Functional Complexity Measurement

Presented at IWSM'2001

By

De Tran-Cao, PhD. candidate, UQAM

Alain Abran, professor, UQAM

Ghislain Lévesque, professor, UQAM

August, 29

Agenda

- Introduction
- What is the complexity?
- Software model and software complexity model
- Measurement methods
- Conclusion

Introduction

- To estimate software development effort:
  - How big the task is?
  - How difficult the task is?
- Software size: LOC, FPA
- FPA is more useful than LOC in predicting early development effort.
- FPA have also some weakness
  - not taking into account the complexity of algorithms and some characteristics of real-time software.
  - subjectivity in determining scale and the weights / degrees of influence.0

FPA review

- Identify software functionalities.
- Classify them in five functional types and *three level of complexity*.
- Weight the functionalities on basic of their complexity ➔ UFP
- Determine the *degree of influence of 14 GSCs* on a scale of six levels: 0..5
- Sum the degrees of influence ➔ VAF
- FP = UFP x (0.65 + 0.01 x VAF)

Lesson learned

- FPA measures software size via the complexity of basic functional types and the complexity of system (14GSCs). That implicitly propose two types of complexity: component complexity and system complexity.

- Card (90) argues: component complexity and system complexity.
- Basili (96): a system is characterized by its elements and the relationships between them.

➔Identify elements of software and relationships between them.

Research direction

- The COSMIC software model identifies the software component as a functional process.

➔software complexity: complexity in each functional process and the complexity in the relationships between processes

What is complexity?

- IEEE (90): the degree to which a system or component has a design or implementation that is difficult to understand and verify
- Basili (80): complexity as a measure of the resources expended by a system while interacting with a piece of software to perform a given task. If the interacting system is a programmer, then complexity is defined by the difficulty of performing tasks such as coding, debugging, testing or modifying the software
- Zuse (91) says that the term complexity measure is a misnomer: The true meaning of the term software complexity is the difficulty to maintain, change and understand software.

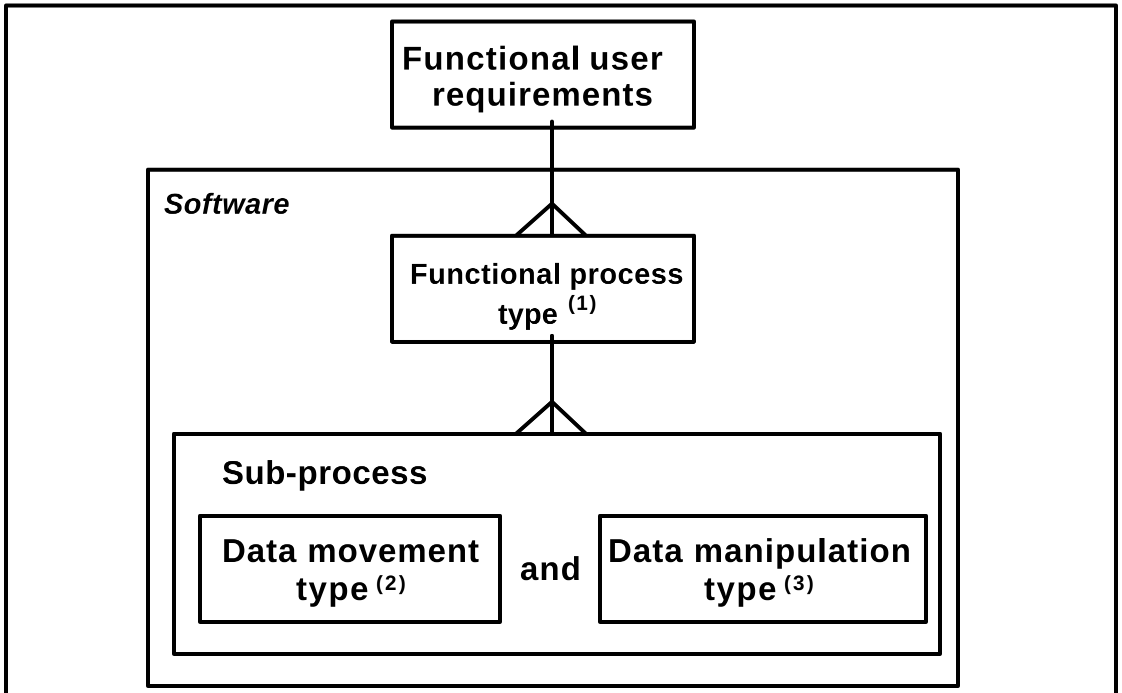
What is complexity?

- These definitions associate software complexity with the difficulty of performing a task on the software. An implicit assumption is that software complexity correlates well with the work effort (man-hours) required to develop or maintain the software.

  ➔ higher complexity, more effort needed.

- The term *'functional complexity'* is interpreted as a candidate explanatory variable for investigating the work effort required to develop the software function, including decomposing and allocating the functional processes and designing each functional process to fulfill user needs as stated in the software specifications.

Software model


Functional user requirements
Software
Functional process type (1)
Sub-process
Data movement type (2)
and
Data manipulation type (3)

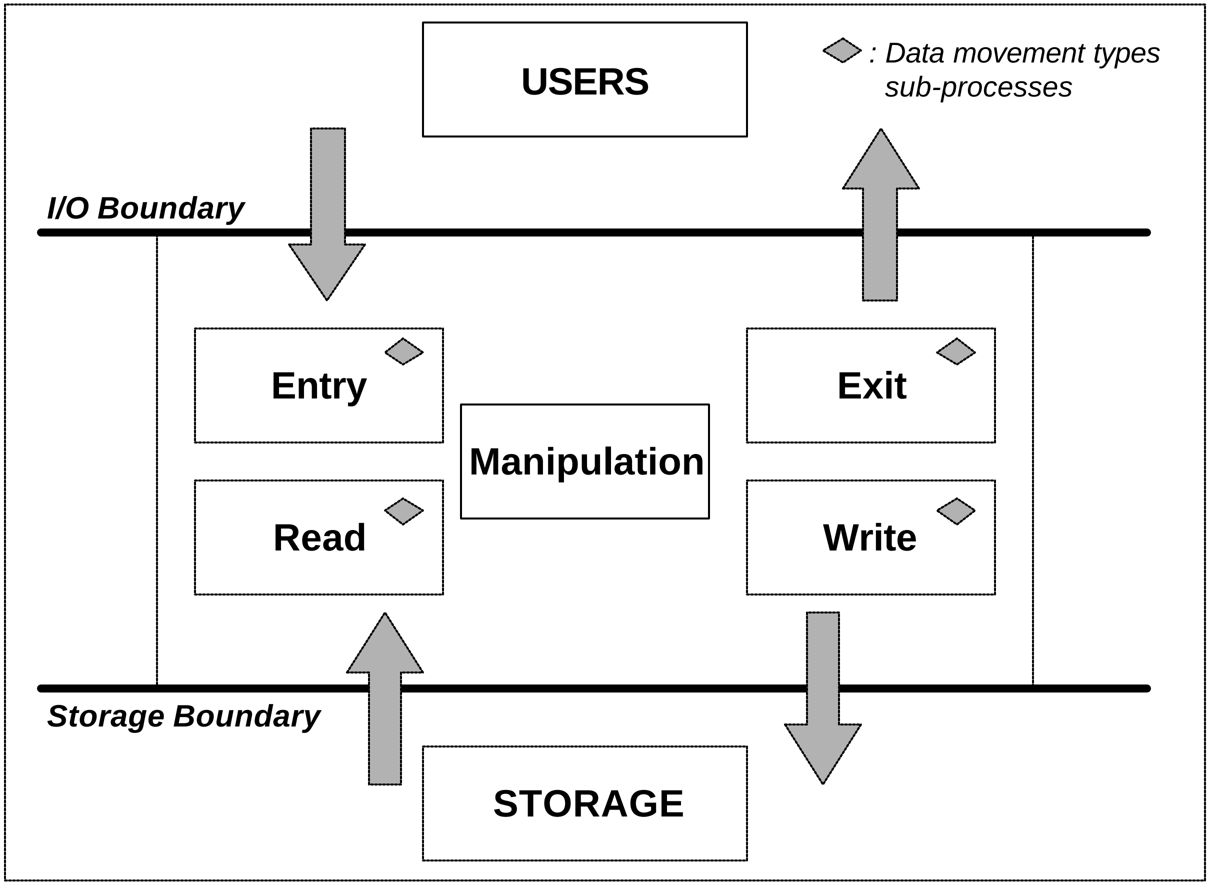
(1): *A sequence of data movement and transformation sub-process steps, triggered event external to the software item, which is complete when the data processed consistent with respect to the external triggering event.*

(2): *A sub-process entering, exiting, reading or writing a data*

(3): *A sub-process transforming a data item to create another one.*

COSMIC-FFP generic software model

Software model (cont.)



COSMIC-FFP sub-process types

Software complexity model

- ***What are the relationships between the functional processes?***

  Three types of relationships:

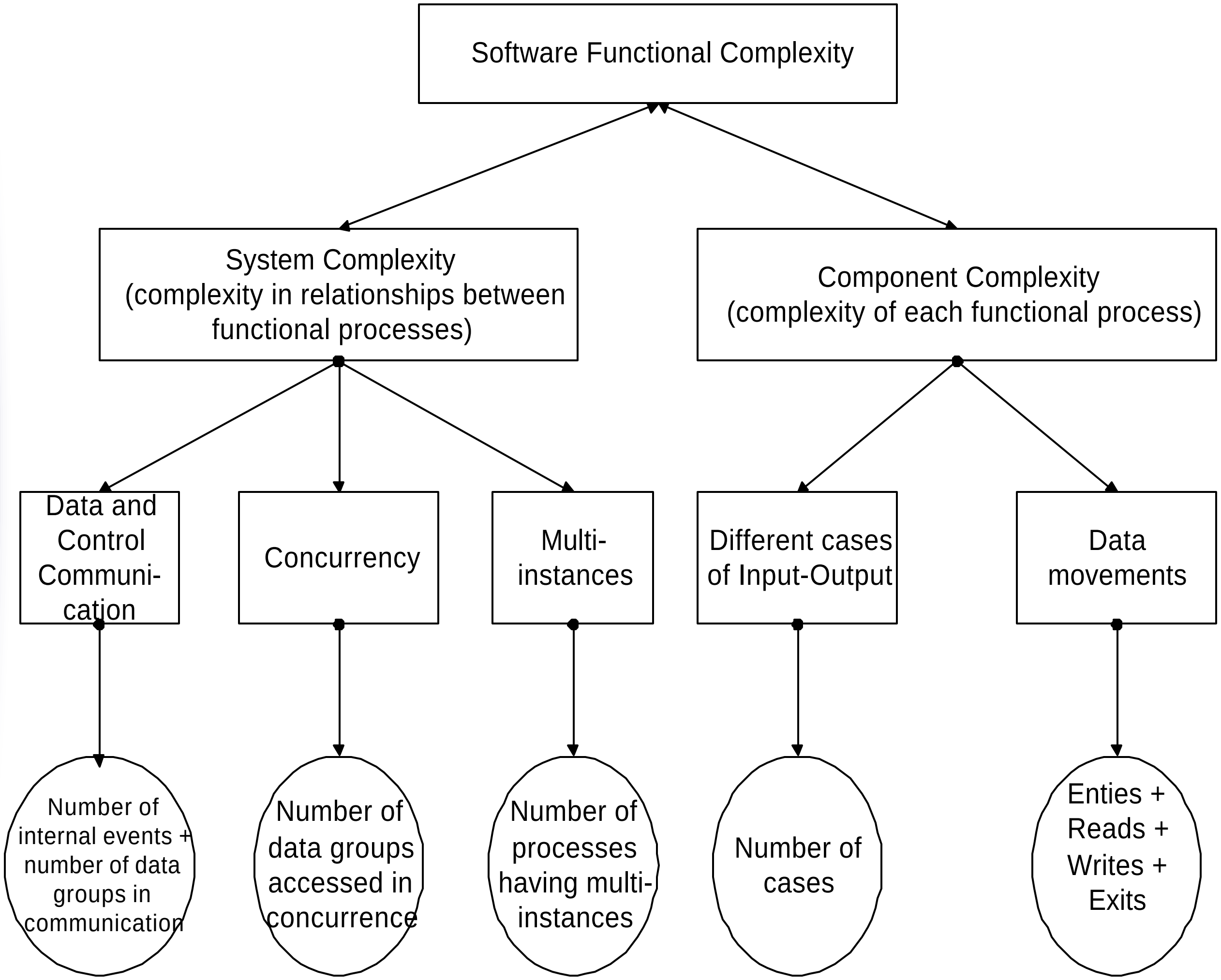
  - **Control and data communication**
  - **Concurrency**
  - **Multi-instances**

- ***What are the tasks carried out by a process?***

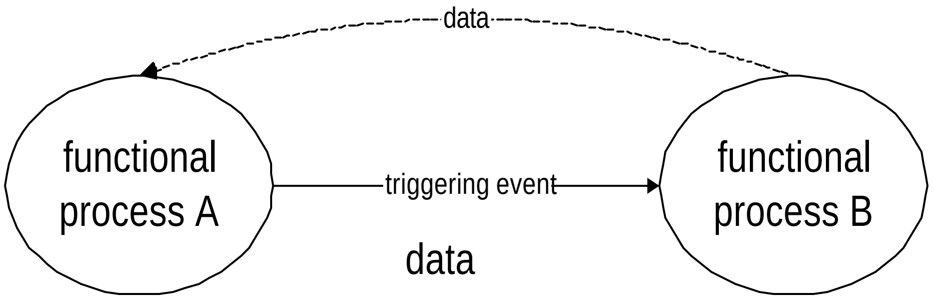
  - **Different cases:**

    If (condition on input) Then (expected output)
  - **Data movements: entry, read, write, exit**

Software complexity model


Software Functional Complexity
System Complexity
(complexity in relationships between functional processes)
Component Complexity
(complexity of each functional process)
Data and Control Communi-cation
Concurrency
Multi-instances
Different cases of Input-Output
Data movements
Number of internal events + number of data groups in communication
Number of data groups accessed in concurrence
Number of processes having multi-instances
Number of cases
Enties + Reads + Writes + Exits

Control and data communication

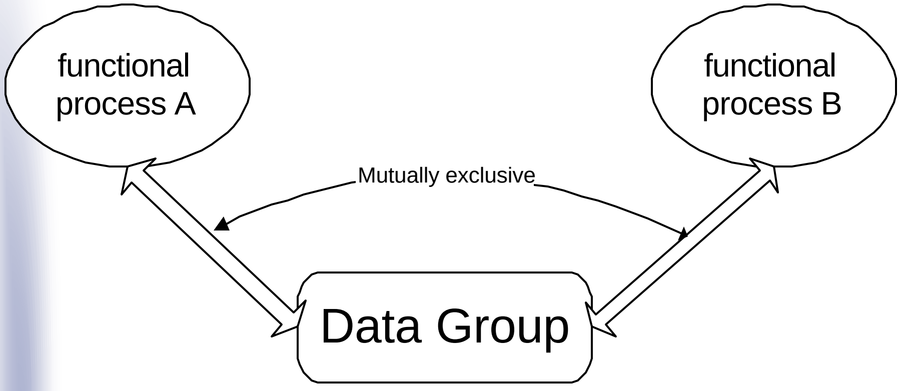


- FPA: data communication, distributed data processing, on-line data entry, complex processing

- Asset-R: parallelism and synchronization.

CDC = å(number of data groups in communication between two processes + 1)

Concurrency



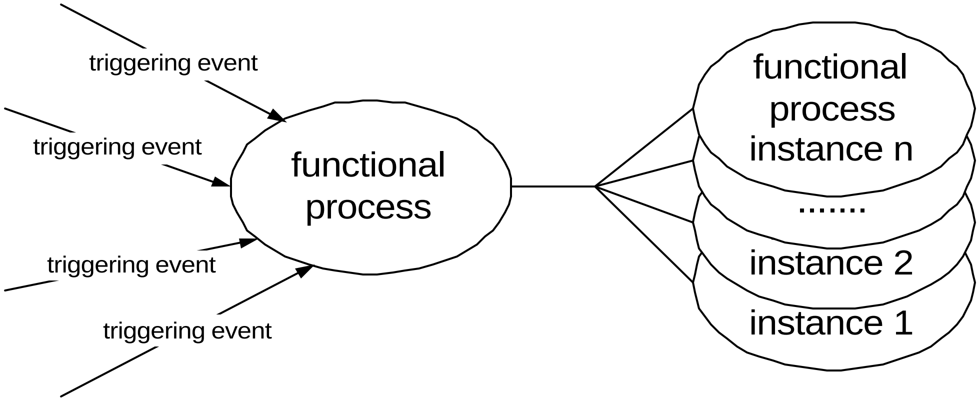
More than one process access simultaneously one data group in the mutually exclusive mode

- FPA: on-line update

- Asset-R: Concurrency

CC = number of data groups accessed concurrently in the system

Multi-instances



- A functional process may be triggered by many different triggering events from the different resources or triggered many times by an event at different moments. When the functional process is working to respond to an event, another event may occur and request the response of the process (interrupt).

- Multi-instances may express a part of complex processing proposed by FPA.

MIC = number of processes having multi-instances

Different cases of a functional process

- **A process may have many variances (or “states”). A functional process may exhibit different behaviours in reaction to one event. Each of them can be interpreted as a case of process.**

  IF (condition on input values) THEN (expected outputs)

- **Assumption: not only the number of inputs, outputs makes difficulty in designing the process but also the number of cases.**

- **DCC= number of cases or number of**

  IF (condition on inputs) THEN (desired outputs)

Data movements

- **Data movements are a factor of complexity since they can be intuitively associated with the tasks that must be performed in the process. They indicate how many tasks must be done in the process.**

DMC = number of Entries + number of Exits + number of Reads + number of Writes

Estimating of software complexity

- Five simple measures of the five factors proposed above are used as the indicators of the effort required in different tasks in software development. We propose

  SFC = f(CDC, CC, MIC, DCC, DMC)

- Each of these can be used as independent measures to describe or quantify the different aspects of software

Conclusion

- **A simple measure has been proposed for each of the factors. These measures may be used independently for different purposes, and also used as parameters for assessing software functional complexity.**
- **The main purpose of this work was to establish and progressively build on a software functional complexity model rather than a complete measuring method of the still ill-defined global concept of software complexity.**
- **More empirical research is need to investigate the relationships between these factors to derive a unique measure for software functional complexity.**

About us

- **De Tran-Cao, PhD. candidate, UQAM**
  de.trancao@lrgl.uqam.ca
- **Alain Abran, professor, UQAM**
  abran.alain@uqam.ca
- **Ghislain Lévesque, professor, UQAM**
  levesque.ghislain@uqam.ca

Software Engineering Management Research Laboratory (LRGL). The University of Quebec at Montreal (UQAM)

www.lrgl.uqam.ca

Conclusion

- **Generic model: complexity of components and complexity of the system.**
- **The software functional model proposed by the COSMIC team was used as the basis for studying software functional complexity.**
    - **Component complexity: complexity in a functional process characterized by different cases of the functional process and data movements in the process.**
    - **System complexity: complexity in relationships between the functional processes characterized by control and data communication, concurrency and multi-instance.**