Mapping the OO-Jacobson Approach to Function Points

Thomas Fetcke, Alain Abran and
Tho-Hau Nguyen

Université du Québec à Montréal
Laboratoire de recherche en gestion des logiciels
Case postale 8888, succursale Centre-Ville
Montréal (Québec) Canada
H3C 3P8
e-Mail: fetcke@cs.tu-berlin.de
WWW: http://www.cs.tu-berlin.de/~fetcke

1996 June 20

Introduction

Objectives

- Application of Function Points for object-oriented software engineering
- Use of OO-Jacobson method
- Goal: Count in early project phases
- Not included: real-time characteristics

Challenges with OO

- OO methods differ in the first steps very much
- Different models are used to find the objects
- No objects are identified in the first steps

Introduction (cont'd)

Benefits from selecting the OO-Jacobson method

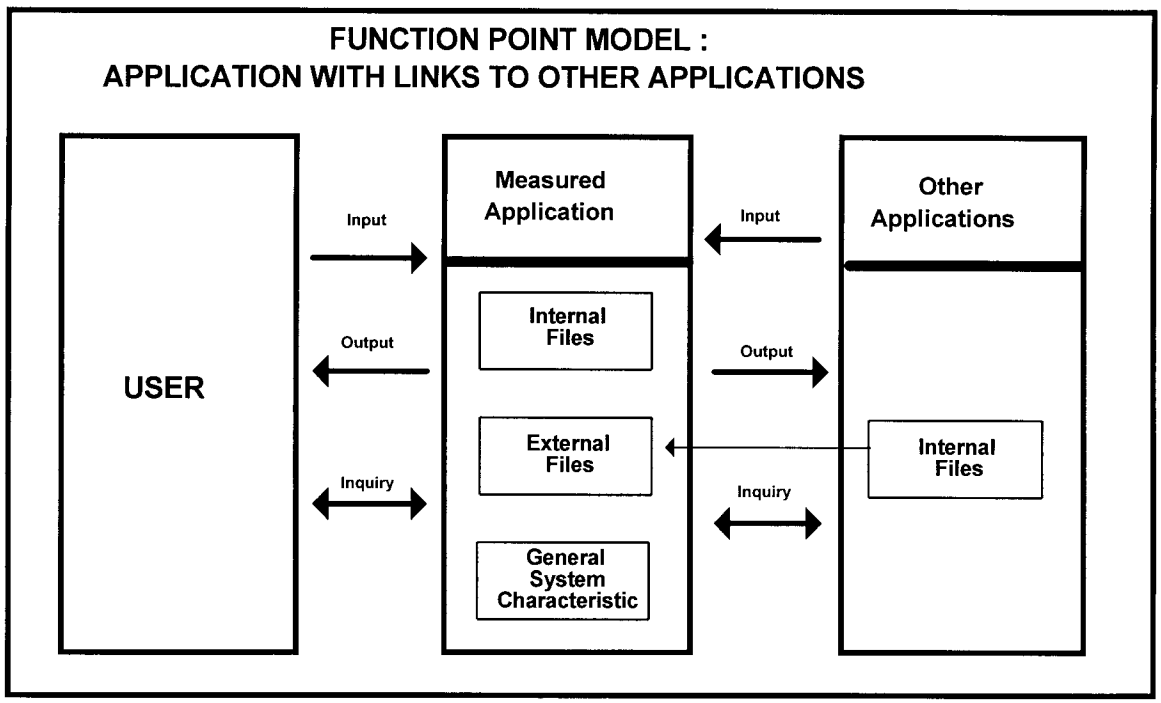
- OO-Jacobson gives a formal method from requirements to OO construction
- The viewpoint of OO-Jacobsson are "use cases" which are similar to the viewpoint of Function Points

Mapping of Concepts

- Can Function Points be counted from the OO requirements and analysis models?
- How can this be formalized?
- A formal mapping of concepts makes the count feasible and consistent.
- Four major steps in FPA are considered
  1. Boundary concept
  2. The items to count within the boundary
  3. Classification of the items
  4. Weighting the items
- Steps that remain independent:
  - The type of count (project, application)
  - The 14 general system characteristics

Step 1: Boundary Concepts

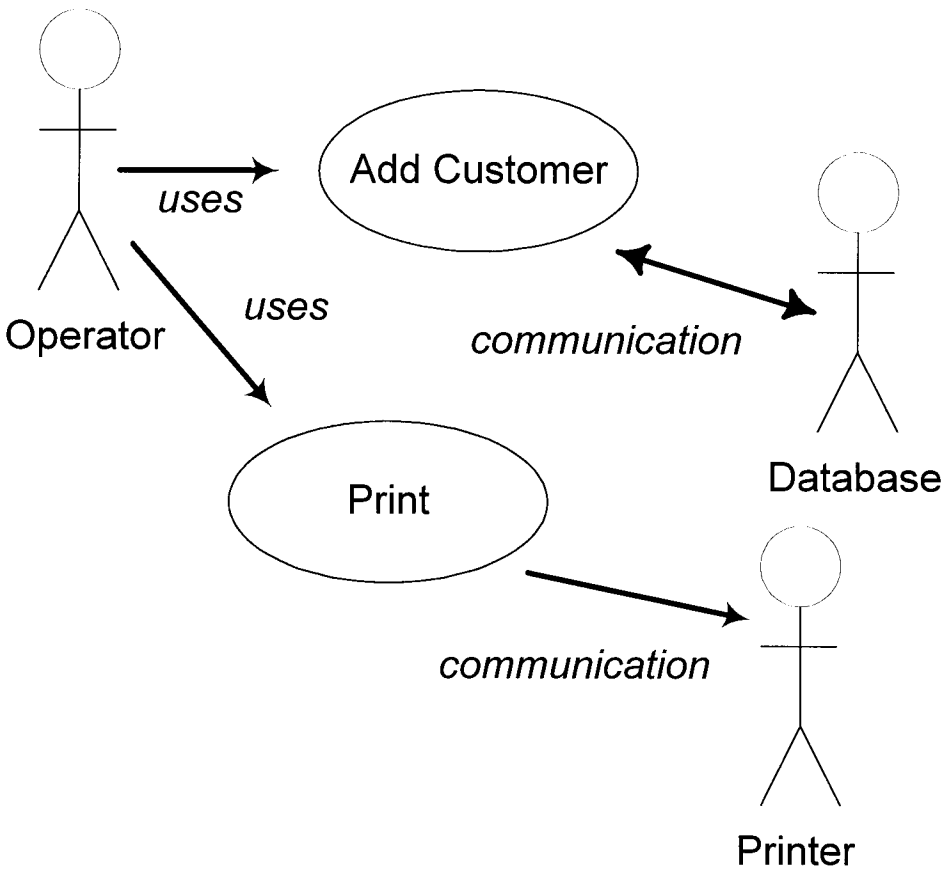
The Function Point counting boundary indicates the border between the project or application being measured and the external applications or user domain



Step1: Boundary Concepts in OO-Jacobson

The "use case model" is the corresponding concept to the boundary in OO-Jacobson

- Actors represent "things" outside
- Use cases represent the functionality

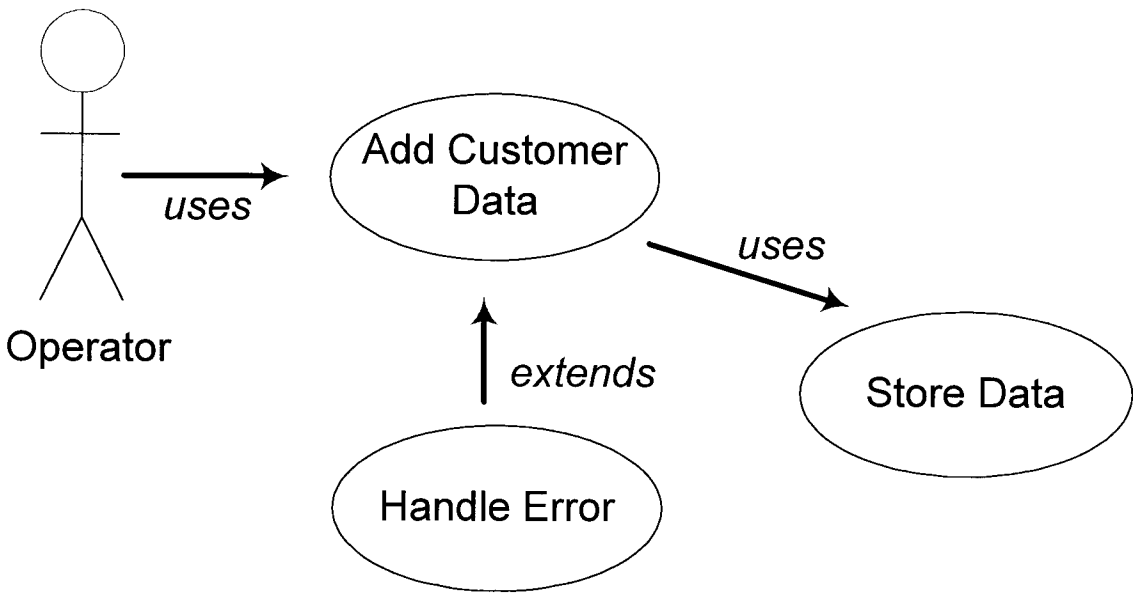


Step 1: Rules for the Counting Boundary

- Actors relate to the Function Point concept of users and external applications
- But the concept of actors contains also underlying systems, system environment and hardware
- To construct the boundary, the set of actors has to be analyzed
- Proposed Rules:
  - Accept human actors as users
  - Accept non-human actors as external applications if they are not part of the environment or underlying application

Use Cases and Transactions

- Not every use case delivers functionality to the user
- A use case may use other *abstract* use cases
- A use case may be extended



Step 2a: Rules for Use Cases

- Use cases relate to FP-transactions if they deliver functionality to the user
- Proposed Rules:
  - Use cases directly related to actors accepted in step 1 are candidates.
  - Use cases extending these candidates are also considered.
- Consequences:
  - Abstract use cases are not counted as FP-transactions.
  - Use cases exclusively related to the environment, e. g. "Store Entry in the Database", are not counted.

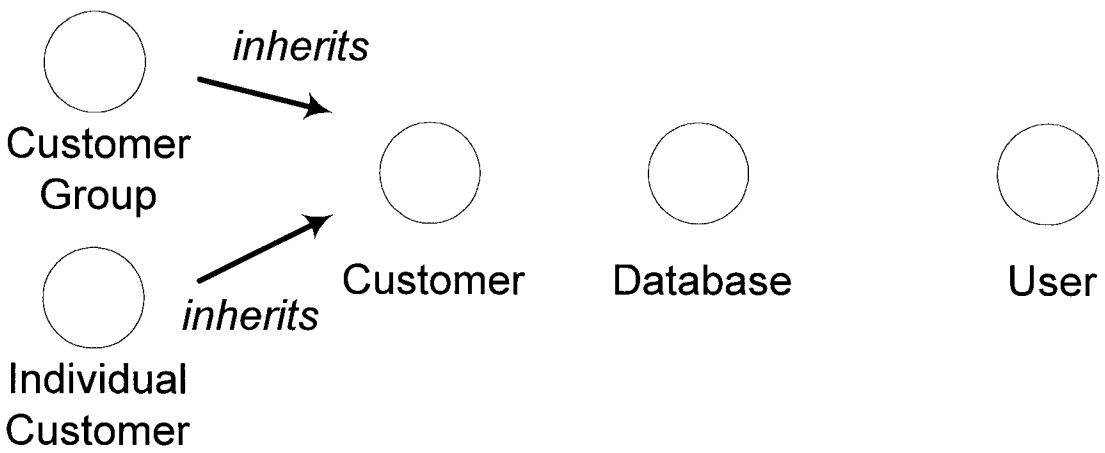
Step 2b: Files

- Objects are the related concept.
- Depending on the project phase, models with different levels of detail are available:
- The Domain Object Model is an optional part of the requirements model.
- The Analysis (object) Model is an essential part of the analysis model.
- Depending on which model is available, the determination of objects that relate to FP-files has to be different.

Domain Objects and Files

(i) Domain Object Model

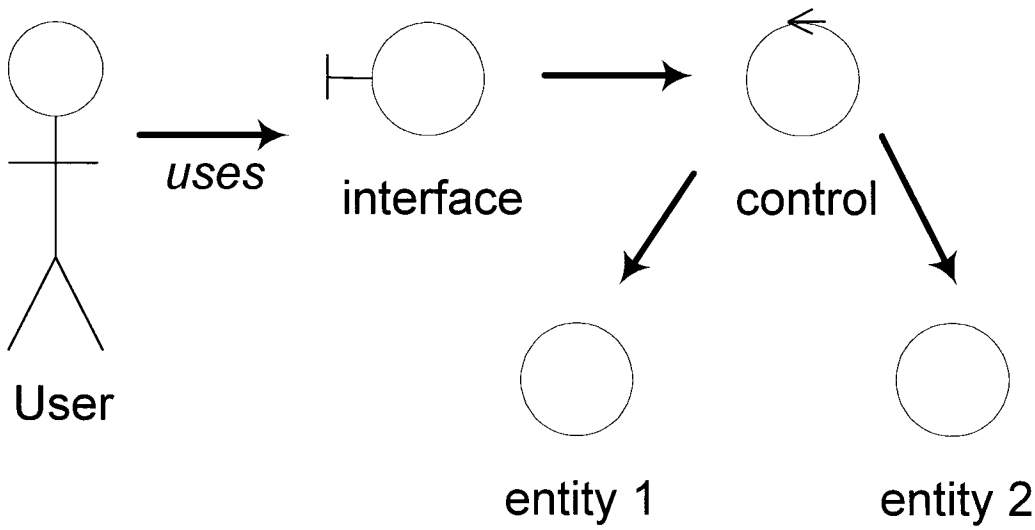
- All (data) concepts which are relevant for the application are identified.
- These contain data entities, but also other objects that do not relate to the FP file concept, e. g. "Database System" or "User".



Analysis Objects and Files

(ii) Analysis (object) Model

- The use case model is transformed into typed objects
  - entity objects
  - interface objects
  - control objects
- Entity objects model information held in the system, they correspond to the file concept.



Step 2b: Rules for Objetcs

(i) Proposed Rules:

- All Domain Objects are candidates for files.
- Each Domain Object has to be analyzed according to Function Point counting rules. This will be done in step 3.
- Domain objects like "Database System" or "Printer" will not be counted as FP-files.

(ii) Proposed Rules:

- All entity objects are candidates for files.
- Interface and control objects will not be counted.
- The set of candidates will be evaluated in step 3, when the type of file (ILF or EIF) is determined.

Step 3: Types of the Items

- Step 3a: Candidate use cases are evaluated with the Function Point rules for EI, EO and EQ
- Step 3b: Candidate objets are evaluated with Function Point rules for ILF and EIF.
- The evaluation is based on the information in the use case requirements description and the object model.

Step 4: Weights of the Items

- Weights are determined with the appropriate Function Point rules
- The information provided in the requirements and analysis model is not sufficient for the application of the FP rules for weights.
- Based on that information, estimation of the weights may be possible.
- For the detailed application of the weight rules, **design** models of use cases and analysis objects are required.

Experimental Results

- 3 industry projects have been counted following the proposed rules.
- The calculated size of the projects in Function Points was

Project 1	265
Project 2	181
Project 3	215

- Use case and Domain Object models were available for project 1. From the detailed requirements descriptions, weights have been determined.
- Projects 2 and 3 did not provide an OO-Jacobsson object model and the use case documentation was less detailed. The counts for these projects are therefore only estimates.