Automated Prediction of Defect Severity Based on Codifying Design Knowledge Using Ontologies

Martin Iliev, Bilal Karasneh, Michel R.V. Chaudron, Edwin Essenius

LIACS, Leiden University; Logica Nederland B.V.

Universiteit Leiden
The Netherlands

logica
be brilliant together

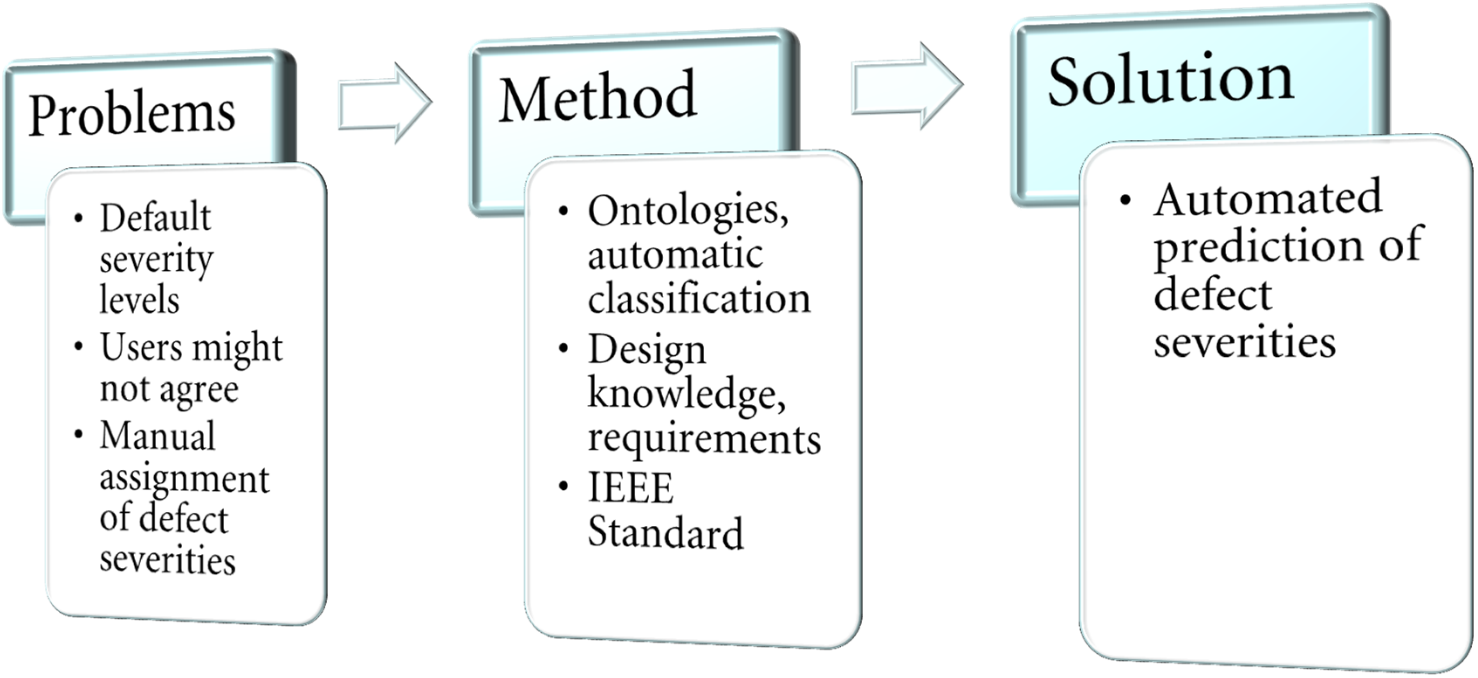
Leiden University. The university to discover.

Overview

- Introduction
- Background information
  - Ontologies
  - Case study
- Case study approach
  - Data collection
  - Data analysis and conversion
  - Data classification
- Results
- Current research
- Conclusion

Introduction

- Software testing and software defects.
- What is defect severity?
- Who assigns severity levels to defects and how?



Problems

- Default severity levels
- Users might not agree
- Manual assignment of defect severities

Method

- Ontologies, automatic classification
- Design knowledge, requirements
- IEEE Standard

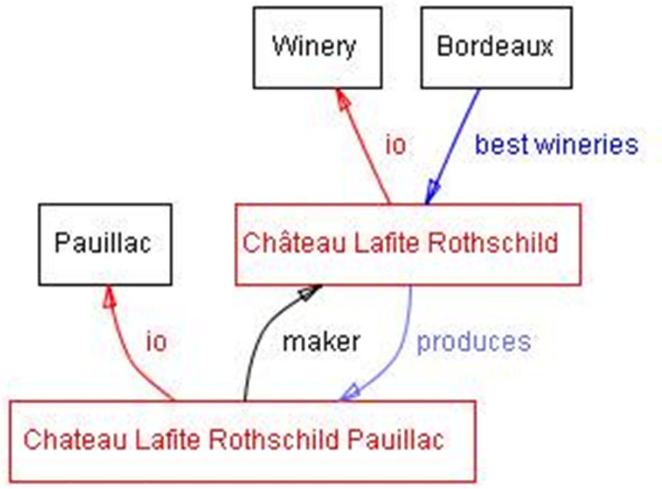
Solution

- Automated prediction of defect severities

Background Information

- Ontologies – explicit formal specifications of the terms in a domain and the relations among them.

- Industrial case study
  - Conducted at Logica, the Netherlands.
  - Logica has developed the front-end software for an embedded traffic control system.



Data Collection

- The data represent defect reports from the testing phase of the project.
- 33 out of 439 defects were selected in a representative sample from the defect tracking system.

Severity Level	**Number of Fixed Defects**		
	In all versions of the system	*In the latest version of the system*	*Selected for the case study*
Minor	85	12	5
Medium	301	93	17
Severe	47	10	10
Showstopper	6	1	1
Total	439	116	33

Data Analysis

- The selected defect reports contain *project-specific* information.
- Convert the project-specific information into *project-independent* defect attributes and their values as defined in the IEEE standard.
- Used attributes from the standard:
  - severity, effect, type, insertion activity, detection activity.

Step 1 **Step 2** Step 3

Data Conversion

Example of the information in the defect reports

Defect ID	*Severity*	*Description*	*Causes*	*Type*	*Reasons for Severity*	*Found during?*
342	Medium	The buttons for directions are reversed. When the left button is pressed…	I/O exception…	Value defect…	Wrong data is displayed…	System testing
…						

Examples after the conversion of the defects' information

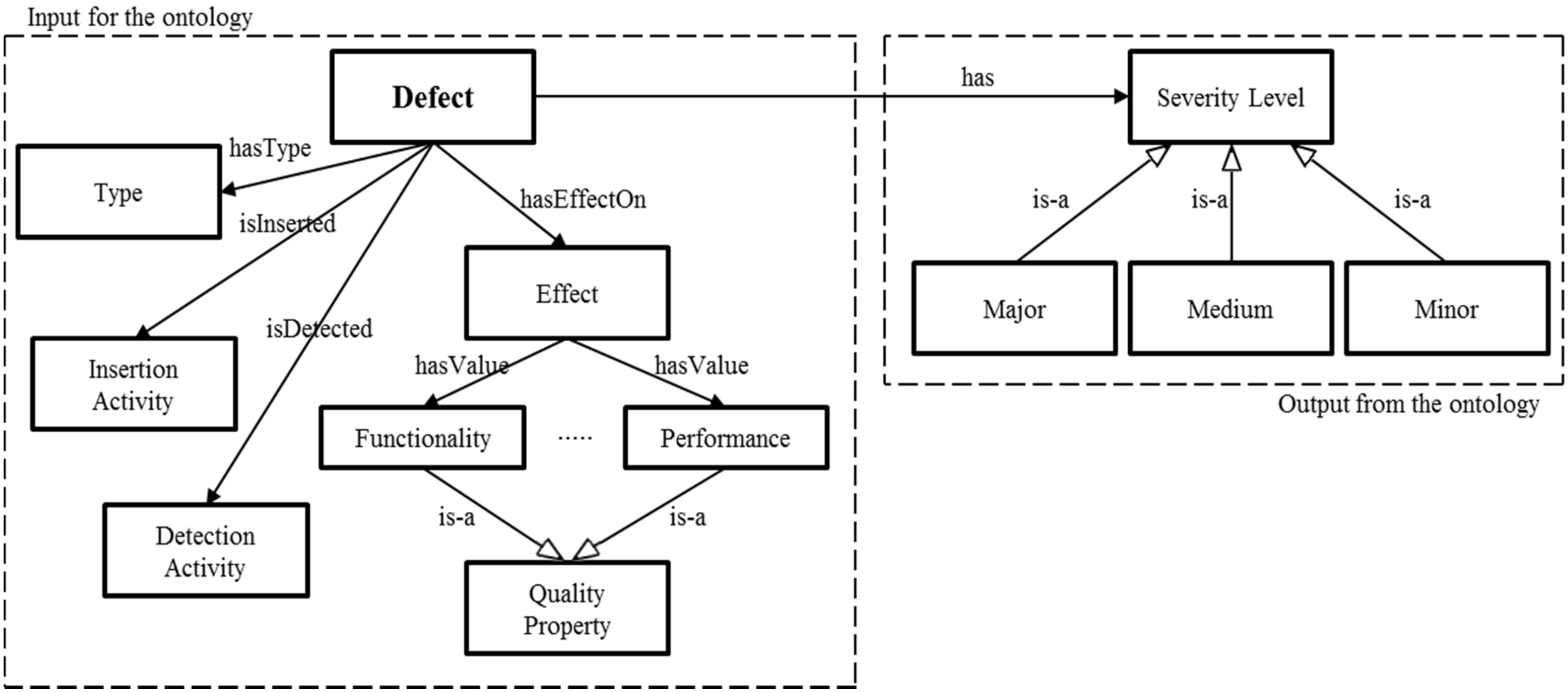
Defect ID	**Attributes**				
	Severity	*Effect*	*Type*	*Insertion Activity*	*Detection Activity*
101	Blocking	Functionality; security; performance; serviceability	Data; interface	Design	Supplier testing
102	Critical	Usability; performance	Logic	Coding	Supplier testing
…					

Data Classification

- Develop the ontology and input the converted information about the defects in it.

- Define the reasoning rules for classifying the defects into the categories
  - Major severity level – Rule 1
  - Medium severity level – Rule 2
  - Minor severity level – Rule 3

Step 1 Step 2 **Step 3**



Rule 1:

…(R1.2) (isInserted only (InDesign or InRequirements)) or ((isInserted only (InCoding or InConfiguration)) and (hasEffectOnNumber min 3)) or …

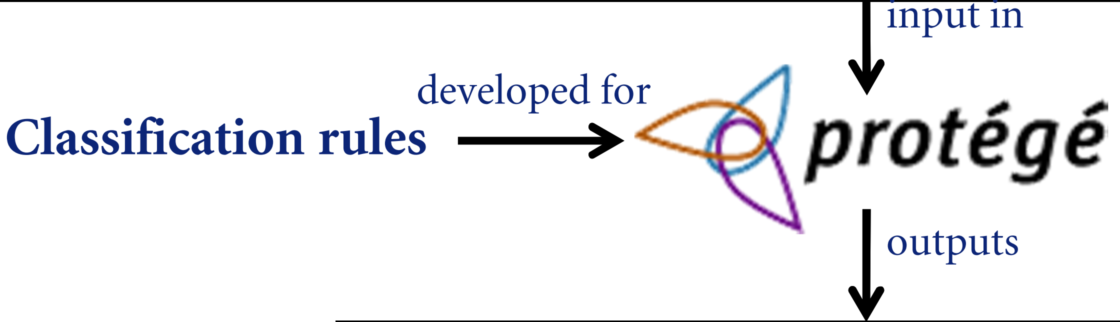
(R1.3) hasEffectOnNumber min 2

(R1.4) hasType only (Data or Interface or Logic)

(R1.5) isDetected only (FromSupplierTesting or FromCoding)

Case Study Results

Defect ID	**Attributes**			
	Effect	*Type*	*Insertion Activity*	*Detection Activity*
101	Functionality; security; performance; serviceability	Data; interface	Design	Supplier testing
102	Usability; performance	Logic	Coding	Supplier testing
103	Functionality; performance	Logic	Design	Supplier testing
…				



input in

Classification rules

developed for

outputs

Defect ID	**Predicted Severity Level**
101	Major
102	Medium
103	Major
…	…

Comparison of the Results

		Automatic (Ontology) Classification		
		MajorSL	*MediumSL*	*MinorSL*
Manual (Original) Classification	*MajorSL*	**8**	3	0
	MediumSL	7	**6**	4
	MinorSL	0	0	**5**

- Out of all defects:
  - 58% – classified in the same SLs by both classifications.
  - 42% – classified differently (21% higher, 21% lower).

- Reasons for the differences.

Current Research

- Achieved more promising results:
  - 2nd case study showed better results.

- In the process of:
  - validating the results and testing the genericity of the classification rules.
  - comparing the ontology classification results with the results obtained by an existing machine learning workbench – the Weka workbench.

WEKA
The University
of Waikato

Conclusion

- The presented method:
  - automates the process of assigning severity levels to defects.
  - could be useful for large software systems with many defects.
  - could aid in the testing phase by decreasing the workload of the test analysts.