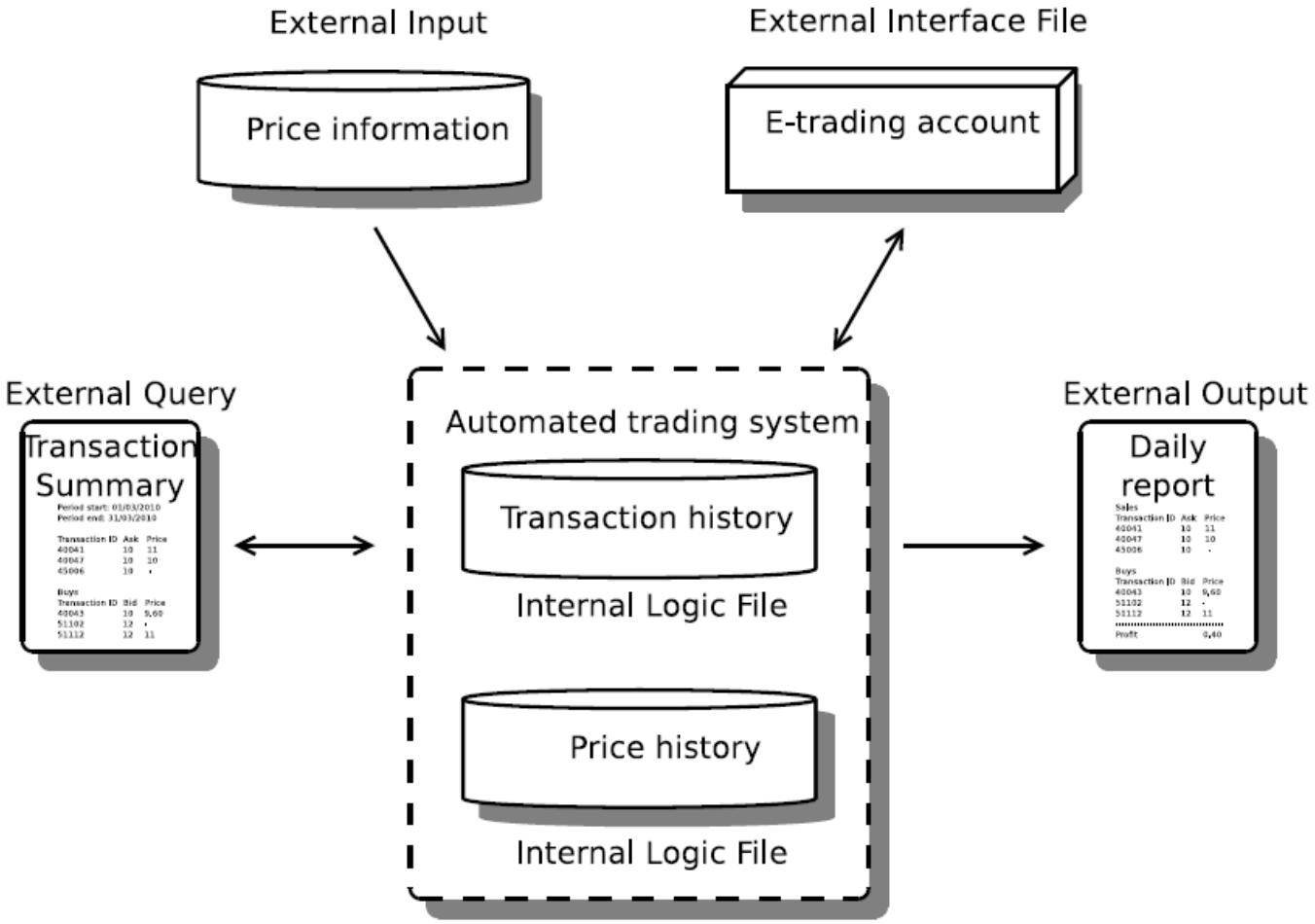
1. Categorize requirements
2. Estimate a *complexity value* for each category or function
3. Compute *count total* from Complexity
4. Estimate *value adjustment factors*
5. Compute *total function point count*

FP Computation Steps

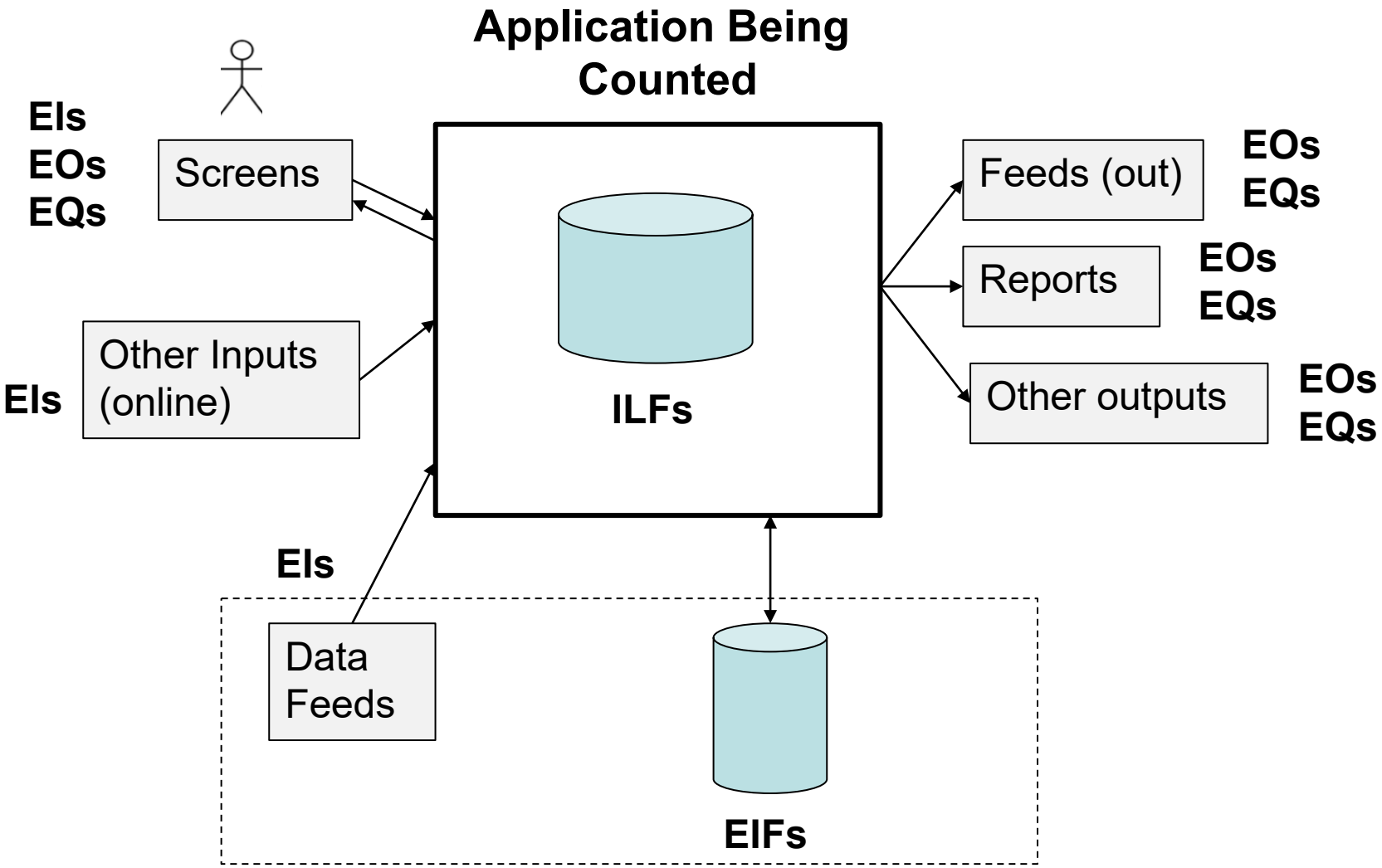
1. Categorize requirements
2. Estimate a *complexity value* for each category or function
3. Compute *count total* from Complexity
4. Estimate *value adjustment factors*
5. Compute *total function point count*

Example: Functional Requirements

Automated Trading System

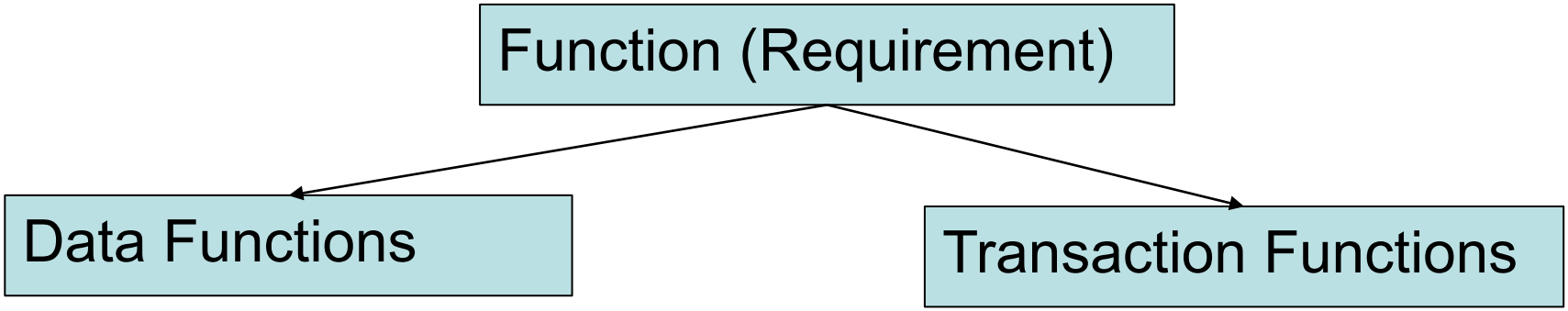


FP Computation – Step 1



FP Computation – Step 1

1. Categorize Requirements



Data Functions:

- Concerned with maintenance of the data for the application – Internal Logical Files (ILF), External Interface Files (EIF)

Transaction Functions:

- Concerned with information being passes to and from the system - External Inputs (EI), External Output (EO), External Inquiries/Queries (EQ)

FP Computation – Step 1

1. Categorize Requirements (five categories)

Data Functions

- **Internal Logical File (ILF)**
  - A logical grouping of data that the system maintains over a period of time, and is modified using external inputs

    examples - tables in a relational database, files containing user setting

- **External Interface File (EIF)**
  - A logical grouping of data that is maintained external to the system, but which may be used by the system.

    examples - are the same as ILFs, except that the data is maintained outside the system, such as data hosted on a third-party servers, or data structures holding information about system state

FP Computation - Step 1

Transaction Functions

- **External Input (EI)**
  - An input to the system from a user or another application, which is used to control the flow of the system, or provide data. External inputs generally modify internal logic files

    examples - data fields populated by users, inputs files (e.g. program source code to a compiler), and file feeds from an external application.

- **External outputs (EO)**
  - An output to the user that provides information about the state of the system

    examples - screens, error messages, and reports that are shown to the user. Individual data fields in these are grouped as one external output

FP Computation – Step 1

– **External Inquiries/Queries (EQ)**

- the input is not used to update an internal logic file, but is used to query the internal logic file and provide an output; the output is retrieved directly, with no derived data included

  examples - reading a user setting, or reading a record from a database table

Example: Automated Trading System

Automated Trading System: Function Categories

R.1	Read the previous day's trading information (high price, low price, opening price, closing price) from a third-party-server.	EI
R.2	Save a complete "price history" in a database.	ILF
R.3	Based on the trends in prices for commodities, decide which commodities to bid for, and which to try to sell.	
R.4	Send information to an external e-trading account which places the bid/ask for each commodity.	EIF
R.5	When the bid/ask is either accepted or expires, record the result in a "transaction history" database.	ILF
R.6	At the end of the day, produce a report that summaries the transactions of the day along with the following information: profit/loss for the day; number of trades of each commodity; average market price of all commodities; account summary.	EO
R.7	At anytime the user can request a transaction history for a period which gives: transaction IDs; commodity type; bid/ask; price.	EQ

FP Computation Steps

1. Categorize requirements
2. Estimate a *complexity value* for each category or function
3. Compute *count total* from Complexity
4. Estimate *value adjustment factors*
5. Compute *total function point count*

FP Computation – Step 2

2. Estimate a *complexity value* for each category or function

- Complexity is ranked either ***simple, average or complex***
- Normally ***assigned for a category*** rather than for each requirement – rather crude
- A technique commonly used is based on Data Element Types (DETs), Record Element Types (RETs), and File Type References (FTRs):

Data Element Types (DETs)	A unique, user-recognizable, non-repeated data field in a system
Record Element Types (RETs)	A user-recognizable subgroup of data elements in an ILF or EIF
File Type References (FTRs)	A file (ILF, EIF) referenced by a transaction

FP Computation – Step 2

Relationship between DETs, RETs, FTRs, and the function categories

Function	DETs	RETs	FTRs
Internal Logical Files (ILFs)	x	x	
External Interface Files (EIFs)	x	x	
External Inputs (EIs)	x		x
External Outputs (EOs)	x		x
External Inquiries (EQs)	x		x

Complexity table for Data Functions

	DETs		
RETs	1-19	20-50	51+
1	Simple	Simple	Average
2-5	Simple	Average	Complex
6+	Average	Complex	Complex

Complexity table for Transaction Functions

	DETs		
FTRs	1-5	6-19	20+
1	Simple	Simple	Average
2-3	Simple	Average	Complex
4+	Average	Complex	Complex

Example: Automated Trading System

Automated Trading System: Function Categories

R.1	Read the previous day's trading information (high price, low price, opening price, closing price) from a third-party-server.	EI
R.2	Save a complete "price history" in a database.	ILF
R.3	Based on the trends in prices for commodities, decide which commodities to bid for, and which to try to sell.	
R.4	Send information to an external e-trading account which places the bid/ask for each commodity.	EIF
R.5	When the bid/ask is either accepted or expires, record the result in a "transaction history" database.	ILF
R.6	At the end of the day, produce a report that summaries the transactions of the day along with the following information: profit/loss for the day; number of trades of each commodity; average market price of all commodities; account summary.	EO
R.7	At anytime the user can request a transaction history for a period which gives: transaction IDs; commodity type; bid/ask; price.	EQ

Example: Automated Trading System

Automated Trading System: Function Categories

			DETs	RETs	FTRs	Complexity
R.1	**EI**	high price, low price, opening price, closing price, date	5		1	Simple
R.2	**ILF**	commodity name, high price, low price, opening price, closing price, date	6	1		Simple
R.3						
R.4	**EIF**	Commodity, bid/asking price, buy/sell	3	1		Simple
R.5	**ILF**	…	4	1		Simple
R.6	**EO**	…	5		3	Simple
R.7	**EQ**	…	4		2	Simple

FP Computation Steps

1. Categorize requirements
2. Estimate a *complexity value* for each category or function
3. Compute *count total* from Complexity
4. Estimate *value adjustment factors*
5. Compute *total function point count*

FP Computation – Step 3

3. Compute *count total* from Complexity

- Using count and complexity estimates compute total count

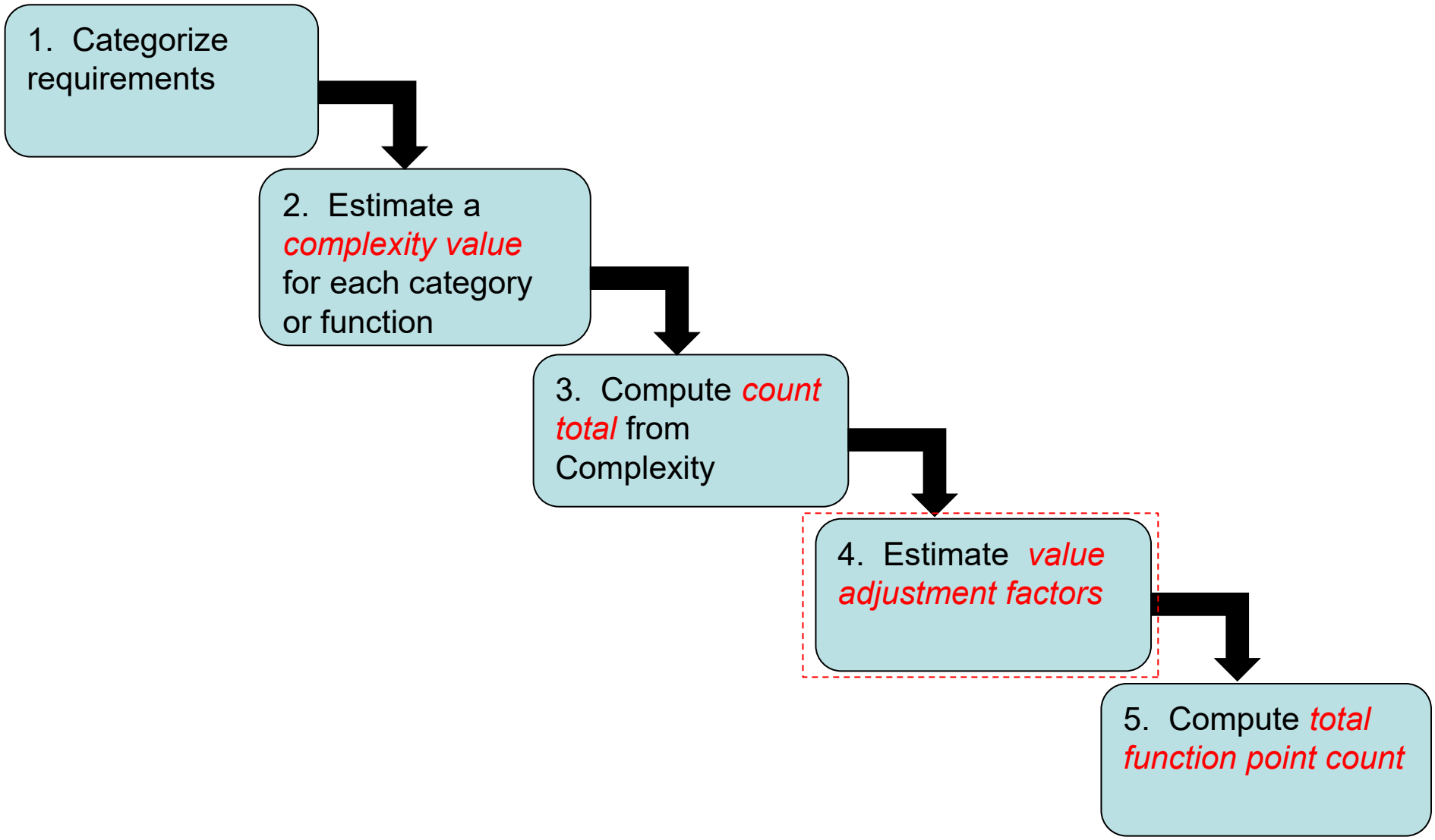
Information Domain Value	Count		Weighting Factor Simple	Average	Complex		
Internal Logical Files (ILFs)		×	7	10	15	=	
External Interface Files (EIFs)		×	5	7	10	=	
External Inputs (EIs)		×	3	4	6	=	
External Outputs (EOs)		×	4	5	7	=	
External Inquiries (EQs)		×	3	4	6	=	
Count total							

Example: Automated Trading System

Automated Trading System: Computing Count Total

Information Domain Value	Count		Weighting Factor Simple	Average	Complex		
Internal Logical Files (ILFs)	2	×	7	~~10~~	~~15~~	=	14
External Interface Files (EIFs)	1	×	5	~~7~~	~~10~~	=	5
External Inputs (EIs)	1	×	3	4	~~6~~	=	3
External Outputs (EOs)	1	×	4	~~5~~	~~7~~	=	4
External Inquiries (EQs)	1	×	3	4	~~6~~	=	3
Count total							29

FP Computation Steps



FP Computation – Step 4

4. Compute value adjustment factors

- is computed based on 14 characteristics
- each of the characteristics is ranked on a scale of 0-5; 0 not important and 5 critical

- Data Communications
- Distributed Data Processing
- Performance
- Heavily Used Configuration
- Transaction Rate
- Online Data Entry
- End-User Efficiency

- Online Update
- Complex Processing
- Reusability
- Installation Ease
- Operational Ease
- Multiple Sites
- Facilitate Change

Example: Automated Trading System

Automated Trading System: Value Adjustment Factors

- Data Communications - 2
- Distributed Data Processing - 4
- Performance - 5
- Heavily Used Configuration - 2
- Transaction Rate - 4
- Online Data Entry - 2
- End-User Efficiency - 5

- Online Update - 2
- Complex Processing - 2
- Reusability - 2
- Installation Ease - 2
- Operational Ease - 2
- Multiple Sites - 2
- Facilitate Change - 2

Assume the rating for all other features is 2

Value Adjustment = 2*5 + 4*2 + 10*2 = 38

FP Computation Steps

1. Categorize requirements
2. Estimate a *complexity value* for each category or function
3. Compute *count total* from Complexity
4. Estimate *value adjustment factors*
5. Compute *total function point count*

FP Computation – Step 5

5. Compute *total function point*

- Count total and value adjustment factors are then plugged-in to the following formula to estimate the total point count

$$FP = count\ total \times (0.65 + 0.01 \times \sum_{i=1}^{14} F_i)$$

F_i- VAF corresponding to the *i-th* VAF question

Example: Automated Trading System

Automated Trading System: Compute total function point count

$$FP = count\ total \times (0.65 + 0.01 \times \sum_{i=1}^{14} F_i)$$

FP = 29 x(0.65 + 0.01*38)
= 29 x1.03
= 29.87

More information and examples of FPs can be found at https://alvinalexander.com/FunctionPoints/FunctionPoints.shtml

Function Points

- **Advantages of Function Points**
  - Measures the size of the solution instead of the size of the problem
  - Requirements are the only thing needed for function points count
  - Can be estimated early in analysis and design
  - Is independent of technology
  - Is independent of programming languages

Function Points

- **Disadvantages of Function Points**
  - A well defined requirements specification is necessary
  - Gaining proficiency is not easy, the learning curve is quite long
  - Could be quite time-consuming thus could be costly

References

1. B. Boehm, C. Abts, W. Brown, S. Chulani, B. Clark, E. Horowitz, R. Madachy, D. Reifer, and B. Steece. Software Cost Estimation with Cocomo II. Prentice Hall, 2000.
2. R. S. Pressman. Software Engineering: A Practitioner's Approach. McGraw Hill, seventh edition, 2009.
3. I Somerville. Software Engineering, Addison-Wesley Publishing, ninth edition, 2010.
4. K. Molkken and M. Jrgensen, A Review of Surveys on Software Effort Estimation, Proceedings of the 2003 International Symposium on Empirical Software Engineering

cont…

References

Molkken, Kjetil and Jrgensen, Magne

5. Danh Nguyen-Cong and De Tran-Cao, A review of effort estimation studies in agile, iterative and incremental software development, The 2013 RIVF International Conference on Computing Communication Technologies - Research, Innovation, and Vision for Future (RIVF)

6. M. Usman, E, Mendes and F. Weidt and R. Britto, Effort Estimation in Agile Software Development: A Systematic Literature Review, Proceedings of the 10th International Conference on Predictive Models in Software Engineering, 2014

7. PMI's PULSE of the PROFESSION -https://www.pmi.org/learning/thought-leadership/pulse/pulse-of-the-profession-2017

References

1. B. Boehm, C. Abts, W. Brown, S. Chulani, B. Clark, E. Horowitz, R. Madachy, D. Reifer, and B. Steece. Software Cost Estimation with Cocomo II. Prentice Hall, 2000.
2. R. S. Pressman. Software Engineering: A Practitioner's Approach. McGraw Hill, seventh edition, 2009.
3. I Somerville. Software Engineering, Addison-Wesley Publishing, ninth edition, 2010.
4. A Review of Surveys on Software Effort Estimation
5. A review of effort estimation studies in agile, iterative and incremental software development, The 2013 RIVF International Conference on Computing Communication Technologies - Research, Innovation, and Vision for Future (RIVF)