University of Waterloo

Federal University of Vale do São Francisco

Requirements Specification Quality

How to Measure Quality of Requirements Specifications?

Ricardo Argenton Ramos

rargenton@uwaterloo.ca

Outline

- The AIRDoc Approach.
- My Postdoc Research - How to Measure Quality of Requirements Specifications?
- My 3 Works on progress:
  - QUALISIS-Br: An Approach to Improve the Quality of Brazilian Health Information Systems
  - SE-Origami: A method to Teach Software Engineering Process in a Classroom.
  - Requirement Elicitation Process for a Data Management on a Biofactory

The AIRDoc

Phd thesis defence at october, 2009.

Supervisor: Jaelson Castro – UFPE
Co-supervisor: João Araújo - UNL

AIRDOC is a acronym of:

Approach to

Improve

Requirement

Documents

Introduction (*the problem*) – 1/2

- Over the past few years, a set of typical issues seems to plague the Use Case Models. For example:
  - □ Use case that have been abandoned and are no longer meaningful,
  - □ Use case descriptions that are unnecessarily long and complex,
  - □ Information that is duplicated, scattered, tangled,
  - □ Among others …

[Lilly 1999]

Introduction (*the problem*) – 2/2

The removal of these problems in early stages of software development process reduces the costs associated with software changes. These cost reductions could be three to six times more in later stages than during requirements activities [Pressman 2005], [Sommerville, 2004] .

Brooks adds, “The hardest single part of building a software system is deciding precisely what to build.... No other part of the work so cripples the resulting system if it is done wrong. No other part is more difficult to rectify later.”

How to Solve these Problems?

- Inspection techniques ?

[Travassos et al., 1999] [Fagan, 1986]

- Aspect Orientation ?

[Moreira et al., 2005], [Silva, L.; Leite, J. 2005], [Sousa, G; Castro, 2004]

- Good practices ?

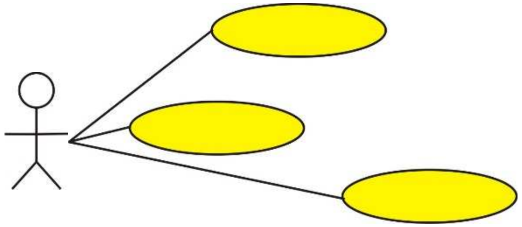
[IEEE Std 830-1998], [IEEE 1061, 1998], [Firesmith, 2007]

- Metrics ?

[Fenton and Neil, 2000]

The Proposed Solution

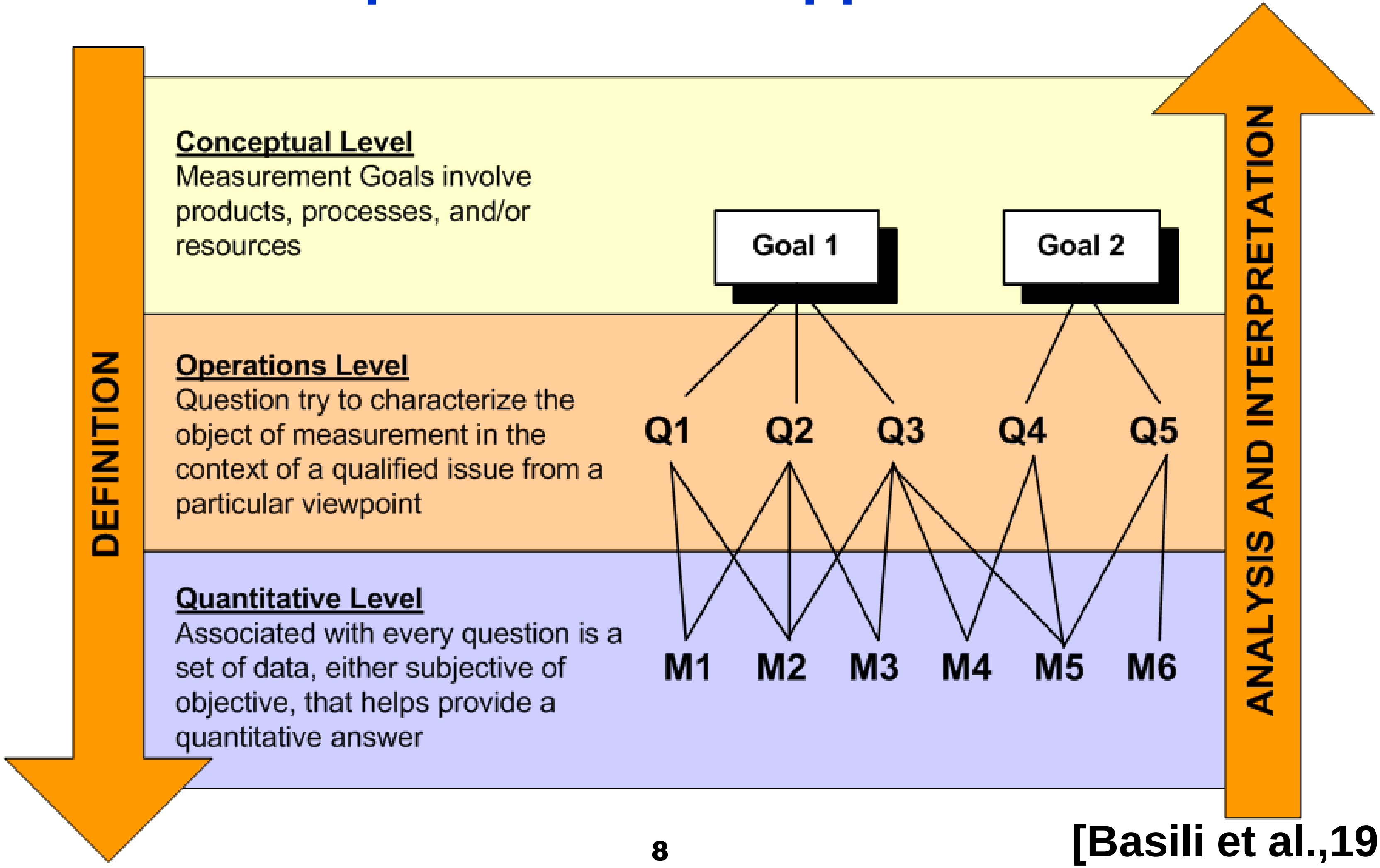
We propose to use metrics to discover the potential problems



- The use of software metrics reduces subjectivity in the assessment and control of software quality by providing a quantitative basis for making decisions about software quality [IEEE 1061, 1998]

The Proposed Solution

We use the Goal Question Metrics approach to help the metrics application



Conceptual Level
Measurement Goals involve products, processes, and/or resources

Operations Level
Question try to characterize the object of measurement in the context of a qualified issue from a particular viewpoint

Quantitative Level
Associated with every question is a set of data, either subjective of objective, that helps provide a quantitative answer

[Basili et al.,1994]

The Proposed Solution

We use the framework proposed by the standard [IEEE 1061, 1998]

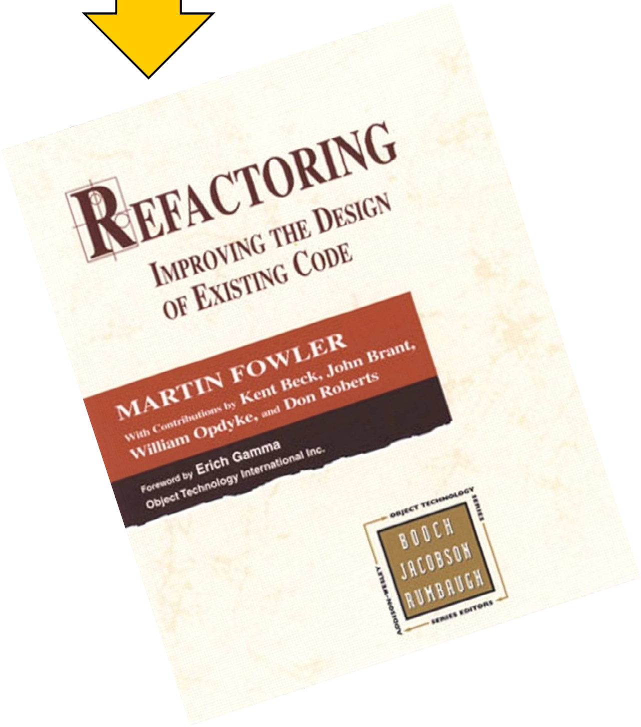
The GQM approach needs to have a quality model to achieve the goal, question and metrics definitions.

IEEE Standard for a Software Quality Metrics Methodology

The Proposed Solution

Once, we have measured the appropriate quality factor, our AIRDoc will be able to possible detect some potential problem.

Insignts from

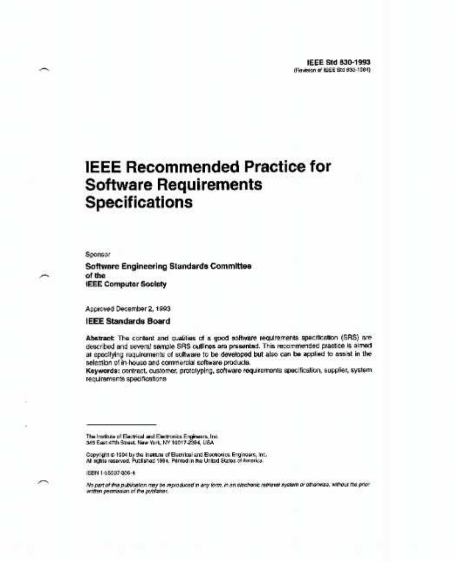

REFACTORING
IMPROVING THE DESIGN OF EXISTING CODE
MARTIN FOWLER
With Contributions by Kent Beck, John Brant, William Opdyke, and Don Roberts
Foreword by Erich Gamma
Object Technology International Inc.
OBJECT TECHNOLOGY SERIES
BOOCH JACOBSON RUMBAUGH
SERIES EDITORS
ADDISON-WESLEY

We propose the solution to the problems in terms of some refactorings to be performed

”Refactoring is the process of restructuring existing computer code without changing its external behavior”

Some Recommended Practice for Software Requirements Specifications

- **Correct**;
- **Unambiguous**;
- **Complete**;
- **Consistent**;
- **Ranked for importance and/or stability**;
- **Verifiable**;
- **Modifiable**;
- **Traceable**;



IEEE Std 830-1993
(Revision of IEEE Std 830-1984)

IEEE Recommended Practice for Software Requirements Specifications

Sponsor

Software Engineering Standards Committee
of the
IEEE Computer Society

Approved December 2, 1993

IEEE Standards Board

Abstract: The content and qualities of a good software requirements specification (SRS) are described and several sample SRS outlines are presented. This recommended practice is aimed at specifying requirements of software to be developed but also can be applied to assist in the selection of in-house and commercial software products.
Keywords: contract, customer, prototyping, software requirements specification, supplier, system requirements specifications

The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street, New York, NY 10017-2394, USA

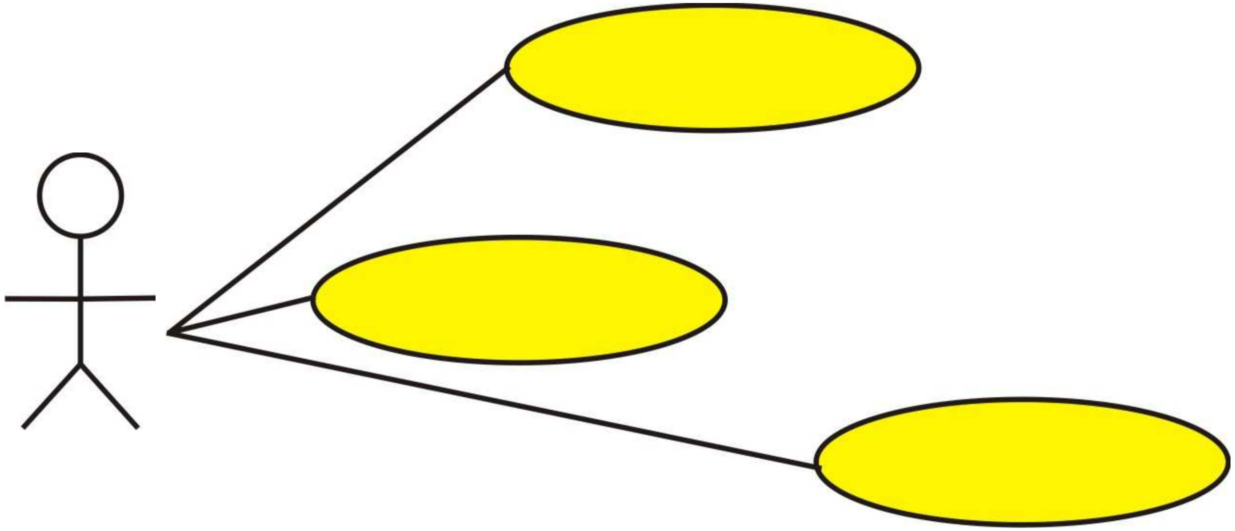
Copyright © 1994 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 1994. Printed in the United States of America.

ISBN 1-55937-395-4

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.

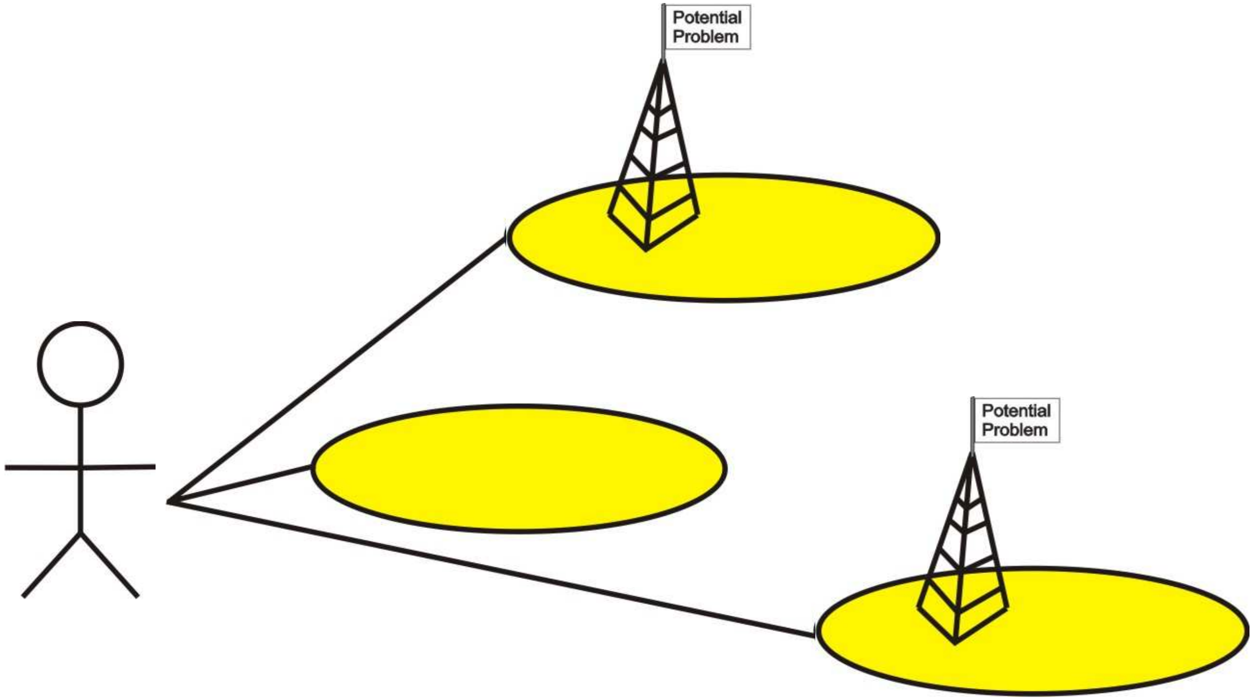
[IEEE Std 830-1998]

AIRDoc - Approach to Improve the Quality of Requirement Documents



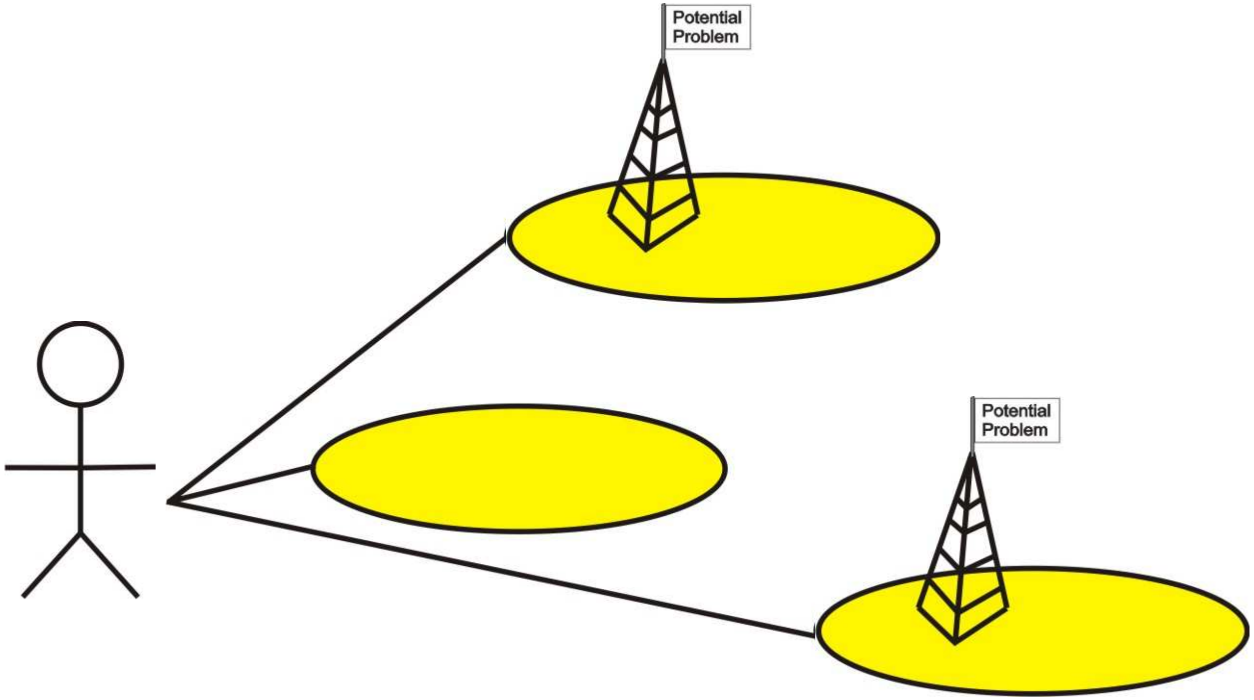
“funny picture”

AIRDoc - Approach to Improve the Quality of Requirement Documents



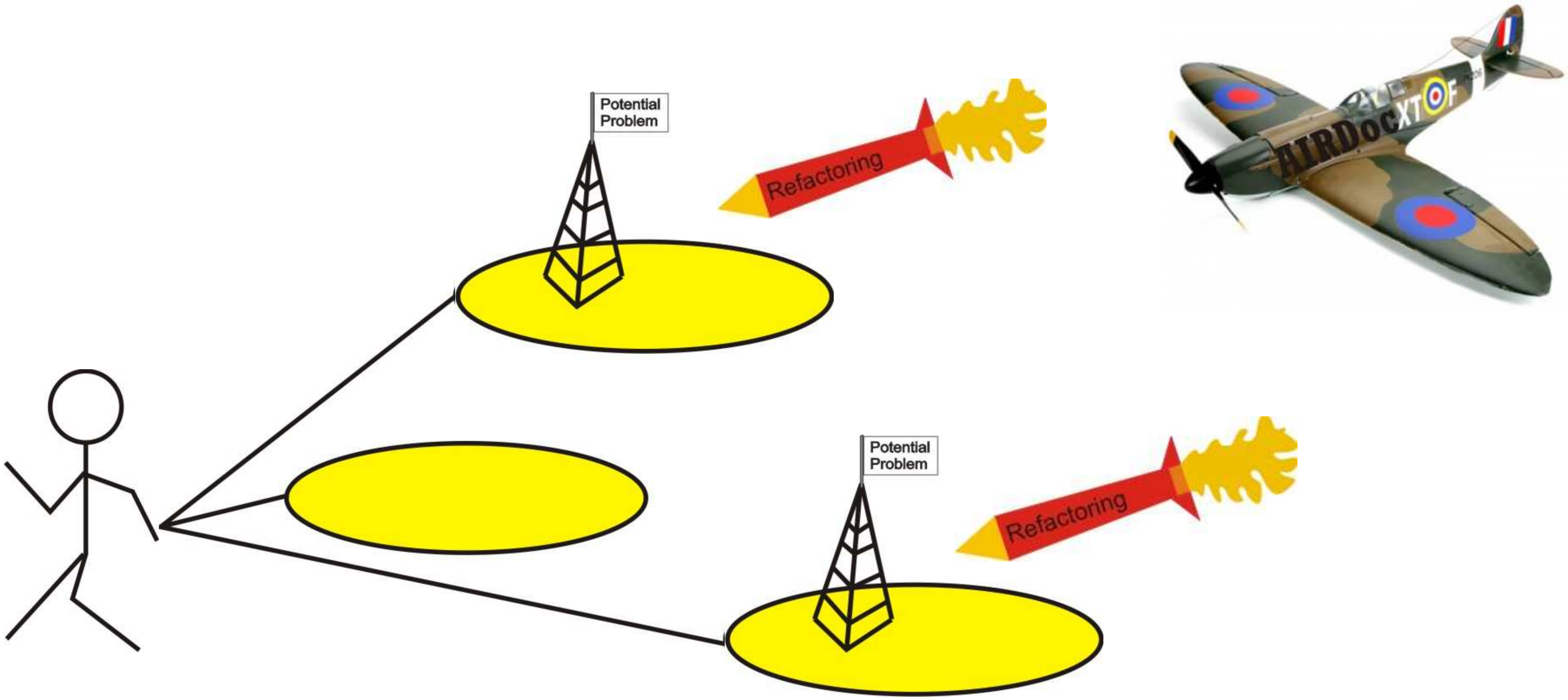
“funny picture”

AIRDoc - Approach to Improve the Quality of Requirement Documents



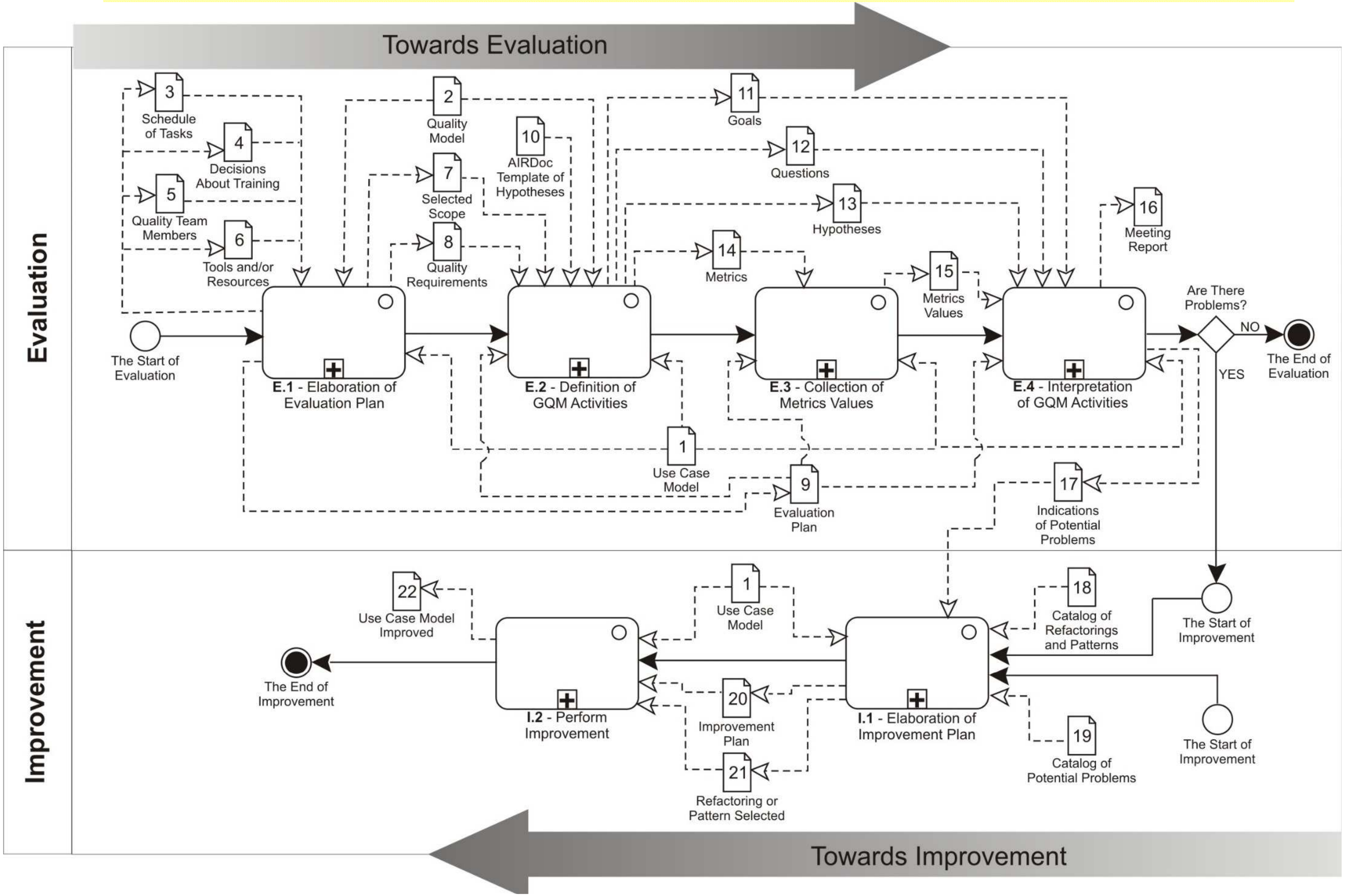
"funny picture"

AIRDoc - Approach to Improve the Quality of Requirement Documents



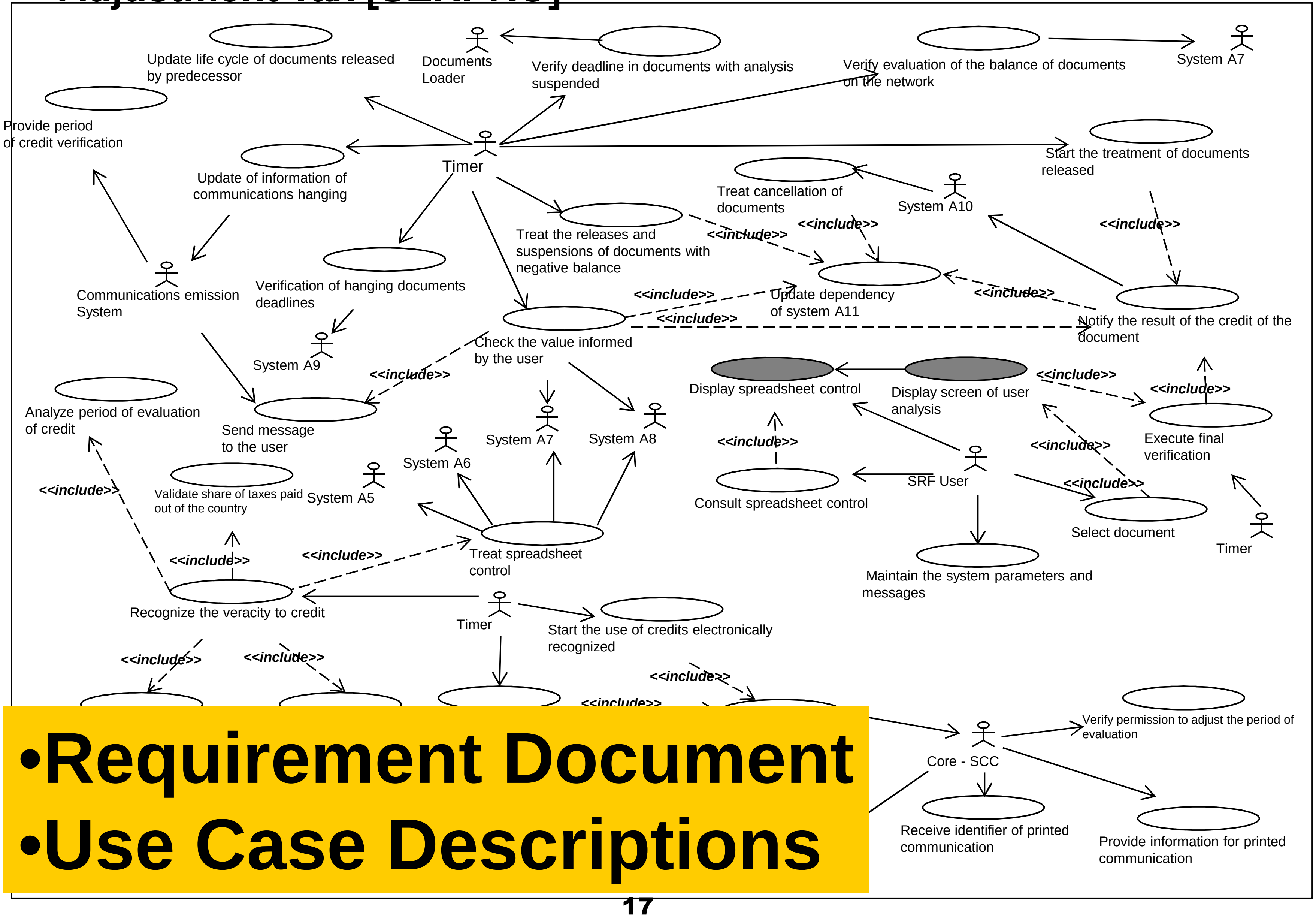
“funny picture”

The AIRDoc *“boring picture”*



A Running Example

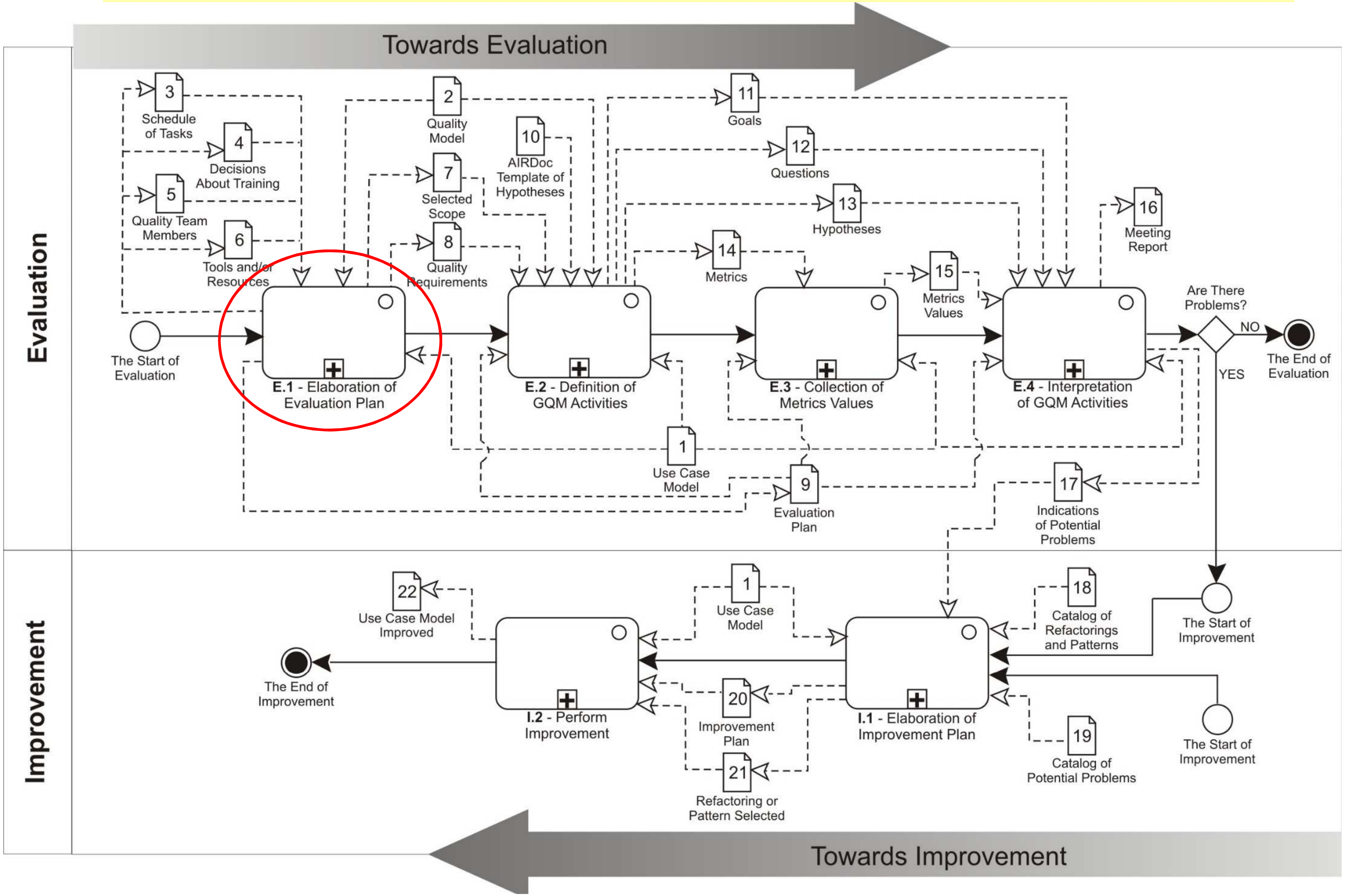
Adjustment Tax [SERPRO]



Update life cycle of documents released by predecessor
Documents Loader
Verify deadline in documents with analysis suspended
Verify evaluation of the balance of documents on the network
System A7
Provide period of credit verification
Timer
Update of information of communications hanging
Start the treatment of documents released
Treat cancellation of documents
System A10
<<include>>
<<include>>
<<include>>
Treat the releases and suspensions of documents with negative balance
Communications emission System
Verification of hanging documents deadlines
<<include>>
Update dependency of system A11
<<include>>
<<include>>
Notify the result of the credit of the document
Check the value informed by the user
System A9
<<include>>
Display spreadsheet control
Display screen of user analysis
<<include>>
<<include>>
Analyze period of evaluation of credit
Send message to the user
System A7
System A8
<<include>>
Execute final verification
System A6
Validate share of taxes paid out of the country
System A5
<<include>>
Consult spreadsheet control
SRF User
<<include>>
<<include>>
Select document
Timer
<<include>>
<<include>>
Treat spreadsheet control
Maintain the system parameters and messages
Recognize the veracity to credit
Timer
Start the use of credits electronically recognized
<<include>>
<<include>>
<<include>>
<<include>>
Verify permission to adjust the period of evaluation
Core - SCC
Receive identifier of printed communication
Provide information for printed communication

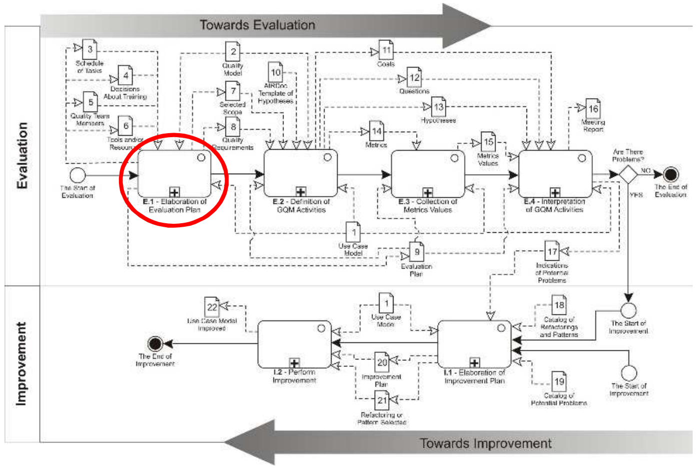
- **Requirement Document**
- **Use Case Descriptions**

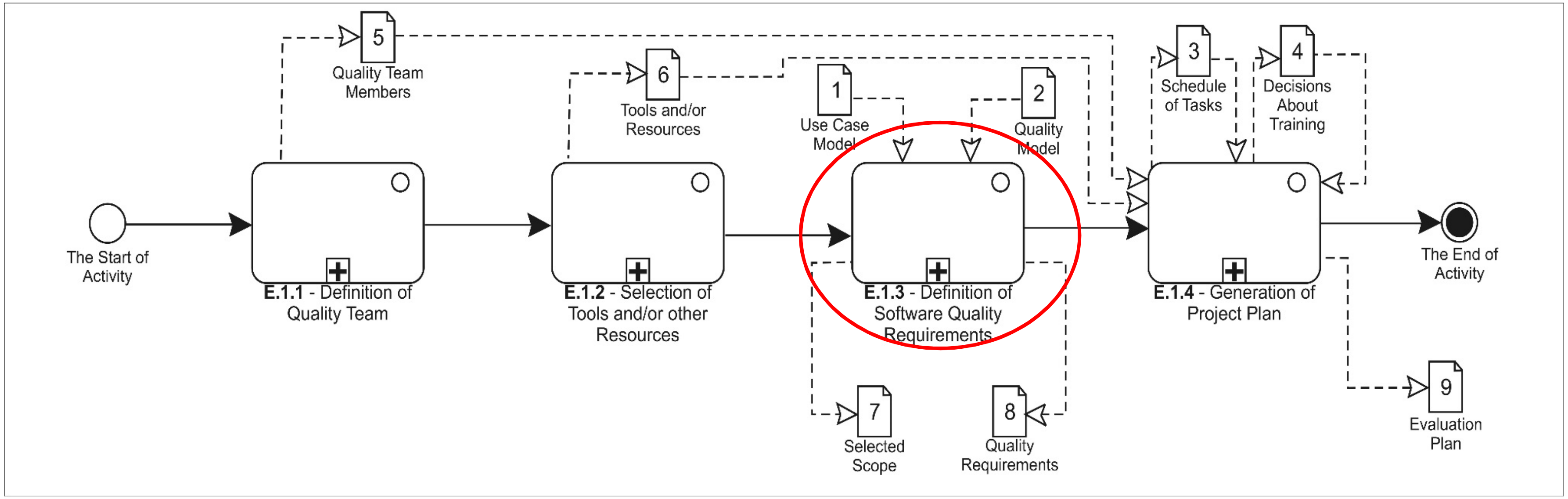
The AIRDoc



Towards Evaluation
Evaluation
3
Schedule of Tasks
4
Decisions About Training
5
Quality Team Members
6
Tools and/or Resources
2
Quality Model
7
Selected Scope
8
Quality Requirements
10
AIRDoc Template of Hypotheses
11
Goals
12
Questions
13
Hypotheses
14
Metrics
15
Metrics Values
16
Meeting Report
The Start of Evaluation
E.1 - Elaboration of Evaluation Plan
E.2 - Definition of GQM Activities
E.3 - Collection of Metrics Values
E.4 - Interpretation of GQM Activities
Are There Problems?
NO
YES
The End of Evaluation
1
Use Case Model
9
Evaluation Plan
17
Indications of Potential Problems
Improvement
22
Use Case Model Improved
1
Use Case Model
18
Catalog of Refactorings and Patterns
The Start of Improvement
The End of Improvement
I.2 - Perform Improvement
I.1 - Elaboration of Improvement Plan
20
Improvement Plan
21
Refactoring or Pattern Selected
19
Catalog of Potential Problems
The Start of Improvement
Towards Improvement

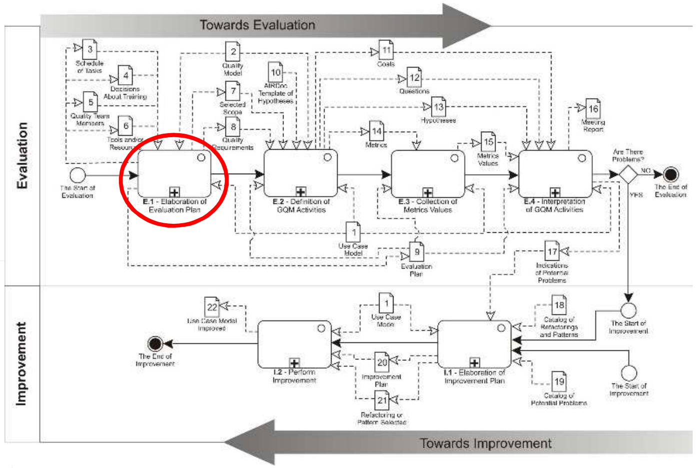
Elaboration of Evaluation Plan

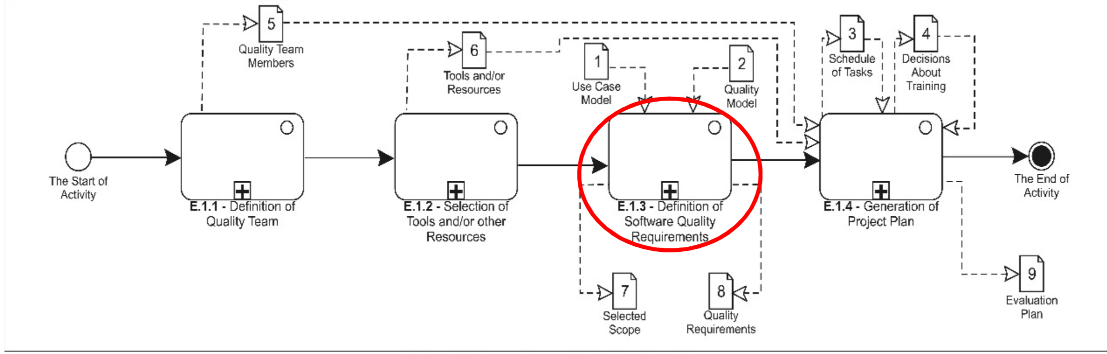




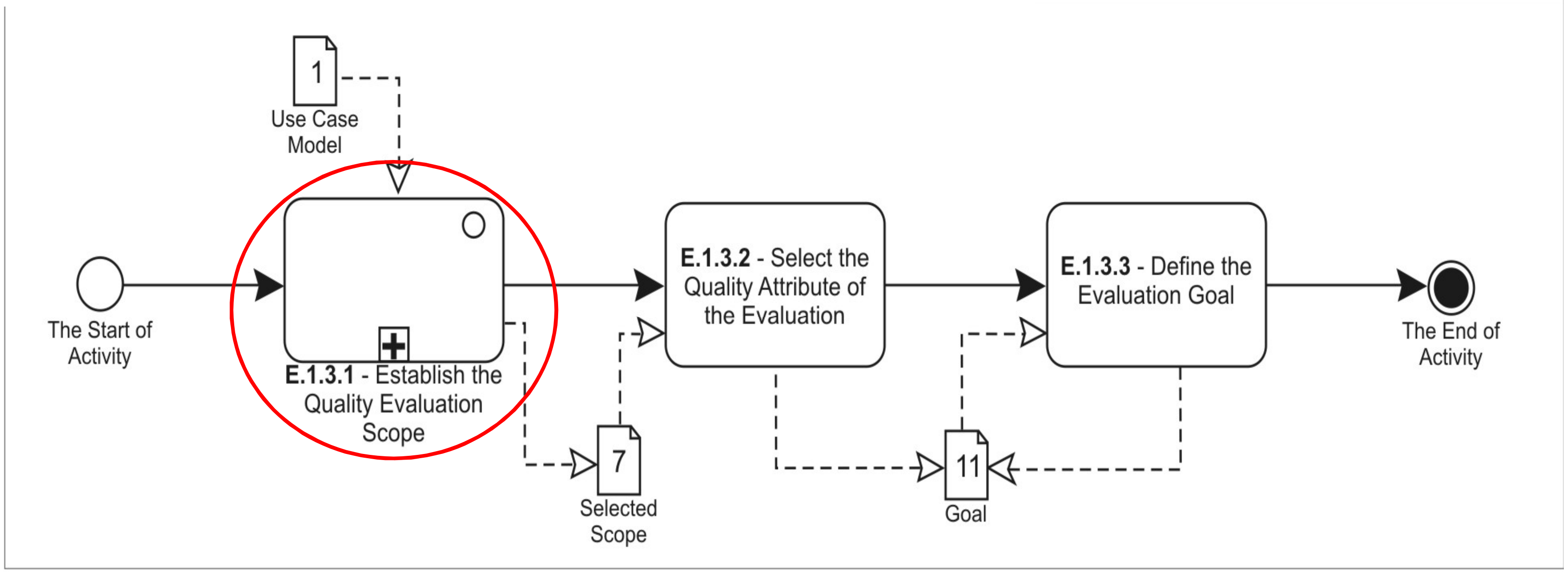
E.1 - Elaboration of Evaluation Plan

Definition of Software Quality Requirements



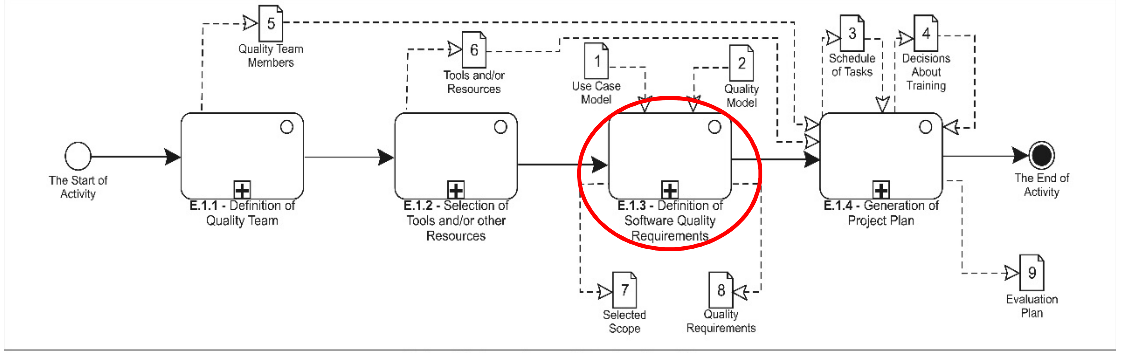


E.1 - Elaboration of Evaluation Plan

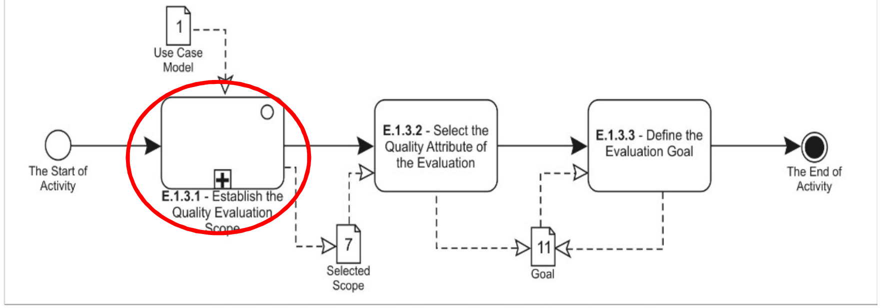

1
Use Case Model
The Start of Activity
E.1.3.1 - Establish the Quality Evaluation Scope
E.1.3.2 - Select the Quality Attribute of the Evaluation
E.1.3.3 - Define the Evaluation Goal
The End of Activity
7
Selected Scope
11
Goal

E.1.3 - Definition of Software Quality Requirements

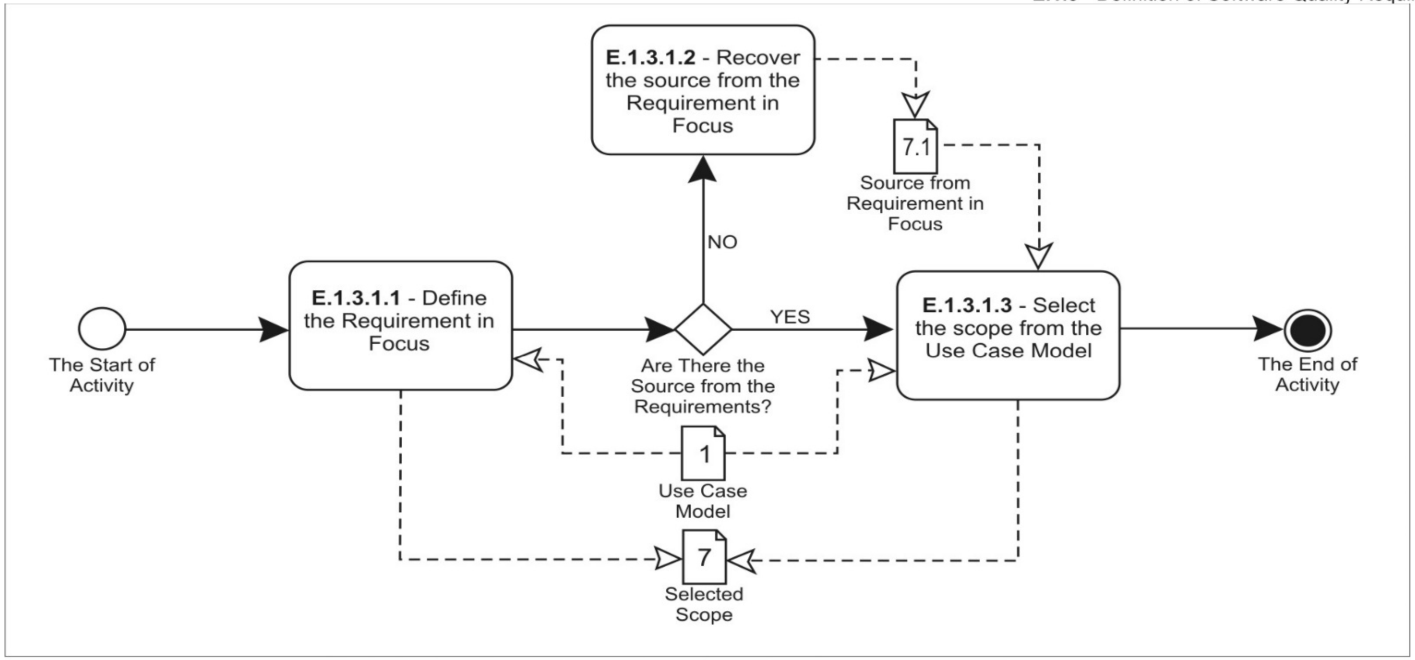
Establish the Quality Evaluation Scope



E.1 - Elaboration of Evaluation Plan

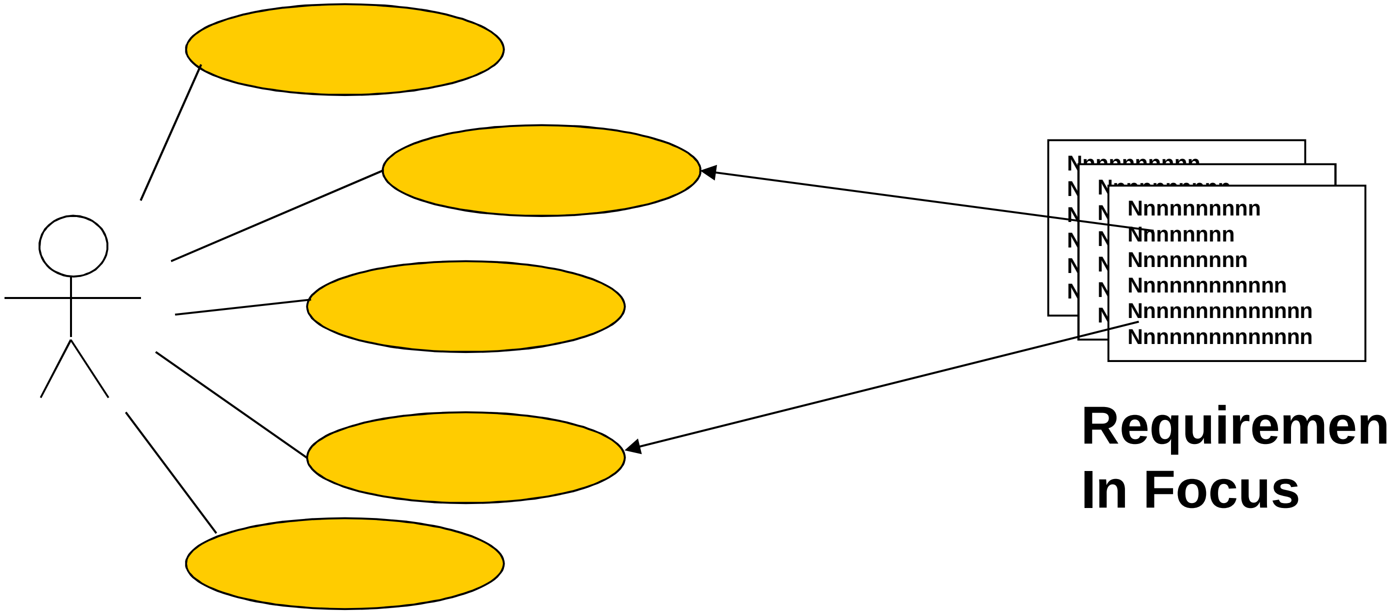

1
Use Case Model
The Start of Activity
E.1.3.1 - Establish the Quality Evaluation Scope
E.1.3.2 - Select the Quality Attribute of the Evaluation
E.1.3.3 - Define the Evaluation Goal
The End of Activity
7
Selected Scope
11
Goal

E.1.3 - Definition of Software Quality Requirements


E.1.3.1.2 - Recover the source from the Requirement in Focus
7.1
Source from Requirement in Focus
NO
YES
The Start of Activity
E.1.3.1.1 - Define the Requirement in Focus
Are There the Source from the Requirements?
E.1.3.1.3 - Select the scope from the Use Case Model
The End of Activity
1
Use Case Model
7
Selected Scope

E.1.3.1 - Establish the Quality Evaluation Scope

Mapping of: Requirement in Focus and Use Cases



Requirement
In Focus

Use Case Diagram

Template to Describe the Requirement in Focus

Requirement in focus	***< Identify the requirement in focus by a name. Create a list with the name(s) of use case(s) that are directly related with the requirement in focus. In some cases it is necessary to describe the steps that are in other use cases.>***
“Display Requirement”	use cases: 1 - “Display spreadsheet control”, 2 - “Display screen of user analysis”

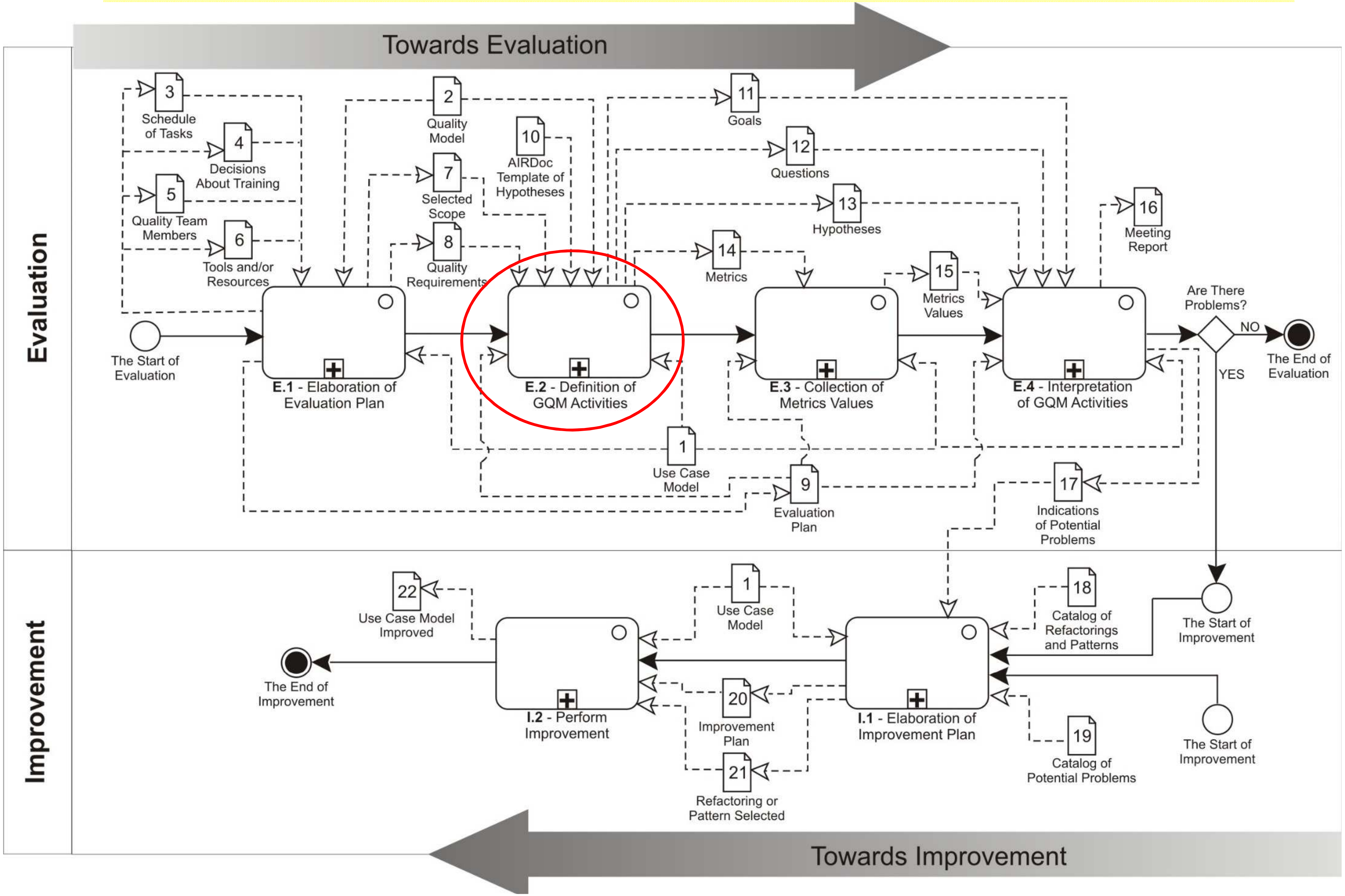
Template to describe the Source from the Requirement in Focus

Source from Requirement in Focus	*<Describe the information about the source, such as:* *- description about the stakeholder who originated the requirement;* *- type of source (interview, annotation, protocols, laws, rules, etc.);* *Include the description where the requirement existence is evidenced.>*
Source from "Display Requirement"	*Stakeholder - SRF User.* *The sources from the requirement in focus are dispersing on:* *- the laws nº 11.773/2008 (DOU of 18.9.2008) and 10.833/2003 (DOU of 30.12.2003)* *- meeting reports from the SERPRO Units.*

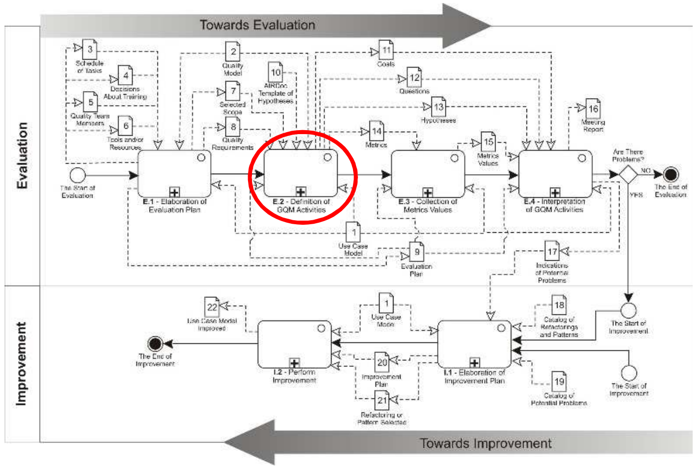
Template of the evaluation goal

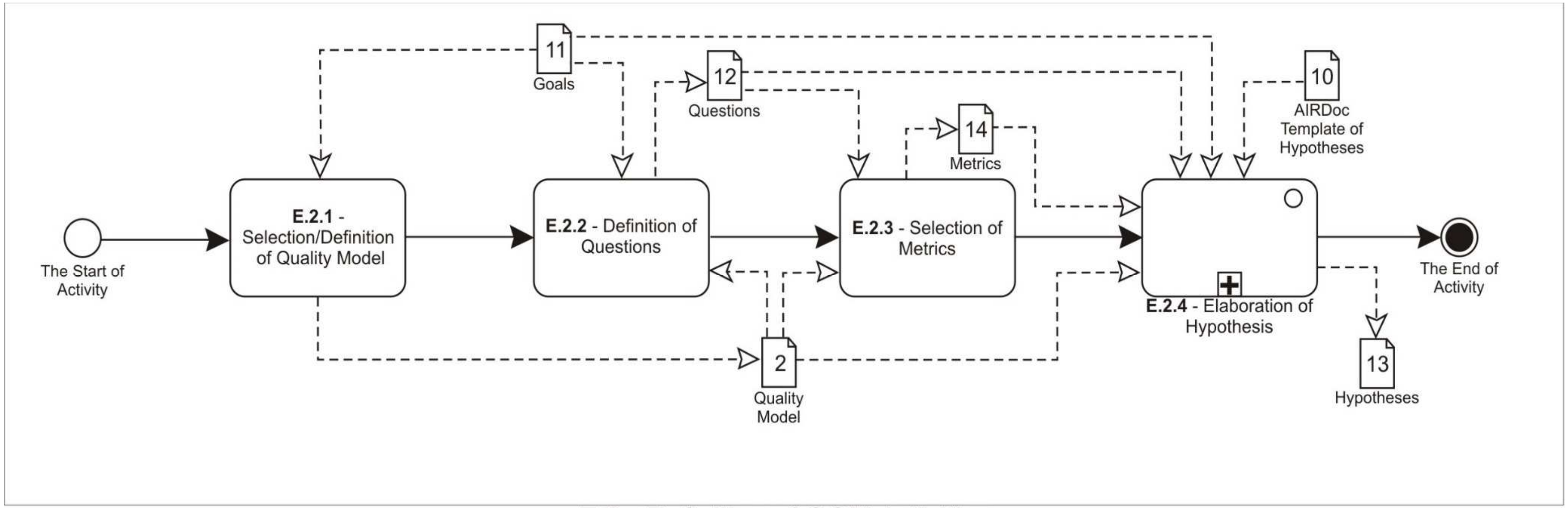
Evaluate in	*<scope>*	*<the quality attribute>*	of	*<requirement in focus>*
Evaluate in	Adjustment Tax	the maintainability	of	the display requirement

The AIRDoc


Towards Evaluation
Evaluation
3 Schedule of Tasks
4 Decisions About Training
5 Quality Team Members
6 Tools and/or Resources
2 Quality Model
7 Selected Scope
8 Quality Requirements
10 AIRDoc Template of Hypotheses
11 Goals
12 Questions
13 Hypotheses
14 Metrics
15 Metrics Values
16 Meeting Report
The Start of Evaluation
E.1 - Elaboration of Evaluation Plan
E.2 - Definition of GQM Activities
E.3 - Collection of Metrics Values
E.4 - Interpretation of GQM Activities
Are There Problems?
NO
YES
The End of Evaluation
1 Use Case Model
9 Evaluation Plan
17 Indications of Potential Problems
Improvement
22 Use Case Model Improved
1 Use Case Model
18 Catalog of Refactorings and Patterns
The Start of Improvement
The End of Improvement
I.2 - Perform Improvement
I.1 - Elaboration of Improvement Plan
20 Improvement Plan
21 Refactoring or Pattern Selected
19 Catalog of Potential Problems
The Start of Improvement
Towards Improvement

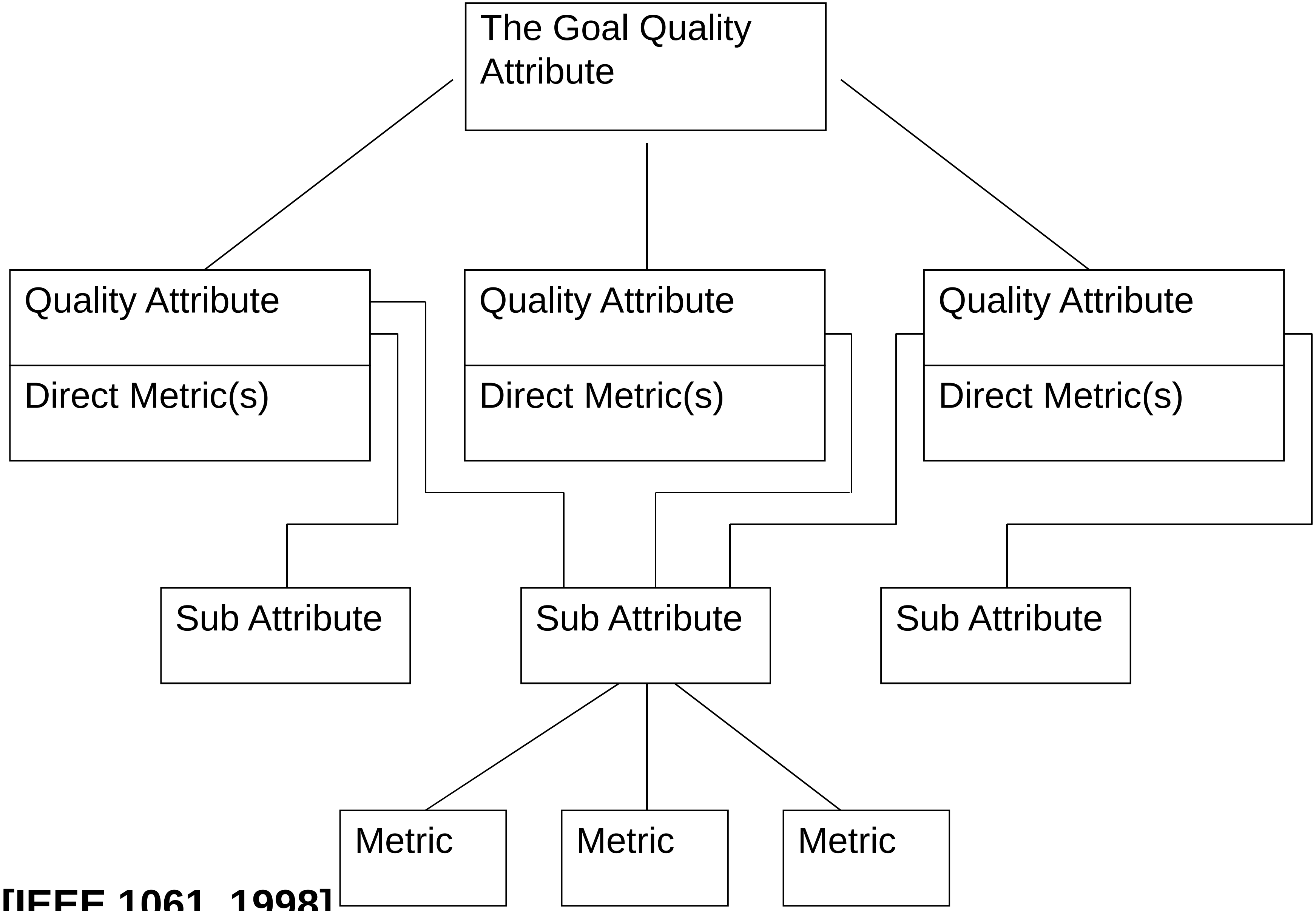
Definition of GQM Activities




11
Goals
12
Questions
14
Metrics
10
AIRDoc Template of Hypotheses
The Start of Activity
E.2.1 - Selection/Definition of Quality Model
E.2.2 - Definition of Questions
E.2.3 - Selection of Metrics
E.2.4 - Elaboration of Hypothesis
The End of Activity
2
Quality Model
13
Hypotheses

E.2 - Definition of GQM Activities

Selection/Definition of Quality Model



The Goal Quality Attribute

Quality Attribute

Direct Metric(s)

Quality Attribute

Direct Metric(s)

Quality Attribute

Direct Metric(s)

Sub Attribute

Sub Attribute

Sub Attribute

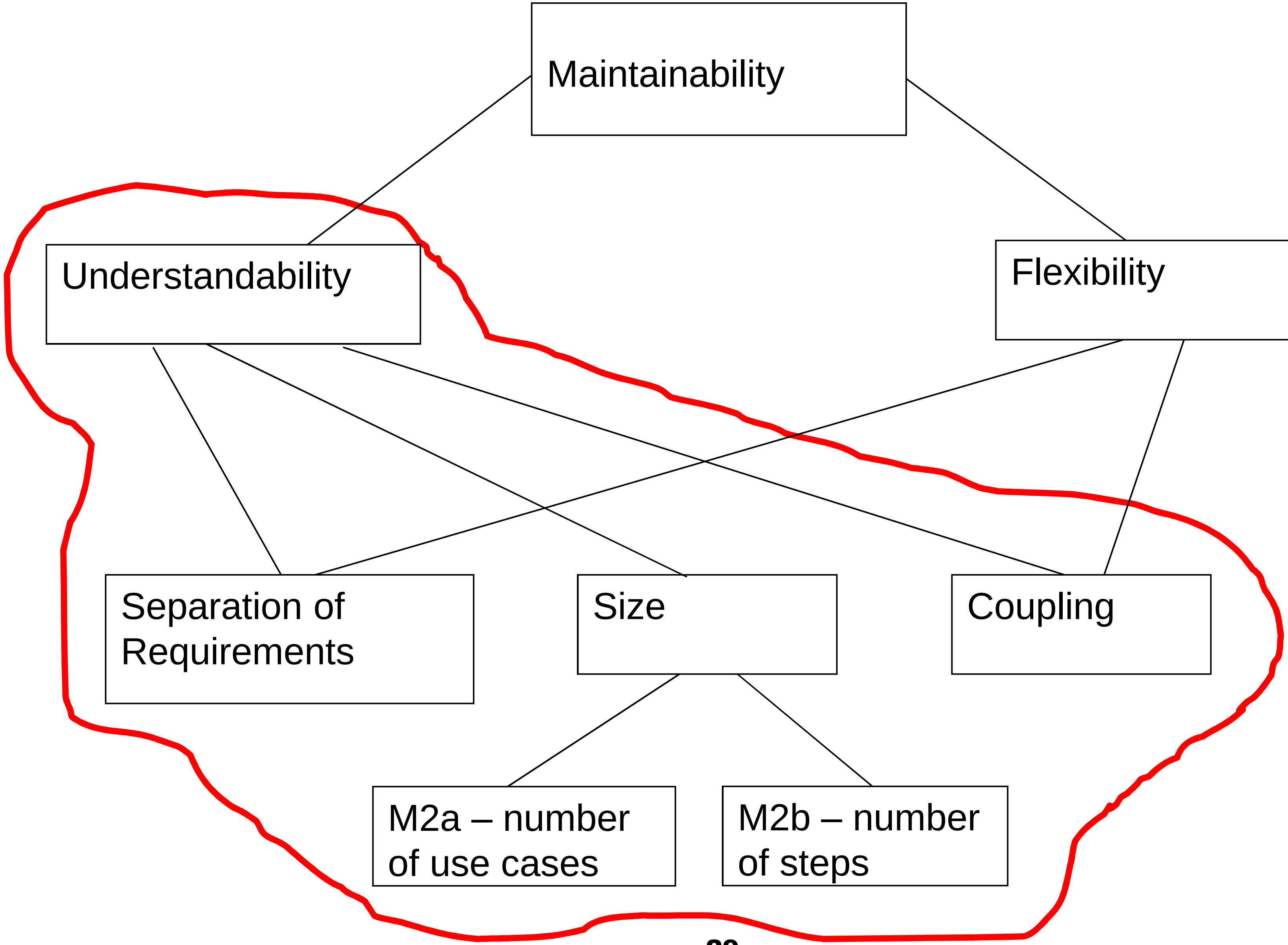
Metric

Metric

Metric

[IEEE 1061, 1998]

An Example of Quality Model



Definition of Questions

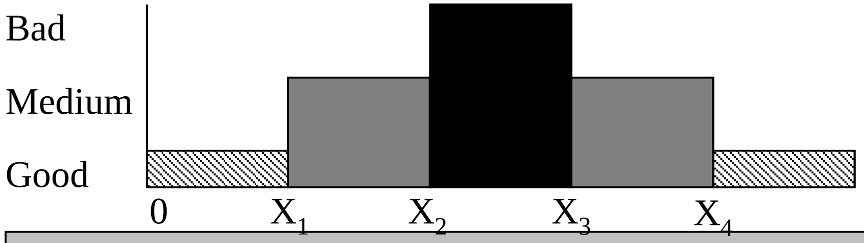
Quality attribute	Question	Source of the Answer
<name of the attribute or sub attribute >	*<question(s) that when answered will provide the insights necessary to achieve the goal. The questions will be answered basically by the words: Good, Medium or Bad>*	*<the sources (metrics or other questions) that need be achieve to answer the question>*
	Template: **How good is the <*quality attribute*> from the <*requirement in focus*>?**	
Understandability	How good is the understandability from the display requirement?	Q1.1. How good is the size from the display requirement? Q1.2. How good is the separation of requirements from the display requirement? Q1.3 - How good is the coupling from the display requirement?

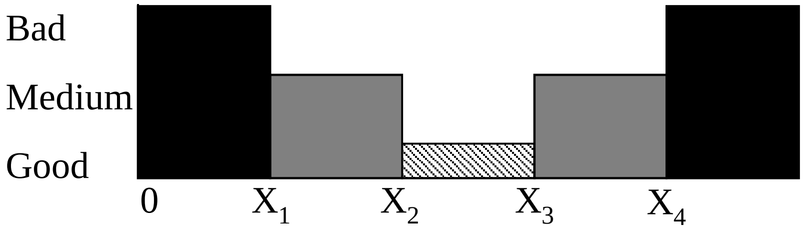
Selection of Metrics

Questions	Metrics	Details
<question that are related with the metric>	*<description of the metric>*	*<details about: the required value, how to obtain the value, among others>*
Q1.1 - How good is the size from the display requirement?	M2a – How many use cases are required to specify the display requirement?	Count the number of use cases where there is, at least, one step that contributes to the specification of display requirements.
	M2b – How many steps are required to specify the display requirement?	Count the total numbers of steps that describe the display requirement.

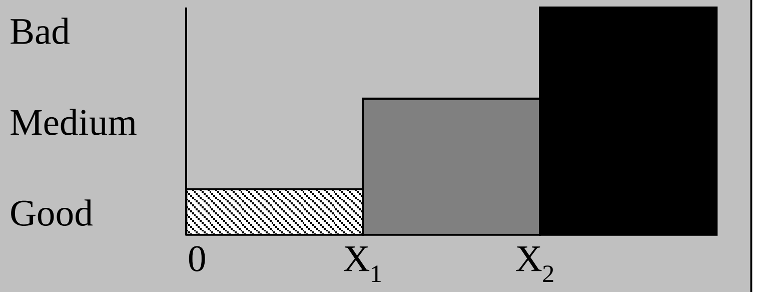
Create a Premisse

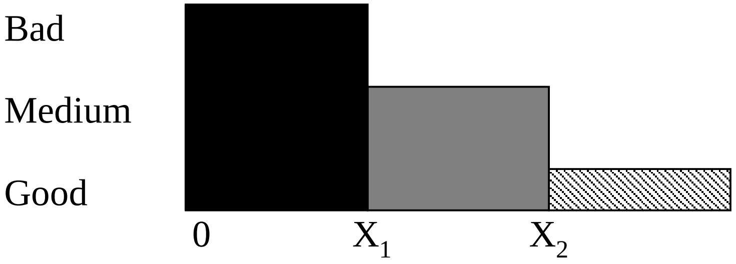
Scales for transformation of numerical values





Template:
The value of <*metric/function*> is "good" if its value is in the range [0, x1], is "medium" if its value is in the range [x1, x2] and is "bad" if its value is in the range [x2, ∞].





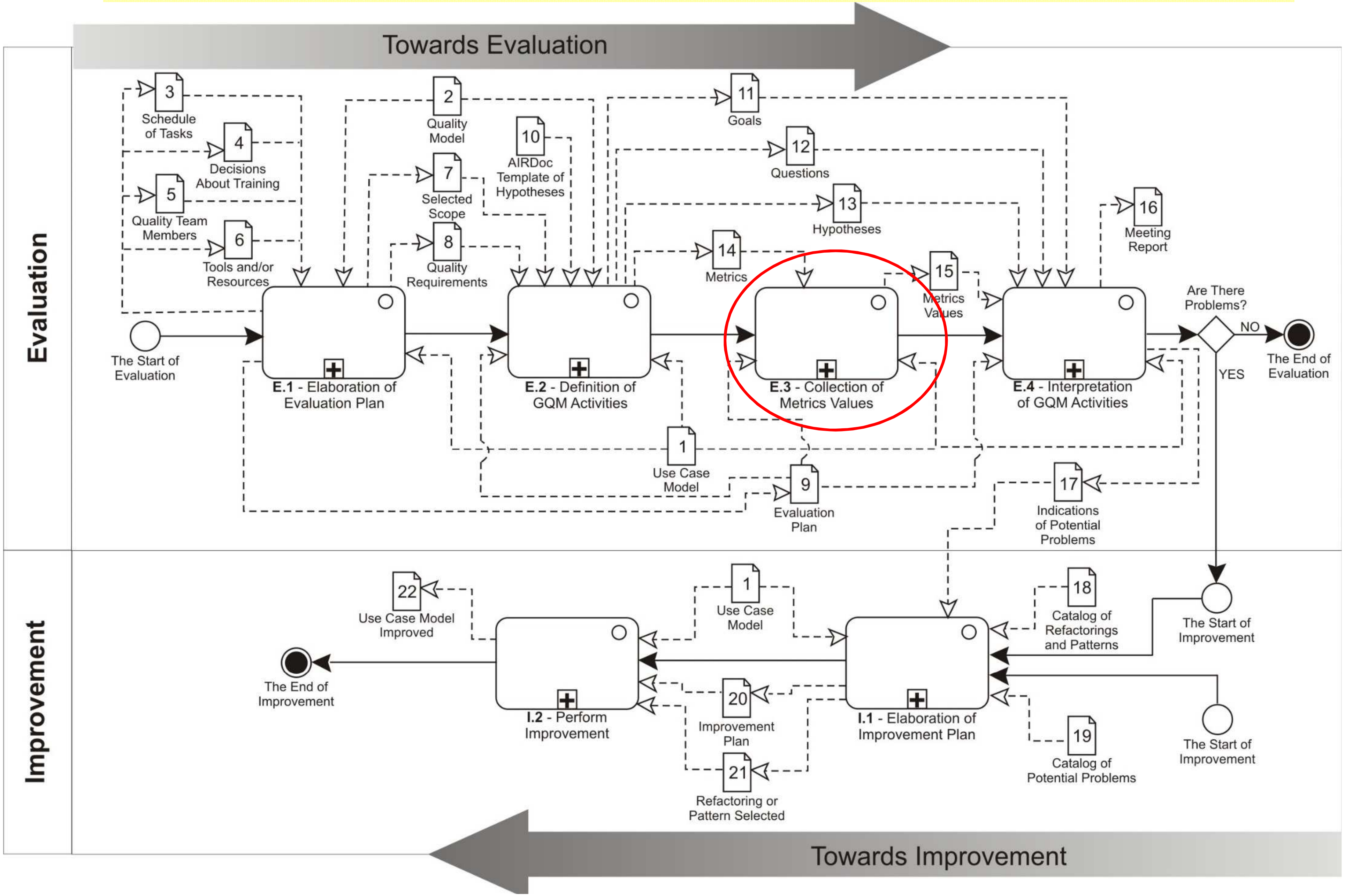
Create a Premisse

Metric	Possible values	Premise
<name or some other identification of the metric>	*<range of possible values>*	*<create a premise analyzing the range of possible values and transform it in a scale of 3 values: Good, Medium and Bad>*
M2a – How many use cases are required to specify the display requirement?	Function2 = M2b/M2a M2a [1 - 50] M2b [1 - 800] Type of Scale: **Increasing** [1 - 35] – Good [36 - 65] – Medium [66 - 800] – Bad	The value of the Function2 is "*good*" if its value is in the range [1, 35], is "*medium*" if its value is in the range [36, 65] and is "*bad*" if its value is in the range [66, ∞].
M2b – How many steps are required to specify the display requirement?		

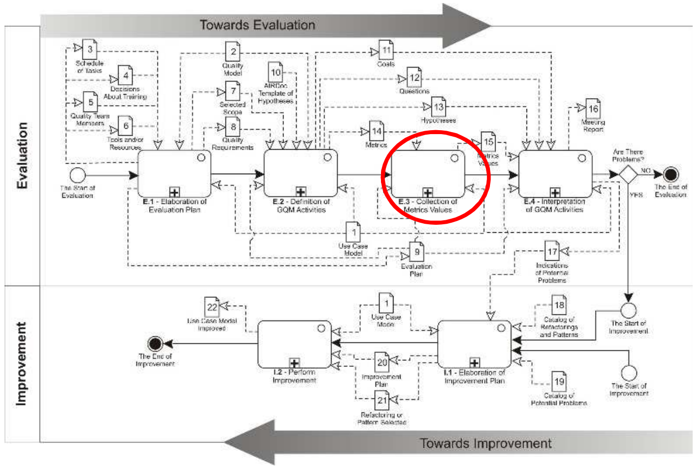
Elaborate Hypothesis to Each Question

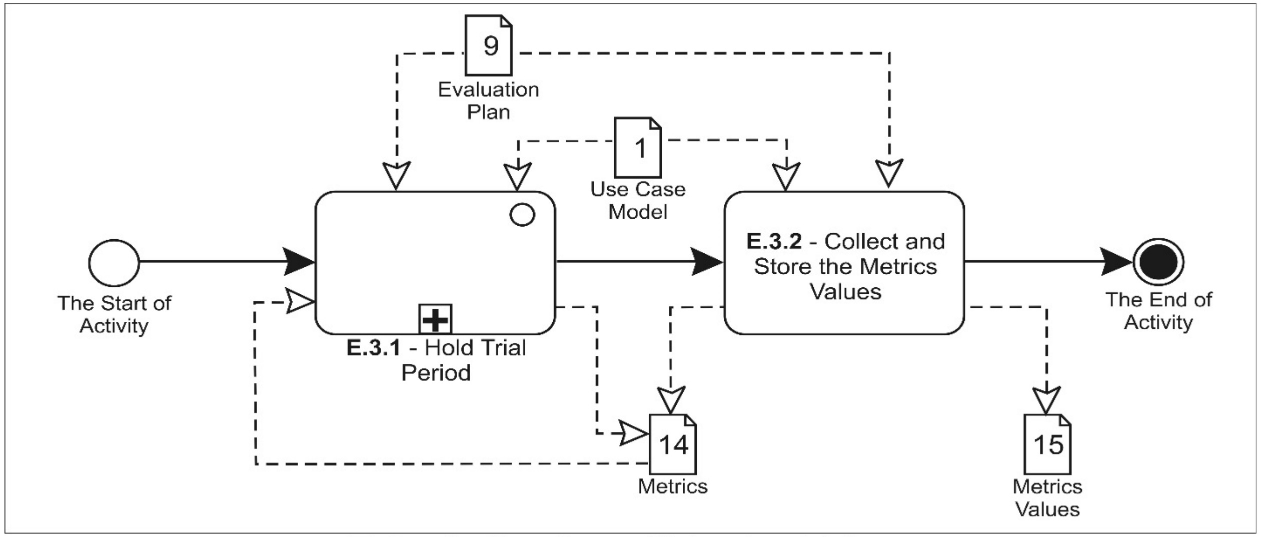
Question	***<question>***
Premise	*<premise (created in step E.2.4.1)>*
Function	*<if some premise is based on a function, it should be described here>*
Hypothesis	The *<quality attribute/sub attribute>* from the *<requirement_in_focus>* is *<Good/Medium/Bad>*. Because the value of the *<metric/function>* is *<equal/lower/higher>* to/than *<metric/function value> (and <equal/lower/higher> than <metric/function value>)* Note: At least three hypotheses must be elaborated, each one to each value "Good, Medium and Bad"
Note	*<if necessary insert some note about the hypothesis>*
Question	**Q1.1 - How good is the size from the display requirement?**
Premise	The value of the Function2 is "*good*" if its value is in the range [1, 35], is "*medium*" if its value is in the range [36, 65] and is "*bad*" if its value is in the range [66, ∞].
Function	Function2 = M2b/M2a
Hypothesis	**H1.1a The size from the display requirement is** *Good.* Because the value of the Function2 is *lower* than *35.* **H1.1b The size from the display requirement is** *Medium.* Because the value of the Function2 is higher than *36* and lower than *65.* **H1.1c The size from the display requirement is** *Bad.* Because the value of the Function2 is *higher* than *66.*
Note	None

The AIRDoc


Towards Evaluation
Evaluation
3
Schedule of Tasks
4
Decisions About Training
5
Quality Team Members
6
Tools and/or Resources
2
Quality Model
7
Selected Scope
8
Quality Requirements
10
AIRDoc Template of Hypotheses
11
Goals
12
Questions
13
Hypotheses
14
Metrics
15
Metrics Values
16
Meeting Report
The Start of Evaluation
E.1 - Elaboration of Evaluation Plan
E.2 - Definition of GQM Activities
E.3 - Collection of Metrics Values
E.4 - Interpretation of GQM Activities
Are There Problems?
NO
YES
The End of Evaluation
1
Use Case Model
9
Evaluation Plan
17
Indications of Potential Problems
Improvement
22
Use Case Model Improved
1
Use Case Model
18
Catalog of Refactorings and Patterns
The Start of Improvement
The End of Improvement
I.2 - Perform Improvement
I.1 - Elaboration of Improvement Plan
20
Improvement Plan
21
Refactoring or Pattern Selected
19
Catalog of Potential Problems
The Start of Improvement
Towards Improvement

Collection of the Metrics Values



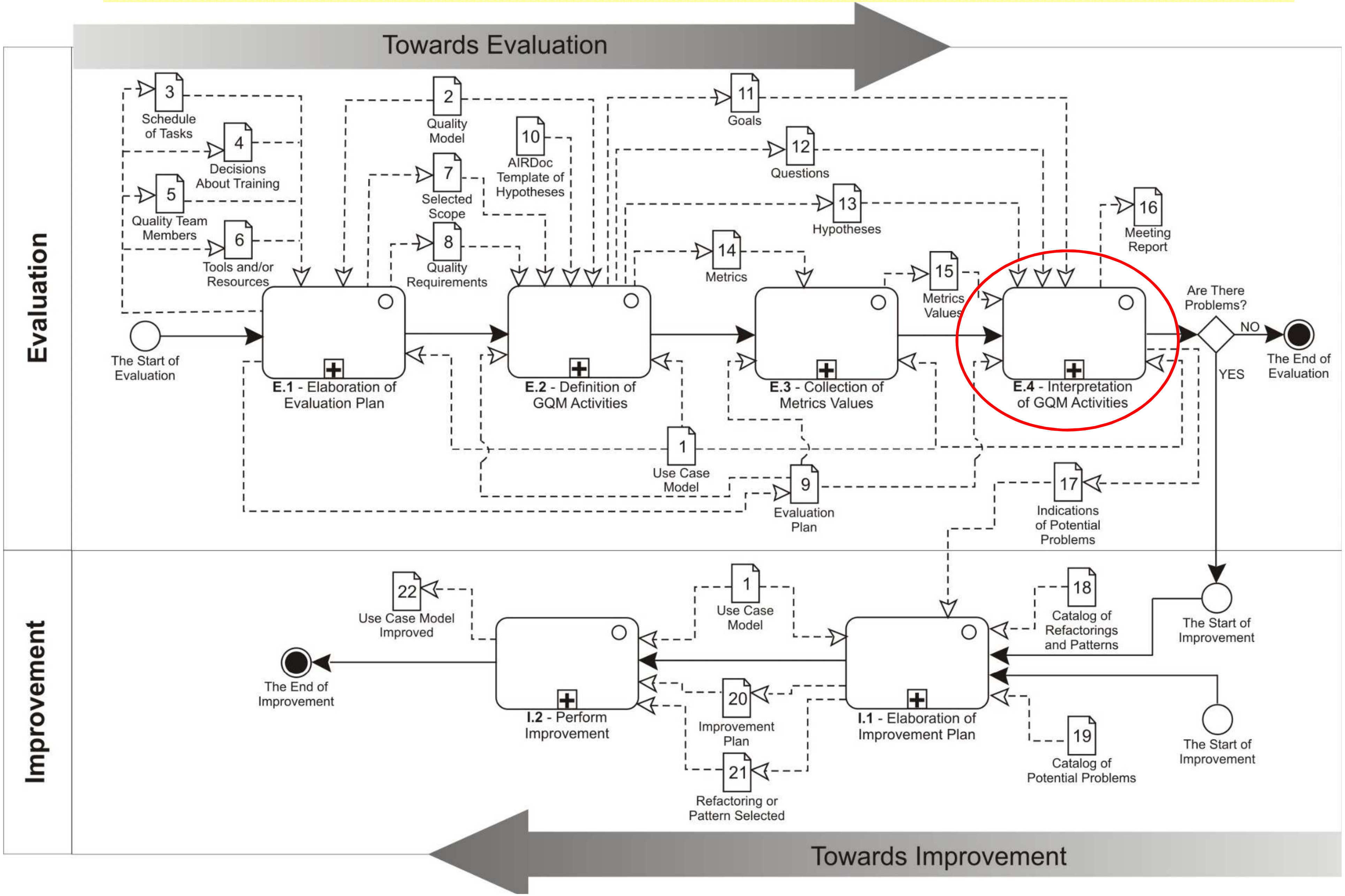


E.3 - Collection of Metrics Values

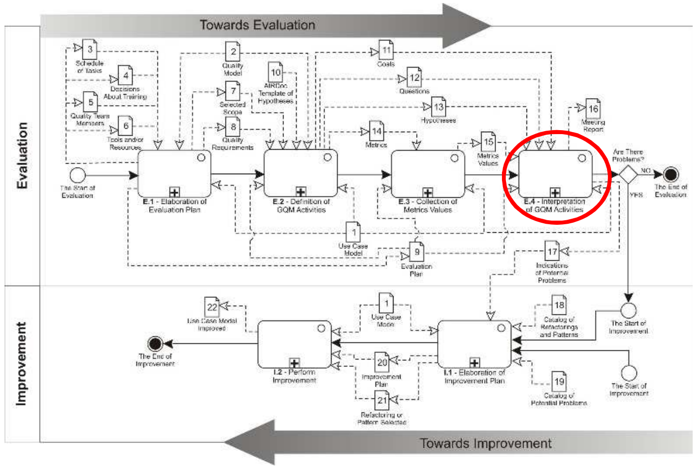
Collect and Store the Metrics Values

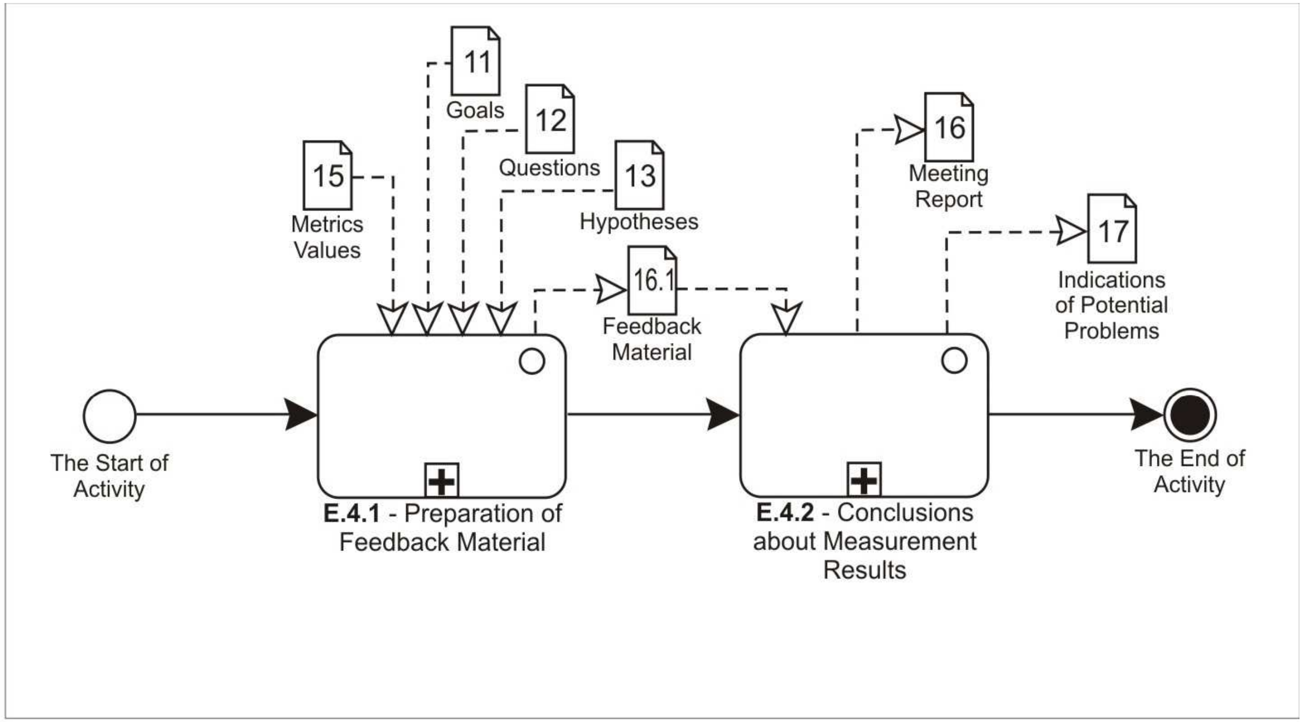
Metric	Value
<metric>	*<numerical value obtained by direct measurement>*
M2a – How many use cases are required to specify the display requirement?	2
M2b – How many steps are required to specify the display requirement?	798

The AIRDoc


Towards Evaluation
Evaluation
Improvement
3
Schedule of Tasks
4
Decisions About Training
5
Quality Team Members
6
Tools and/or Resources
2
Quality Model
7
Selected Scope
8
Quality Requirements
10
AIRDoc Template of Hypotheses
11
Goals
12
Questions
13
Hypotheses
14
Metrics
15
Metrics Values
16
Meeting Report
The Start of Evaluation
E.1 - Elaboration of Evaluation Plan
E.2 - Definition of GQM Activities
E.3 - Collection of Metrics Values
E.4 - Interpretation of GQM Activities
Are There Problems?
NO
YES
The End of Evaluation
1
Use Case Model
9
Evaluation Plan
17
Indications of Potential Problems
22
Use Case Model Improved
1
Use Case Model
18
Catalog of Refactorings and Patterns
The End of Improvement
I.2 - Perform Improvement
I.1 - Elaboration of Improvement Plan
The Start of Improvement
20
Improvement Plan
21
Refactoring or Pattern Selected
19
Catalog of Potential Problems
The Start of Improvement
Towards Improvement

Interpretation of GQM Activities





E.4 - Interpretation of GQM Activities

Accept the hypothesis and Answer the Questions

Question	Answer	Note
<question>	*<hypothesis accepted>*	*<some note about the question or the answer>*
Q1.1 - How good is the size from the display requirement?	**H1.1c The size from the display requirement is Bad. Because the value of the Function2 is higher than 66.**	The value obtained in the Function2 = M2b/M2a is 798/2 = 399.

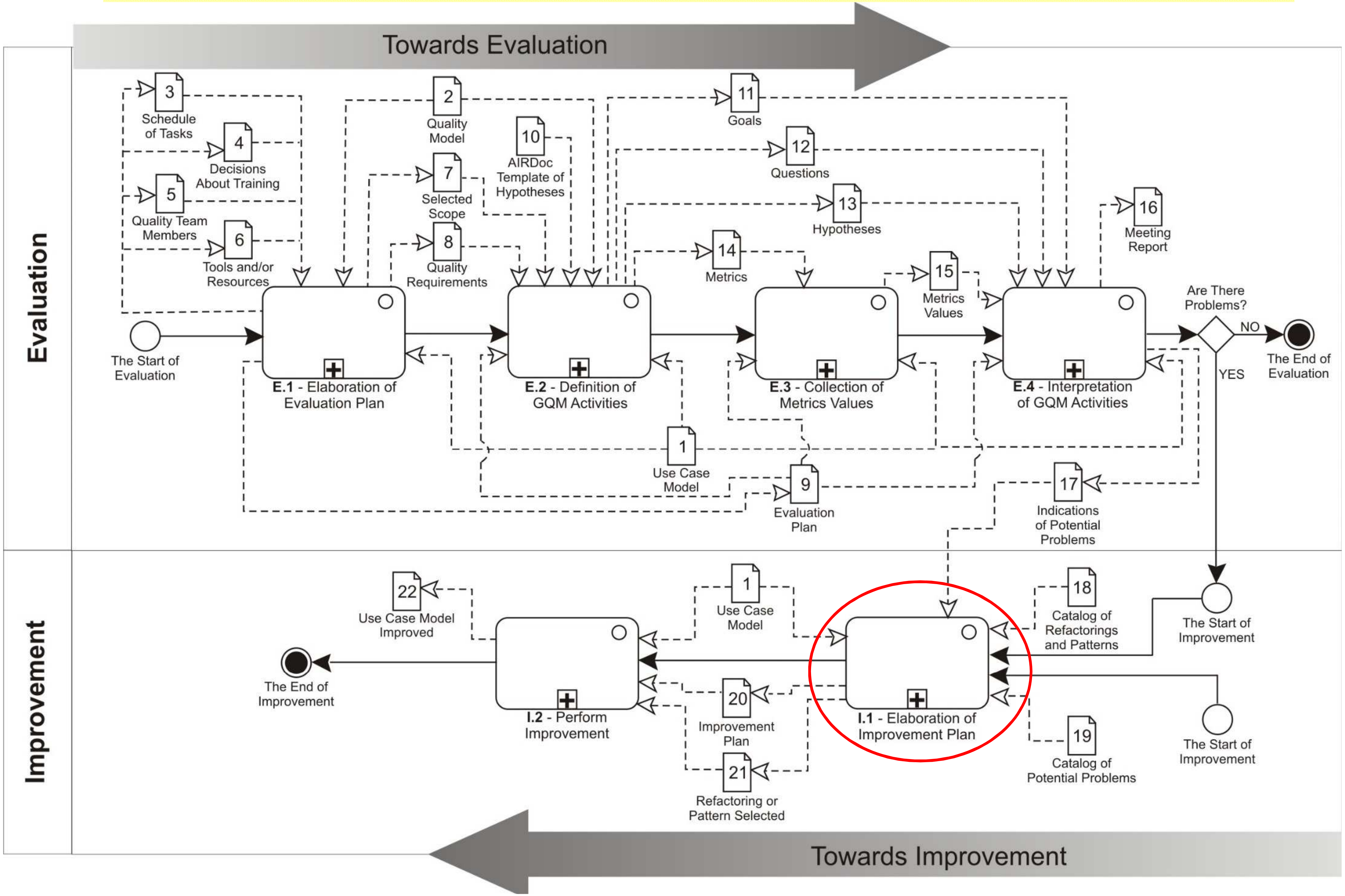
Analyze and Interpret the Hypotheses Questions and Goals

Question	Answer in analysis	Note
<question>	*<hypotheses accepted with represent a Bad or Medium values and/or hypotheses rejected with represent a Good value>*	*<some note about the analysis>*
Q1.1 - How good is the size from the display requirement?	**Accepted -> H1.1c The size from the display requirement is Bad. Because the value of the Function2 is higher than 66.**	The two use cases that describe the “display requirement” contain a lot of steps.

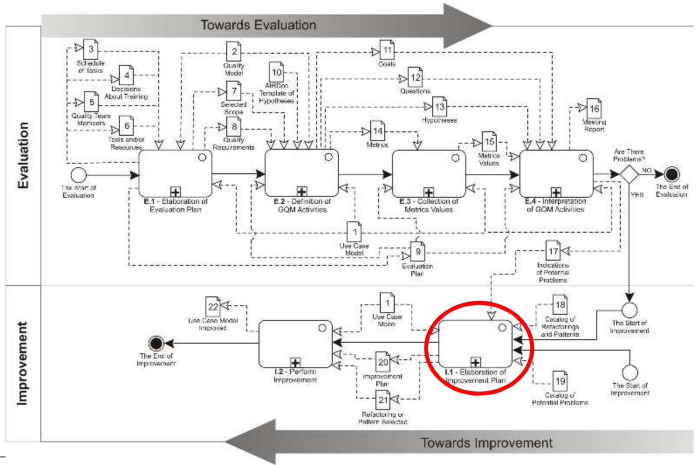
Create a Document to Indicate where the Worst Results were Found

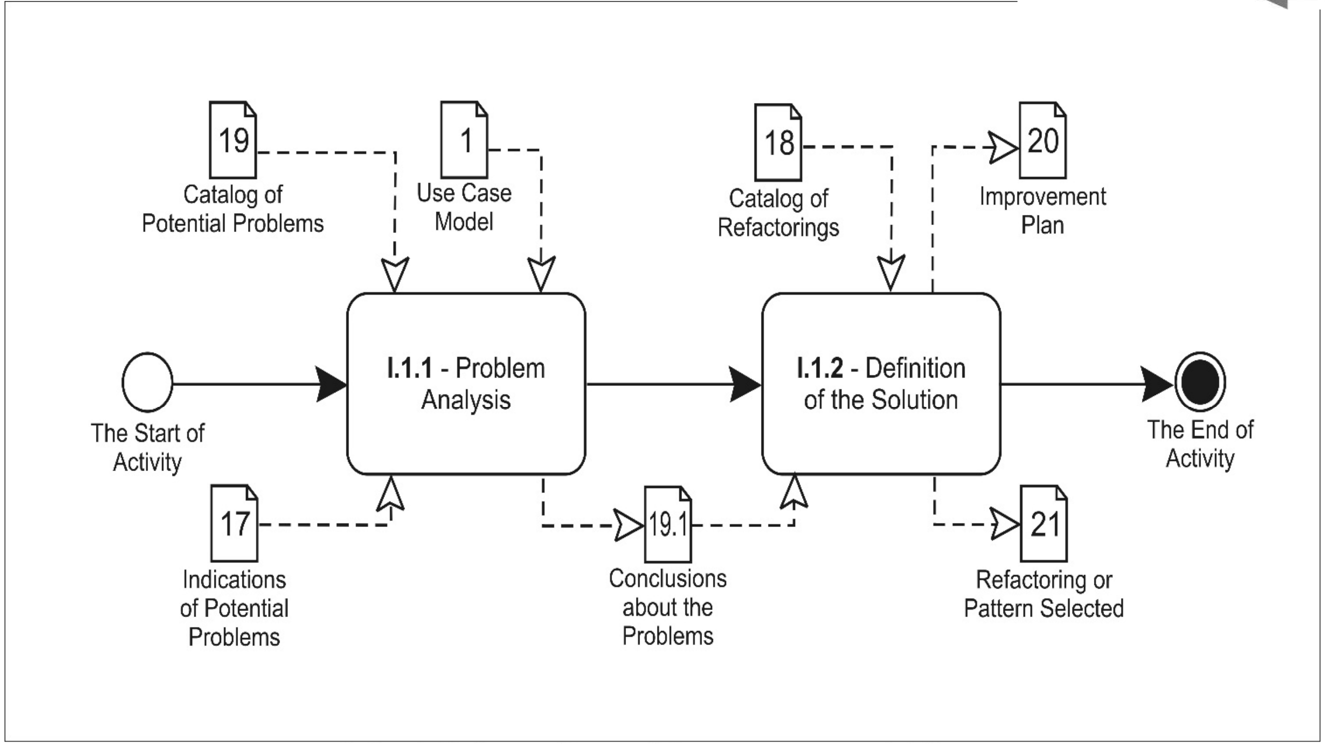
Potential Problem	Localization
<indicate the type of the problem>	*<indicate the name of use case(s) and, if necessary, the specific step(s)>*
The two use cases that describe the “display requirement” contain a lot of steps.	Use case 1 – “Display spreadsheet control”. Use case 2 – “Display screen of user analysis”. Note: All steps of both use cases describe the “display requirement”

The AIRDoc


Towards Evaluation
Evaluation
3
Schedule of Tasks
4
Decisions About Training
5
Quality Team Members
6
Tools and/or Resources
2
Quality Model
7
Selected Scope
8
Quality Requirements
10
AIRDoc Template of Hypotheses
11
Goals
12
Questions
13
Hypotheses
14
Metrics
15
Metrics Values
16
Meeting Report
The Start of Evaluation
E.1 - Elaboration of Evaluation Plan
E.2 - Definition of GQM Activities
E.3 - Collection of Metrics Values
E.4 - Interpretation of GQM Activities
Are There Problems?
NO
YES
The End of Evaluation
1
Use Case Model
9
Evaluation Plan
17
Indications of Potential Problems
Improvement
22
Use Case Model Improved
1
Use Case Model
18
Catalog of Refactorings and Patterns
The Start of Improvement
The End of Improvement
I.2 - Perform Improvement
I.1 - Elaboration of Improvement Plan
20
Improvement Plan
21
Refactoring or Pattern Selected
19
Catalog of Potential Problems
The Start of Improvement
Towards Improvement

Elaboration of Improvement Plan




19
Catalog of Potential Problems
1
Use Case Model
18
Catalog of Refactorings
20
Improvement Plan
The Start of Activity
I.1.1 - Problem Analysis
I.1.2 - Definition of the Solution
The End of Activity
17
Indications of Potential Problems
19.1
Conclusions about the Problems
21
Refactoring or Pattern Selected

I.1 - Elaboration of Improvement Plan

Large Use Case Problem

Context

Large Use Case occurs when (i) a use case is trying to handle several different requirements at the same time or (ii) there are many alternative flows and steps.

This problem is particularly significant when the maximum size of each use case has already been set by the organization’s Software Quality Assurance Team.

Possible Solutions

Use the *Extract Use Case* refactoring [Ramos et al., 2007c] to extract information related to a given concern and insert it into a new use case. This operation could be repeated for each major concern addressed by this large use case. This solution needs to be analyzed with caution, because it may increase the number of the use cases. To solve the problem of the increase of the use cases number, the *Package Use Cases* refactoring could be applied.

If the flows or other components of a use case could be moved to another use case, the *Move Activity* refactoring [Ramos et al., 2007c] could be used.

After extracting or relocating requirements, we sometimes need to rename the use case to better express the intention of the newly created one or of the one that was modified. In this case, the *Rename Use Case* refactoring [Ramos et al., 2007c] could be used to provide more appropriated names.

This refactoring opportunity is particularly important when there is a limit on the size of each use case, set by the organization’s Software Quality Assurance Team.

Another possible solution is to use the *Extract Early Aspectual Use Case* refactoring [Ramos et al., 2008a]. This solution employs aspect-oriented requirements engineering and may be a favorable option if the requirements engineer desires to work with Aspect-Oriented Development of Software.

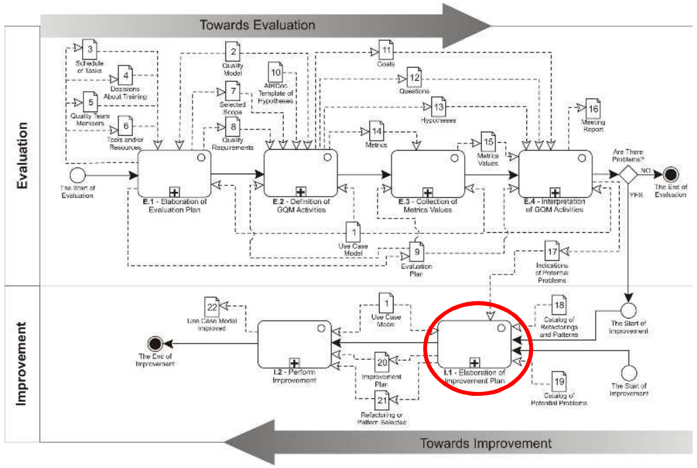
Problem Analysis

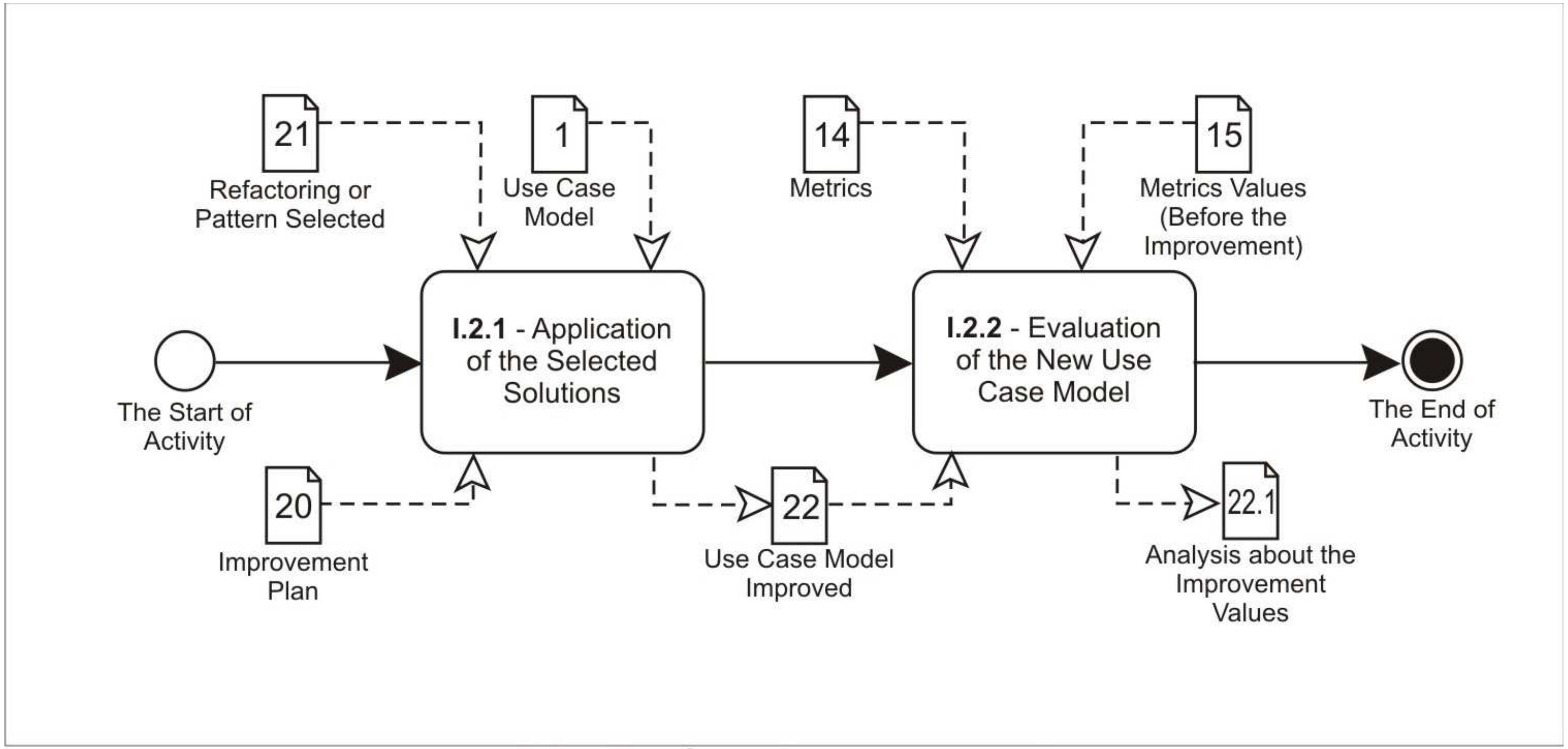
The Potential Problem	Selected Solution	Analysis of Cost and benefit
<name of the potential problem in agreement the catalog of Potential Problems >	*<list of the refactorings to solve the potential problems>*	*<describe the possible cost and benefits envisage with the application of the refactorings>*
Large Use Case	Extract Use Case Package Use Case	The selected solution will have the cost of rearrange the use cases that describe the display requirement with the intention of decrease the size of it. We infer that this rearrangement will benefit the maintainability of this requirement.

Extract Use Case

Context	A set of inter-related information is used in several places or could be better modularized in a separate use case. Alternatively a use case description is too large or contains information related to a concern that is scattered across several use cases or is tangled with other concerns.
Solution	Extract the information to a new use case and name it according to the context.
Motivation	This refactoring should be applied when there are large use cases descriptions that can be split into two or more new use case(s). These large use cases include a great deal of information that is difficult to understand. Furthermore, it is not easy to locate the needed information quickly [Alexander and Stevens 2002], [Sommerville, 1997].

Perform Improvement

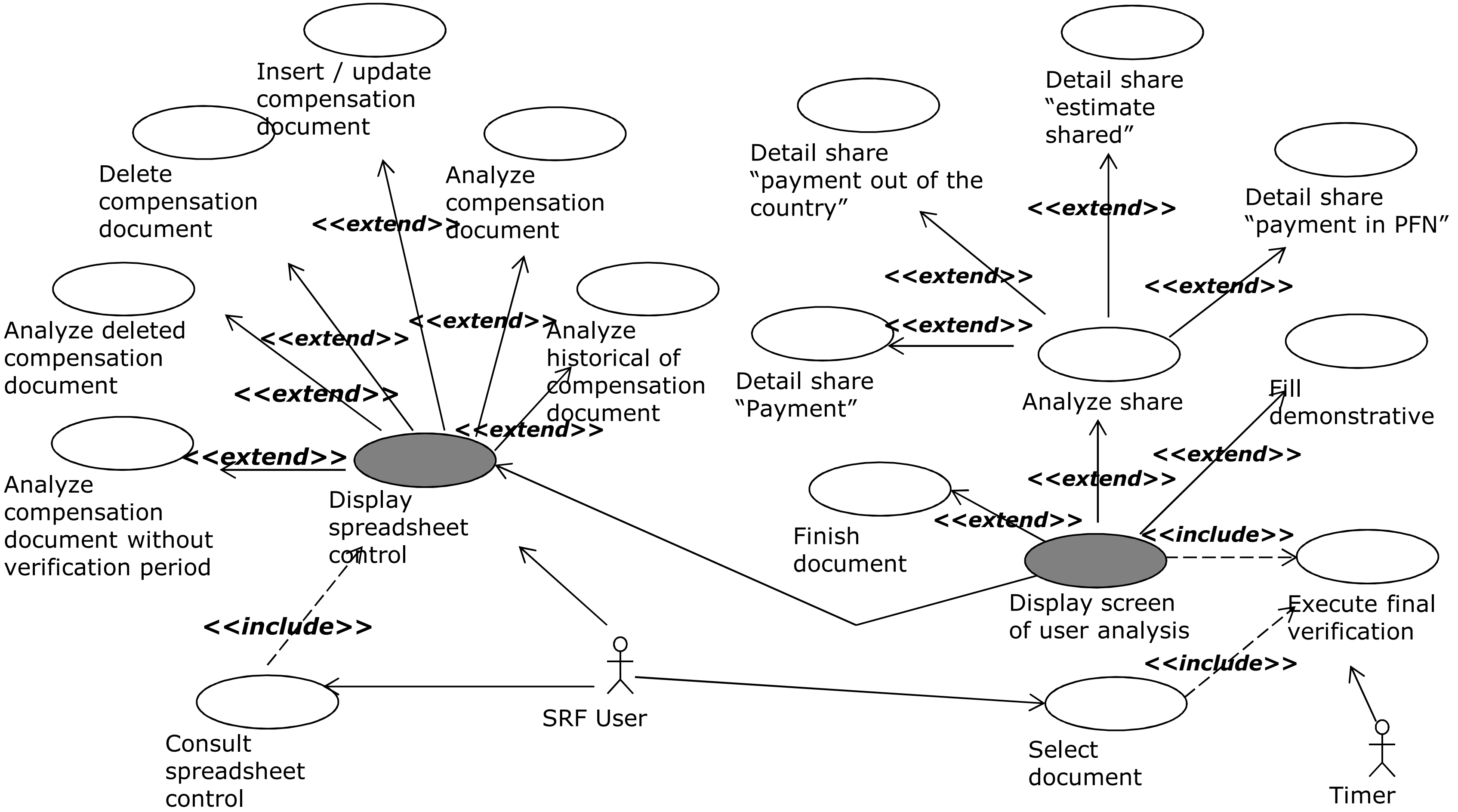



21
Refactoring or Pattern Selected
1
Use Case Model
14
Metrics
15
Metrics Values (Before the Improvement)
The Start of Activity
I.2.1 - Application of the Selected Solutions
I.2.2 - Evaluation of the New Use Case Model
The End of Activity
20
Improvement Plan
22
Use Case Model Improved
22.1
Analysis about the Improvement Values

I.2 - Perform Improvement

Use Case Model After the Improvement

I) Apply the solutions selected



Insert / update compensation document
Delete compensation document
Analyze compensation document
Analyze deleted compensation document
Analyze historical of compensation document
Analyze compensation document without verification period
Display spreadsheet control
Consult spreadsheet control
SRF User
<<extend>>
<<include>>
Detail share "estimate shared"
Detail share "payment out of the country"
Detail share "payment in PFN"
Detail share "Payment"
Analyze share
Fill demonstrative
Finish document
Display screen of user analysis
Execute final verification
Select document
Timer

The Extract Use Case Refactoring was applied

Contributions

- We proposed a process to perform the evaluation and improvement in Use Case models.
- This process is based on GQM [Basili et al.,1994] and complies with the IEEE Standard for a Software Quality Methodology [IEEE 1061, 1998] and with the IEEE Recommended Practice for Software Requirements Specifications [IEEE 830, 1998].

Contributions

- The AIRDoc process includes a catalog of known problems which may help to better categorize the potential problems. It also provides a refactorings catalog which to can assist the user to improve the use case model quality.

Catalog of Potential Problema

- Currently there are 11 potential problems;
- For each potential problem we describe:
  - □ (a) a context to identify occurrences of the problem and,
  - □ (b) the refactorings that can be used to solve the effects of the problem occurrences.

Duplicated Requirement

Large Use Case

Complex Conditional Structures

Lazy Use Case

Naming Problems

Tangled Requirements

Scattered Requirements

Large Use Case Model

Inconsistent Requirement

Ambiguous Activity

Lack of Rank

The Catalog of Solutions for Improvement

- We propose a collection of requirements refactorings which are described in the format recommended by [Fowler et al., 2000];
- We describe 8 different refactorings;

Extract Use Case

Rename Use Case

Move Activity

Inline Use Case

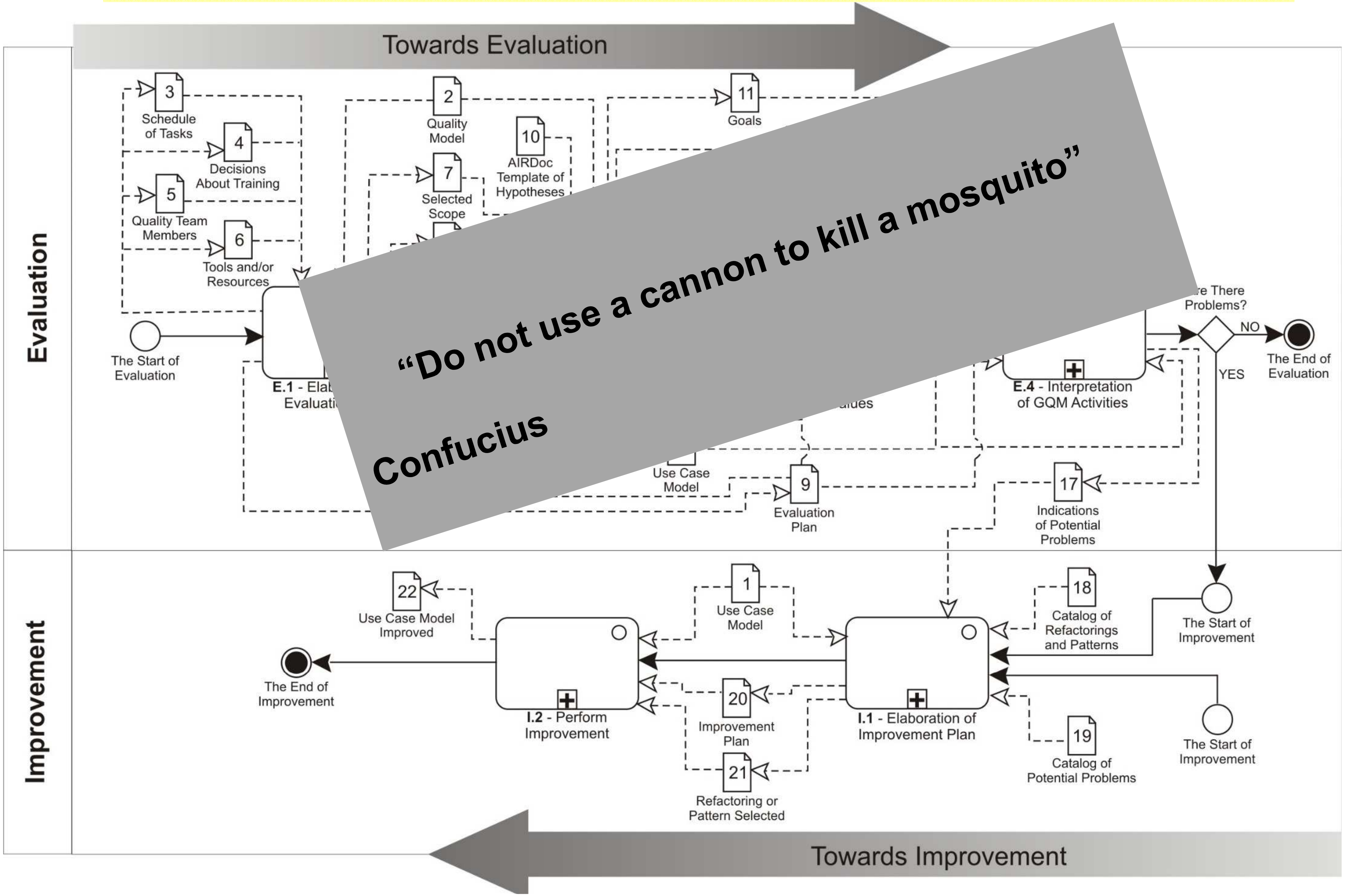
Extract Alternative Flows

Extract Early Aspectual Use Case

Use Cases Package

Rank Use Case

After 6 years What I learn about The AIRDoc



“Do not use a cannon to kill a mosquito”

Confucius

After 6 years - 3 master's work

- AIRDoc-i* (The i* framework proposes an agent-oriented approach to requirements engineering centering on the intentional characteristics of the agent.) http://www.cs.toronto.edu/km/istar/
- AIRDoc-BPM (work on progress)
- AIRDoc -> QUALISIS-Br (Health Information Systems)

How to Assess the Quality of a Requirements Specification?

"work on progress"

A Systematic Literature Review

Context

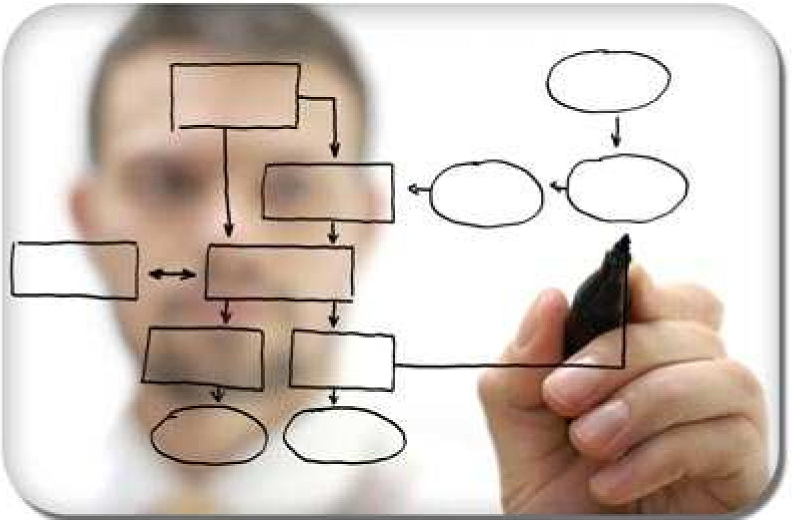
Ensuring a good quality in a requirements specification means that we will produce a quality software.

Context

Works have been generated by recommendations, such as: how to write a requirements specification, what we should to do and we should not to do.

Context

Researchers developed methods and technics for the software engineer to assess the quality of requirements specification.



The Goal

Is ensuring quality by assessing the requirements specification a guarantee of success in software development?

A quality evaluation in the requirements specification will predict how good will be the software project success. Who shows evidence to support this?

The Goal

This work aims is looking for whom answered this questions.

To do it possible, We are doing a systematic review of the literature

Main Contributions

- Updating researchers and practitioners on the trends of the searched area.
- Identifying possible gaps and research opportunities.
- Indicating ways to be followed by those who desire to improve a requirements specification.

Phase 1: Plan Review

- 1.1. Specify Research Questions
- 1.2. Develop Review Protocol
- 1.3. Validate Review Protocol

Phase 2: Conduct Review

- 2.1. Identify Relevant Research
- 2.2. Select Primary Studies
- 2.3. Asses Study Quality

Applying Exclusion Criterias

- 2.4. Create a List of Valid Papers
- 2.5. Extract Required Data

Answering the questions

- 2.6. Synthesise Data

Phase 3: Document Review

- 3.1. Write Review Report
- 3.2. Validade Report

Specify Research Questions

- **RQ - 01** - What are the effectives methods of assessing the quality of a requirements specification?
  - □ Effective = successful in producing a desired or intended result.
  - □ Context = ensure that the software developed inherit the quality from the Requirement Specification.

RQ - 01 - What are the effectives methods of assessing the quality of a requirements specification?

- To answer this question, we will search for papers that describe methods or techniques to assess the requirement quality. The papers found need to report how it was experimented to prove or give some evidence that the method/technique are effective.

We need to define where to search for papers (www.scopus.com), the inclusion criteria and exclusion criteria.

Tool

- We use SCOPUS tool to search for relevant papers.
  - SCOPUS indexes IEEE, ACM, Elsevier publications, main workshops and conferences;
  - For software engineering researchers this means it indexes many of the leading publications

www.scopus.com

Inclusion Criteria

- Key words to extract the papers:
  - □ “requirements specification” and measure;
  - □ “requirements specification” and inspection;
  - □ “requirements specification” and evaluation;
  - □ "requirements specification" and evaluate;
  - □ "requirements specification" and metric;
  - □ "requirements document" and measure;
  - □ "requirements document" and inspection;
  - □ "requirements document" and evaluation;
  - □ "requirements document" and evaluate;
  - □ "requirements document" and metric;

Parameters of Search in SCOPUS
Where: in Article Title, Abstract and key words
Document type: Article or conference paper
Published: 1974 to 2014
Subject Area: Computer Science

Executing the Search Strings at SCOPUS Tool

- Example:

  ```
  (TITLE-ABS-KEY("requirements specification") AND TITLE-ABS-KEY(Measure))
  AND DOCTYPE(ar OR cp) AND SUBJAREA(COMP) AND PUBYEAR > 1973 AND PUBYEAR <
  2015
  ```

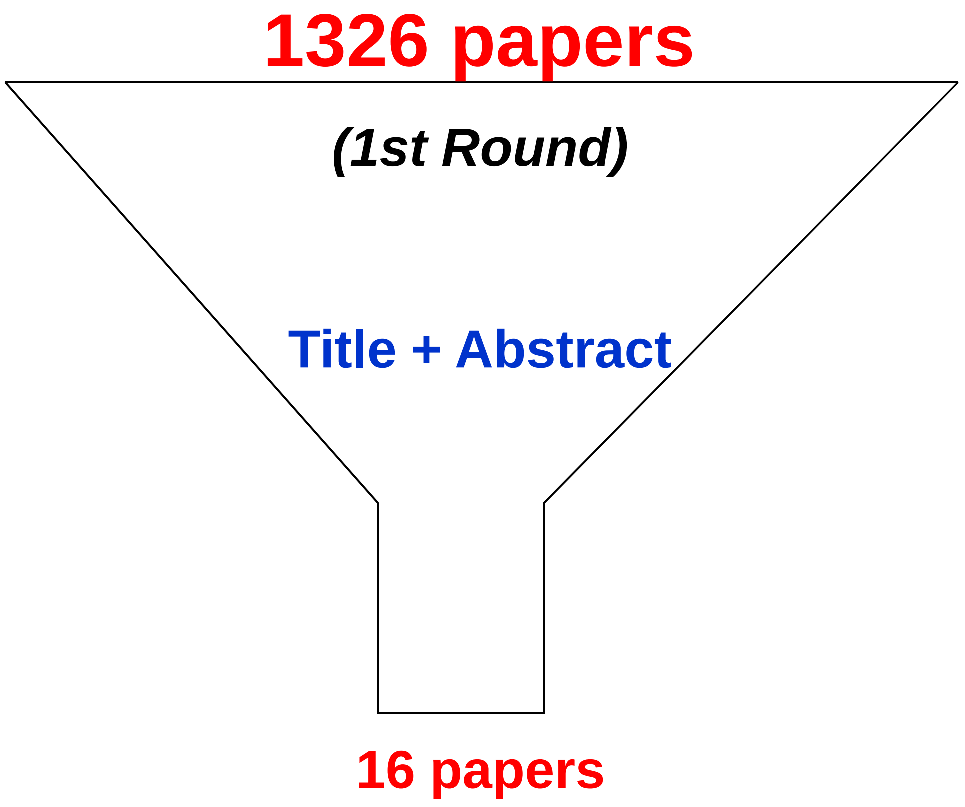
- We found **1326** results:
  - “requirements specification” and measure (95 RESULTS)
  - “requirements specification” and inspection (50 RESULTS)
  - “requirements specification” and evaluation (309 RESULTS)
  - "requirements specification" and evaluate (107 RESULTS)
  - "requirements specification" and metric (68 RESUTS)
  - "requirements specification" and quality (423 RESUTS)
  - "requirements document" and measure (20 RESULTS)
  - "requirements document" and inspection (36 RESUTS)
  - "requirements document" and evaluation (71 RESULTS)
  - "requirements document" and evaluate (29 RESULTS)
  - "requirements document" and metric (27 RESULTS)
  - "requirements document" and quality (101 RESULTS)

Applying Exclusion Criterias

- Papers that are based only on expert opinion.
- Short-papers, introductions to special issues, tutorials, and mini-tracks.
- Studies not related to any of the research questions scope.
- Preliminary conference versions of included journal papers.
- Studies not in English, Portuguese or Spanish.
- Studies whose findings are unclear and ambiguous (i.e., results are not supported by any evidence).
- Papers that do not provide any relevant information, as well as repeated measures proposed by more than one author.
- Repeated papers.

Applying Exclusion Criterias

(1st Round)



1326 papers

(1st Round)

Title + Abstract

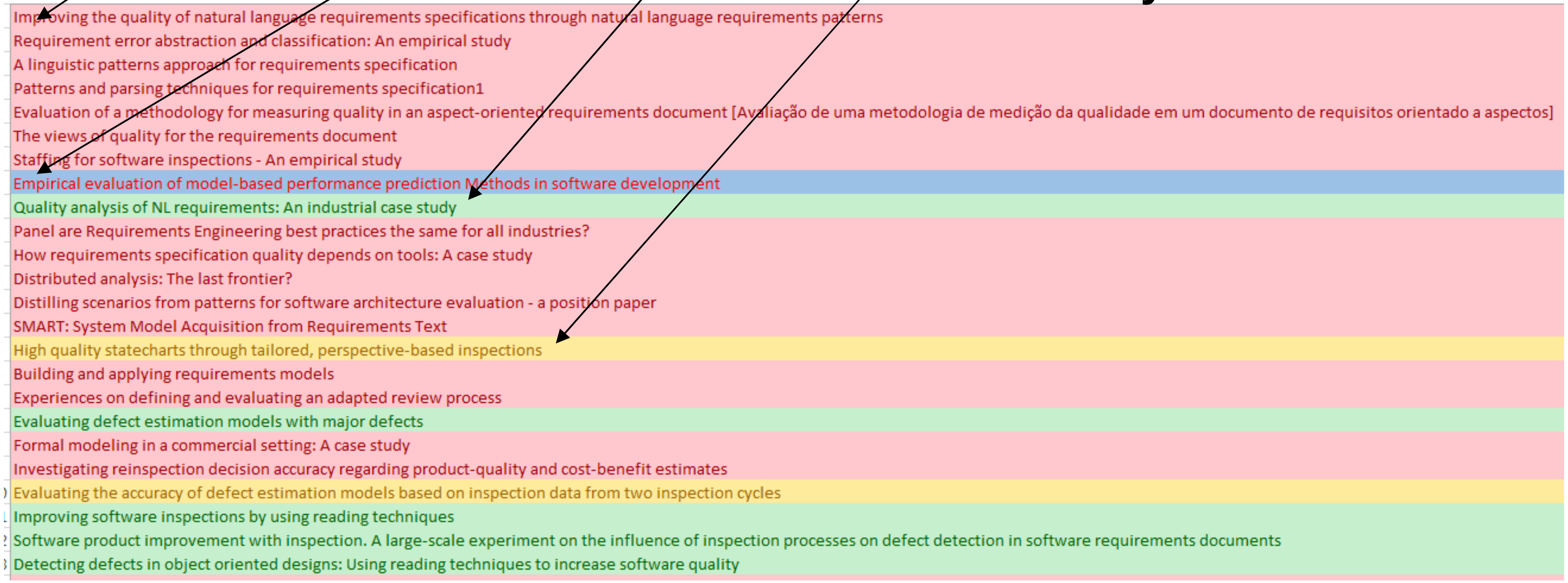
16 papers

Excluded by the title

Included by the title

Included by the abstract

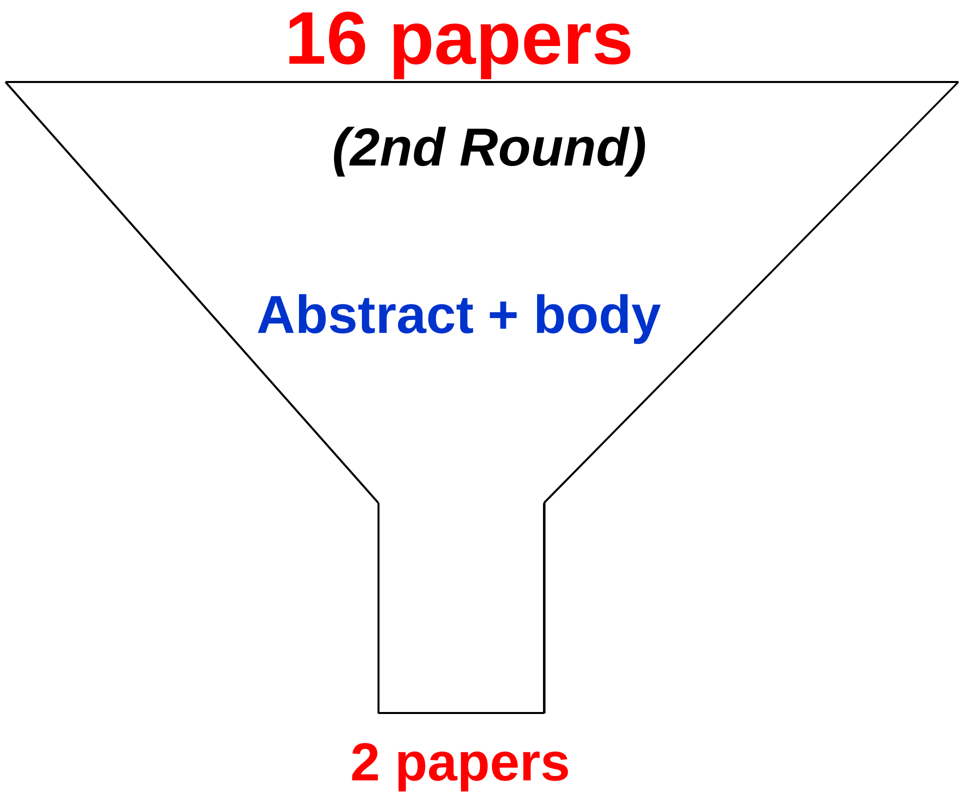
Excluded by the abstract



Improving the quality of natural language requirements specifications through natural language requirements patterns
Requirement error abstraction and classification: An empirical study
A linguistic patterns approach for requirements specification
Patterns and parsing techniques for requirements specification1
Evaluation of a methodology for measuring quality in an aspect-oriented requirements document [Avaliação de uma metodologia de medição da qualidade em um documento de requisitos orientado a aspectos]
The views of quality for the requirements document
Staffing for software inspections - An empirical study
Empirical evaluation of model-based performance prediction Methods in software development
Quality analysis of NL requirements: An industrial case study
Panel are Requirements Engineering best practices the same for all industries?
How requirements specification quality depends on tools: A case study
Distributed analysis: The last frontier?
Distilling scenarios from patterns for software architecture evaluation - a position paper
SMART: System Model Acquisition from Requirements Text
High quality statecharts through tailored, perspective-based inspections
Building and applying requirements models
Experiences on defining and evaluating an adapted review process
Evaluating defect estimation models with major defects
Formal modeling in a commercial setting: A case study
Investigating reinspection decision accuracy regarding product-quality and cost-benefit estimates
Evaluating the accuracy of defect estimation models based on inspection data from two inspection cycles
Improving software inspections by using reading techniques
Software product improvement with inspection. A large-scale experiment on the influence of inspection processes on defect detection in software requirements documents
Detecting defects in object oriented designs: Using reading techniques to increase software quality

Applying Exclusion Criterias

(2nd Round)



[Answer]

RQ - 01 - What are the effectives methods of assessing the quality of a requirements specification?

- Fagan’s inspection [1]
- Requirements Metrics [2]

[1] - Doolan, E. P. "Experience with Fagan's inspection method." Software: Practice and Experience 22.2 (1992): 173-182.

[2] - Knauss, Eric, Christian El Boustani, and Thomas Flohr. "Investigating the impact of software requirements specification quality on project success." Product-Focused Software Process Improvement. Springer Berlin Heidelberg, 2009. 28-42.

	Method	How does it was validated	Result	Considerations by the authors	Research opportunities suggested by the authors
[1]	Fagan’s inspection	A cost-benefit analysis of the defects uncovered by inspecting software requirements specifications according to the method of Fagan	The analysis made indicates that Fagan's inspection is worthwhile.	However, we should not ascribe all the benefits of this process to Fagan’s inspection methodology alone. One very clear message emanating from the emphasis placed by the SSSG on software requirements specifications is that the greater visibility and control afforded by merely getting these requirements down on paper already constitutes an enormous benefit.	Fagan’s inspection is not only applicable to validating software requirements specifications; it can equally well be used to inspect any item (e.g. scope documents, user documentation, design, code, test plan, test results, etc.) produced during the software lifecycle of a project.. Any effort to apply it to other areas- management documents, for example could be very profitable.

[1] - Doolan, E. P. "Experience with Fagan's inspection method." Software: Practice and Experience 22.2 (1992): 173-182.

	Method	How does it was validated	Result	Considerations by the authors	Research opportunities suggested by the authors
[2]	Requirement Metrics : - Grammar; - Rules of Expression; - Ambiguous terms; - Exist. Identifier -Unexplained tech. terms -Contradictoriness Completeness Verifiable goals of req. Correctness Redundancy Feasibility Necessary Contradictoriness (bet. req.) Legally classified Assigned priority Out of date	They formulated hypotheses about how good the quality goals are reached at the moment. Those hypotheses are expected measurements results. After the elicitation of data they are able to verify the hypotheses and determine if they were correct or not. These metrics were applied in roughly 40 student's software projects	The quality of a SRS strongly influences the probability of its project success	Based on our results we found two specific thresholds: A lower threshold: Projects that have a SRS's quality below this value are highly endangered. A higher threshold: Projects that have a SRS's quality above this value are likely to succeed.	To compare our teaching projects to industry projects;

[2] - Knauss, Eric, Christian El Boustani, and Thomas Flohr. "Investigating the impact of software requirements specification quality on project success." Product-Focused Software Process Improvement. Springer Berlin Heidelberg, 2009. 28-42.

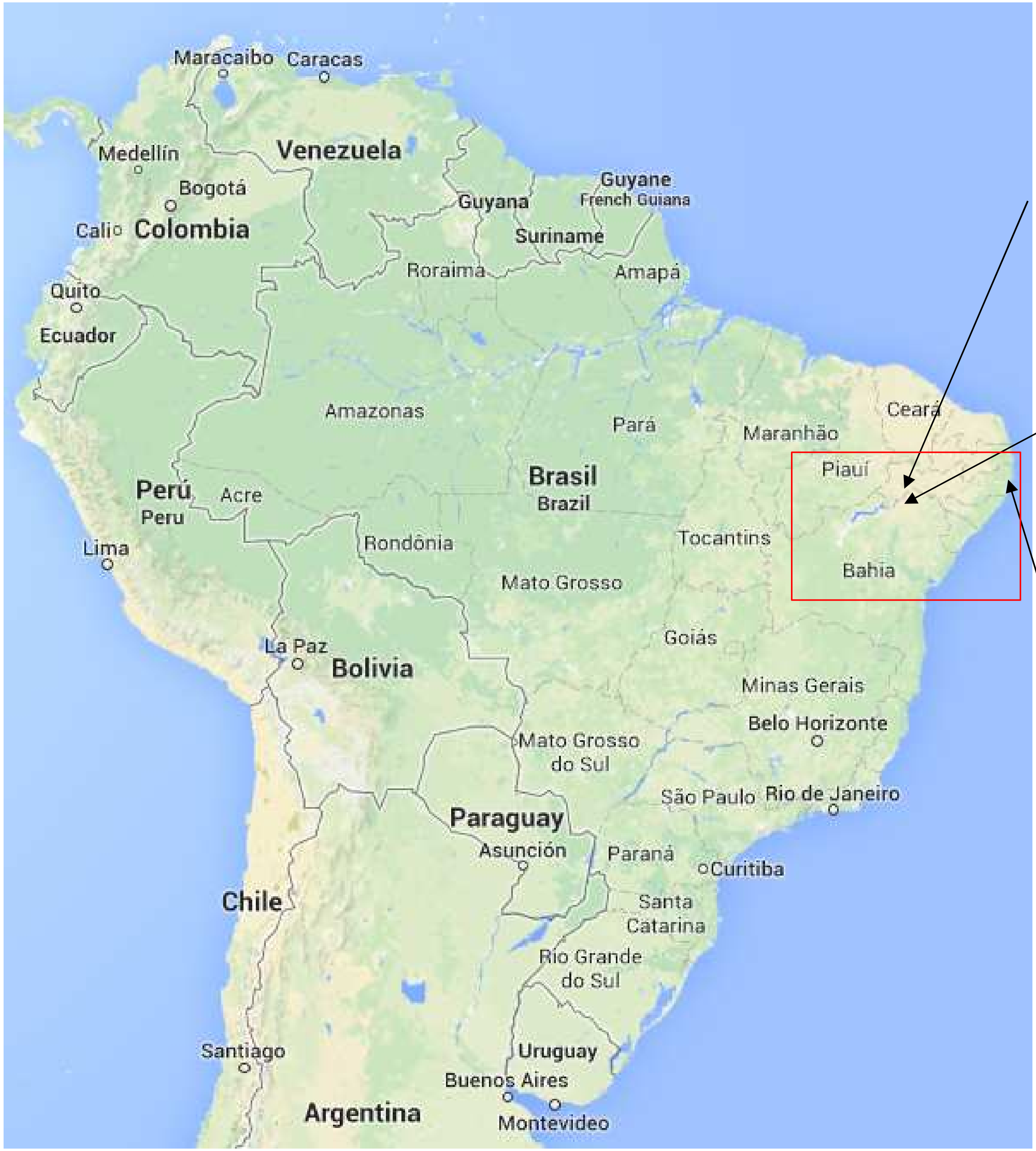
First findings

- There are not researches with real cases on the effectiveness of methods to assess requirements specifications;

Bias

The selection of publications to be included due to our access to “relevant” sources depending on the appropriateness of search strings used. The diversity of terms used in software engineering means that we might have miss some relevant studies.

A little bit about where I come from.



Petrolina - Pernambuco

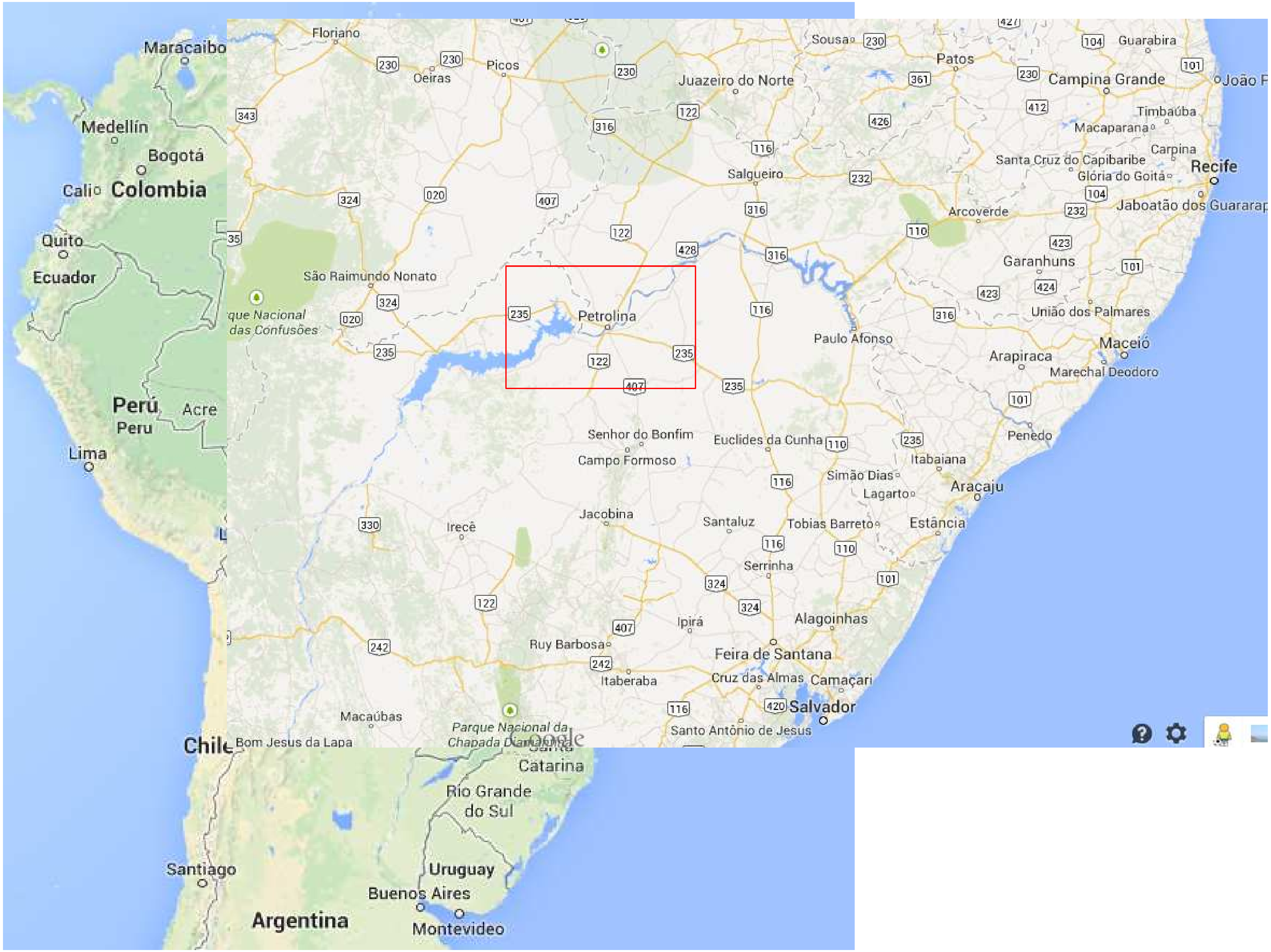
The city where I live

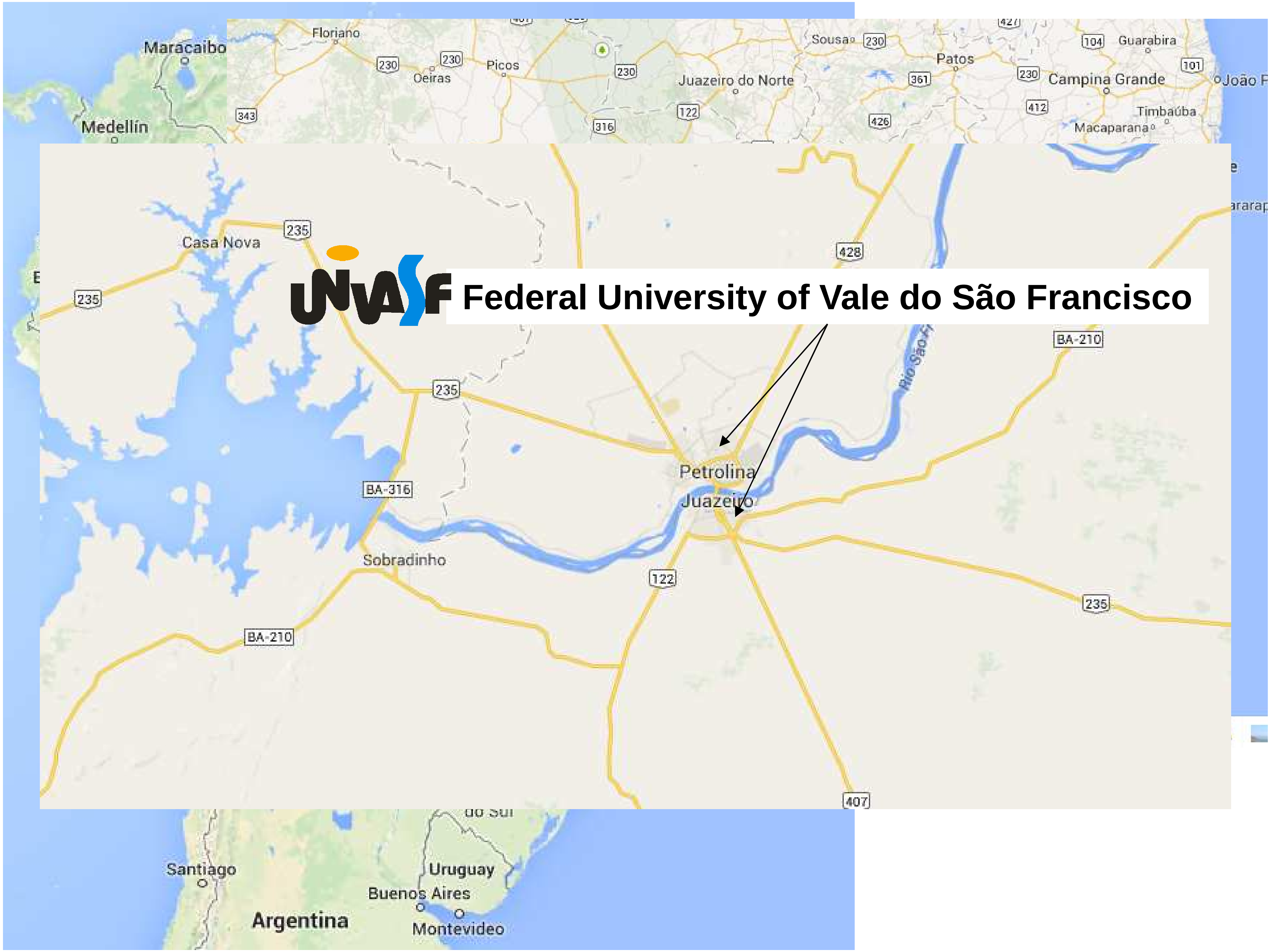
Juazeiro - Bahia

The city where I Work as associate professor at the Federal University of Vale do São Francisco

Recife - Pernambuco

The city where I held a Ph.D in Computer Science in 2009 from the Federal University of Pernambuco


Maracaibo
Medellín
Bogotá
Cali
Colombia
Quito
Ecuador
Perú
Peru
Acre
Lima
Chile
Santiago
Argentina
Buenos Aires
Montevideo
Uruguay
Rio Grande do Sul
Santa Catarina
Floriano
230
230
Oeiras
Picos
230
Juazeiro do Norte
Sousa
230
Patos
427
104
Guarabira
101
230
Campina Grande
João P
361
412
Timbaúba
Macaparana
343
316
122
426
116
Carpina
Santa Cruz do Capibaribe
Glória do Goitá
Recife
Salgueiro
232
104
232
Jaboatão dos Guararap
324
020
407
316
Arcoverde
35
122
428
316
110
423
Garanhuns
101
São Raimundo Nonato
423
424
que Nacional das Confusões
324
020
235
Petrolina
116
316
União dos Palmares
Paulo Afonso
235
122
235
Maceió
Arapiraca
Marechal Deodoro
407
235
101
Senhor do Bonfim
Euclides da Cunha
110
235
Penedo
Campo Formoso
Itabaiana
116
Simão Dias
Lagarto
Aracaju
Jacobina
330
Irecê
Santaluz
Tobias Barreto
Estância
116
110
Serrinha
101
324
122
324
Alagoinhas
407
Ipirá
242
Ruy Barbosa
Feira de Santana
242
Itaberaba
Cruz das Almas
Camaçari
116
420
Salvador
Macaúbas
Parque Nacional da Chapada Diamantina
Google
Bom Jesus da Lapa
Santo Antônio de Jesus



UNIVASF Federal University of Vale do São Francisco

230
Oeiras
230
Picos
230
316
Juazeiro do Norte
122
Sousa
230
361
426
428
Casa Nova
235
235
BA-316
Petrolina
Juazeiro
Sobradinho
122
Rio São Fr
BA-210
235
UNIVASF

Federal University of Vale do São Francisco

Federal University of Vale do São Francisco

- A new University – 10 years of existence;
- Localized in central region of Brazilian Northeast.

- So far from the biggest cities and big Universities.
- Created to Initially dedicate to graduation courses. At last 2 years were created 3 New post graduation courses.

Professor at UNIVASF

- Interdisciplinary Master “Health and Biological Sciences”.

[http://www.univasf.edu.br/~cpgcsb/]

- Computer Engineer Graduate Course

3 Works on progress

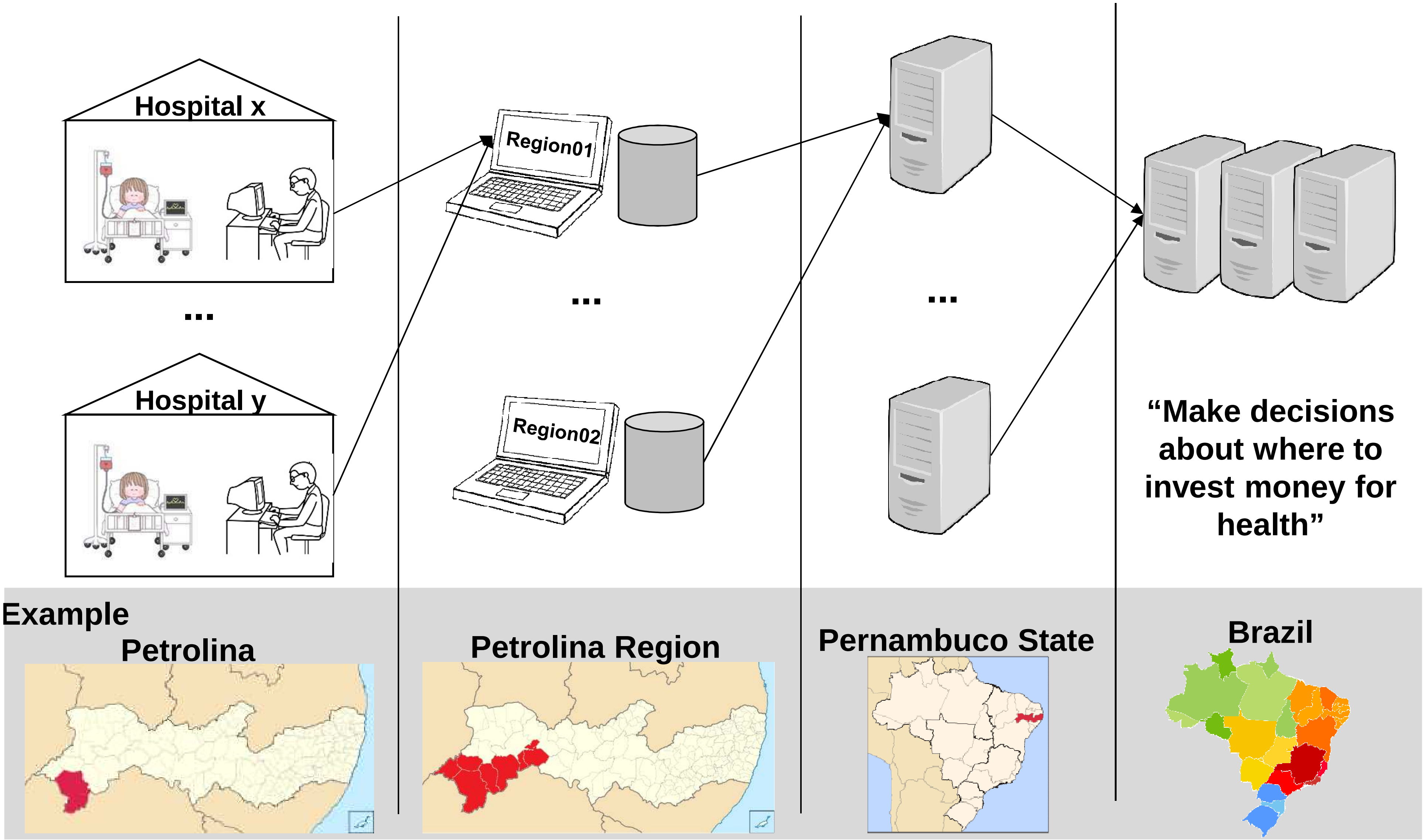
- □ QUALISIS-Br: An Approach to Improve the Quality of Brazilian Health Information Systems.
- □ Requirement Elicitation Process for a Data Management on a Biofactory.
- □ SE-Origami: A method to Teach Software Engineering Process in a Classroom.

QUALISIS-Br: An Approach to Improve the Quality of Brazilian Health Information Systems

- The main source of Brazilian health information comes from health information systems.
- Consequently, in order to obtain a reliable and secure information from this health system the data quality insurance are an essential step.

How it’s Work?

Town | Region | State | Federal



“Make decisions about where to invest money for health”

Example

Petrolina

Petrolina Region

Pernambuco State

Brazil

The Problem

What is the problem?

- Inconsistent data;
- Missing/incomplete data;
- Final reports that do not reflect the reality

Where is the problem?

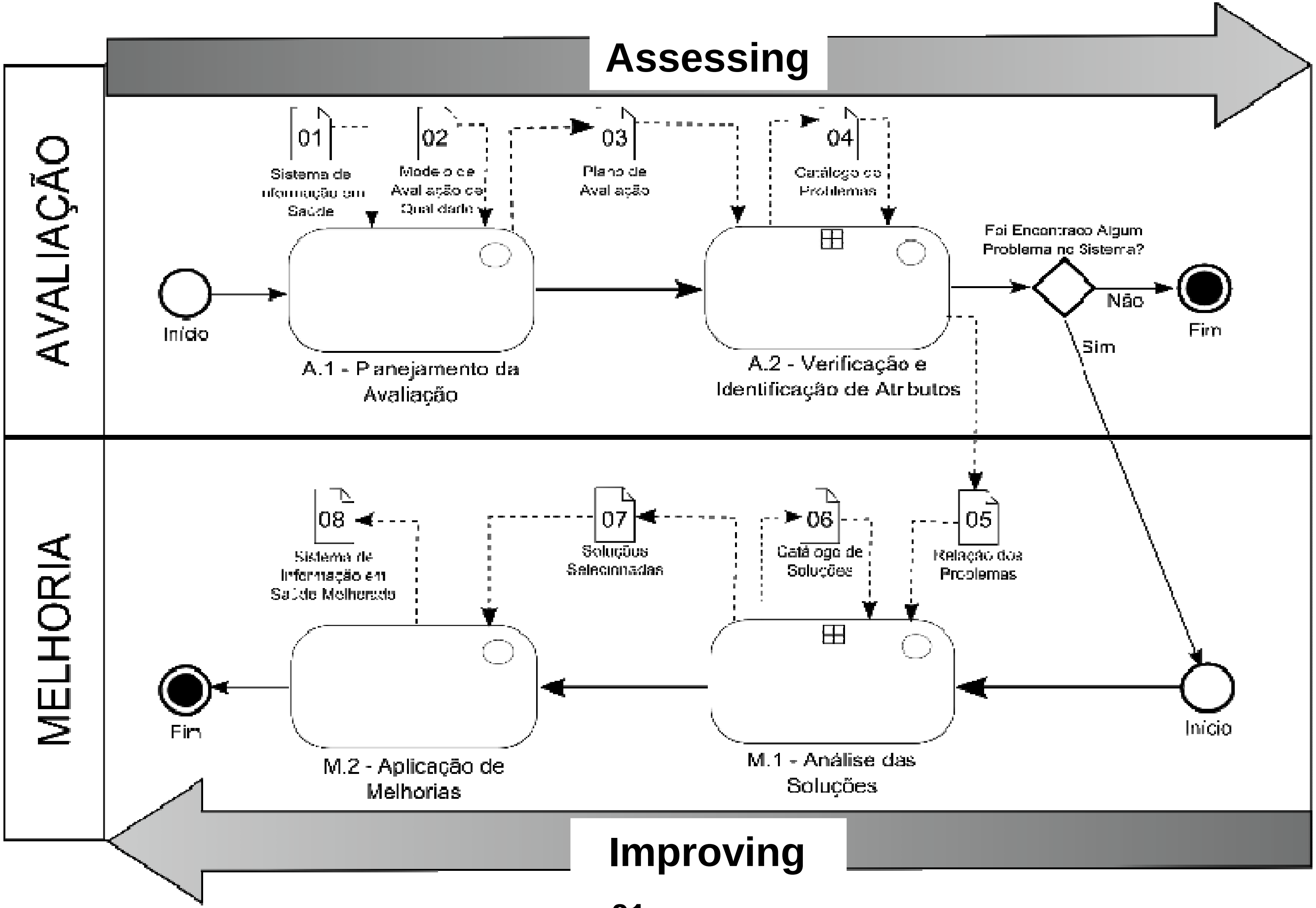
- The software system;
- The user;

How to Solve the problem?

- User Training;
- Software Update;

QUALISIS-BR

QUALISIS-BR


Assessing
AVALIAÇÃO
01
Sistema de Informação em Saúde
02
Modelo de Avaliação de Qualidade
03
Plano de Avaliação
04
Catálogo de Problemas
Início
A.1 - Planejamento da Avaliação
A.2 - Verificação e Identificação de Atributos
Foi Encontrado Algum Problema no Sistema?
Não
Sim
Fim
MELHORIA
08
Sistema de Informação em Saúde Melhorado
07
Soluções Selecionadas
06
Catálogo de Soluções
05
Relação dos Problemas
Fim
M.2 - Aplicação de Melhorias
M.1 - Análise das Soluções
Início
Improving

Preliminary Results

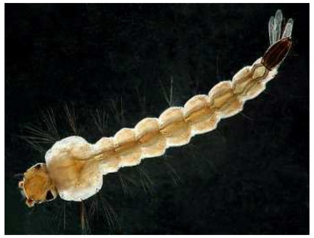
- The QUALISIS-BR was conducted in a well-known Brazilian health information system named SINAN, in Pernambuco State;
- This first conduction produced information, in catalog format, about the problems and possible solutions from SINAN.

Nurse → **JERONIMO, A. S. ; RAMOS, R. A. . QUALISIS-Br: An Approach to Improve the Quality of Brazilian Health Information Systems. IEEE Latin America Transactions, v. 13, p. 1, 2015.**

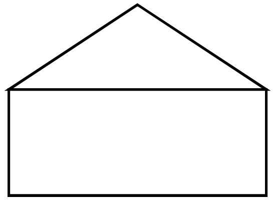
JERONIMO, A. S. ; RAMOS, R. A. . Towards to Improvement of Quality of Health Information Systems in Brazil. Brazilian Journal of Scientific Management, v. 5, p. 1, 2014.

Requirement Elicitation Process for a Data Management on a Biofactory.

Requirement Elicitation Process for a Data Management on a Biofactory

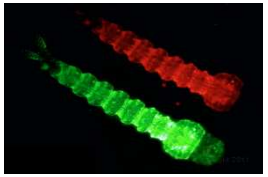


create larvae (carrying the deadly gene)



Brazilian Biofactory for the production of insects genetically modified and biological pest control

trap



Monitoring of larvae in nature

Checking the larvae bioluminescence

Production of *Aedes aegypti* genetically modified

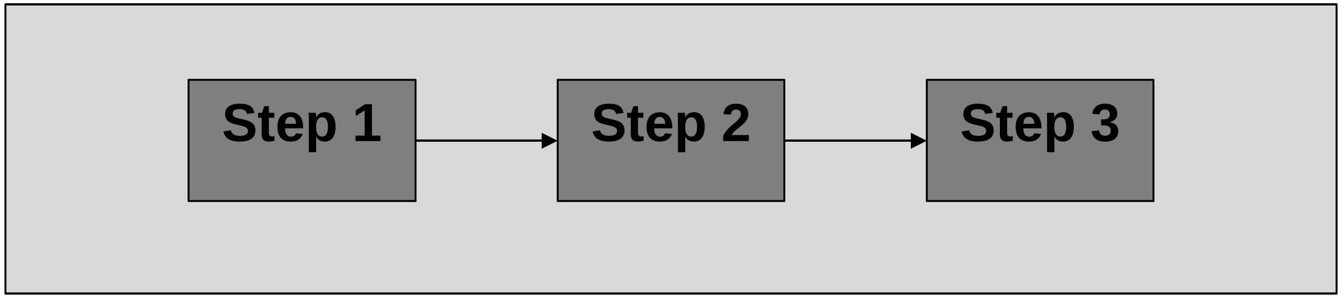
separate male and female

Liberating males

store and track growth

http://www.moscamed.org.br/

Creating a Abstraction Model



Step 1

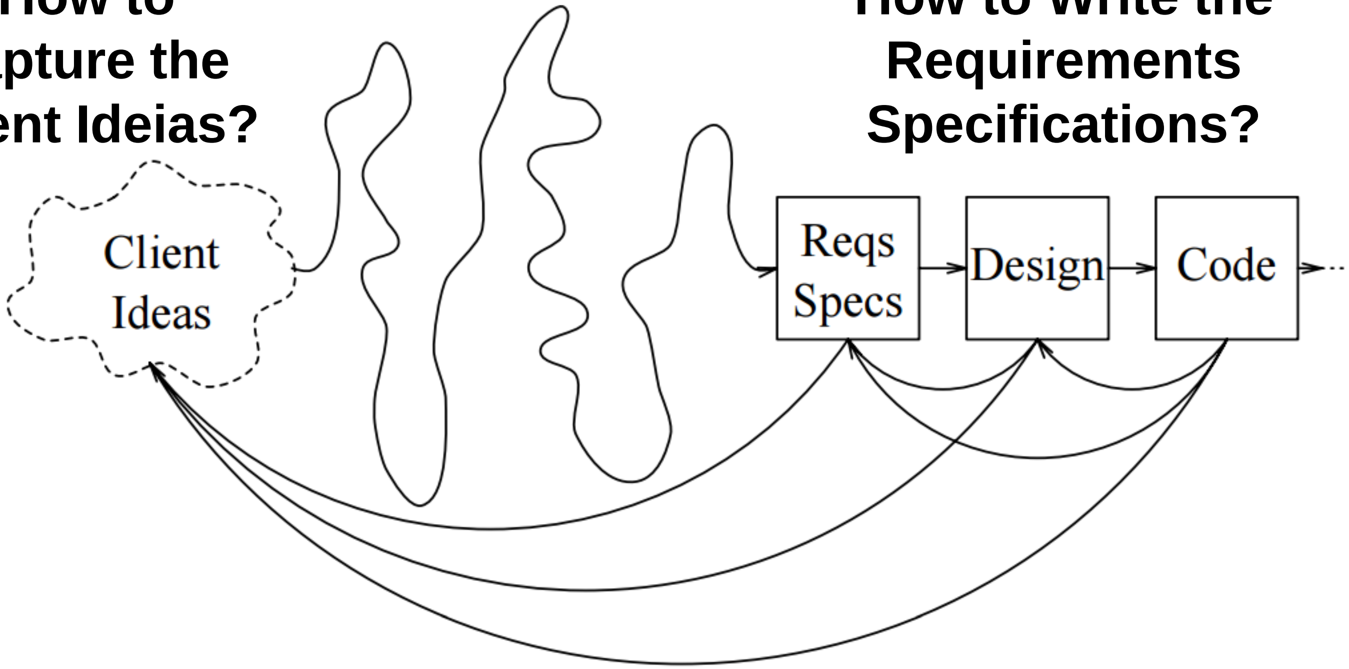
Step 2

Step 3

Create a abstraction model from it to help Software Engineering

How to capture the Client Ideias?

How to Write the Requirements Specifications?



Preliminary Results

- We finished the requirements elicitation, using ethnography, interviews and questionnaires.
- Currently, we are analyzing the requirements.
- We are planning how to validate the model that is generated.

Librarian → ALVES, R. M. ; RAMOS, R. A. . New Opportunities in Learning Studies Information and Interaction with Digital Technologies. In: Proceedings of XVIII Seminário Nacional de Bibliotecas Universitárias, 2014

SE-Origami: A method to Teach Software Engineering Process in a Classroom.

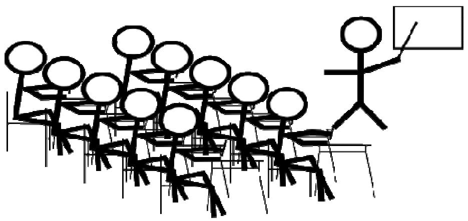
SE-Origami: A method to Teach Software Engineering Process in a Classroom

- How to teach systems development life cycle for students who are not from computer area?

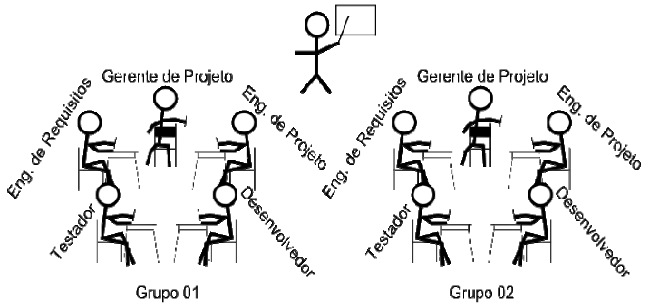
  - □ Waterfall Model
  - □ Spiral Model
  - □ V-Model

- The students will discover the advantages and disadvantages from each model.

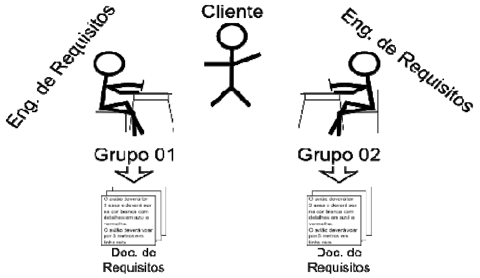
Origami Airplane by Waterfall Model



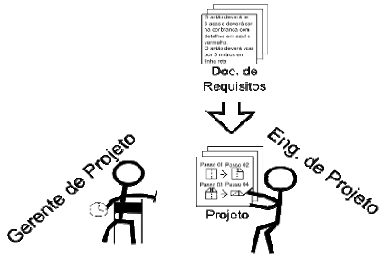
Contextualization



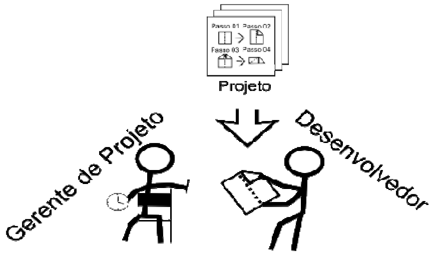
Defining the roles



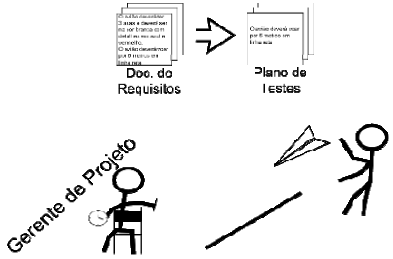
Requirements Elicitation



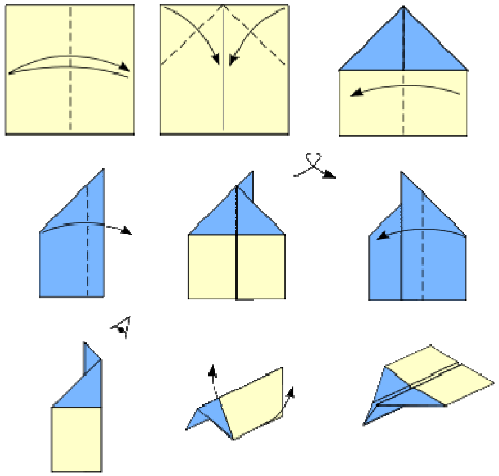
Projecting



Developing



Testing



The end!

ricargentonramos@gmail.com

Thanks to professor Dan Berry!