Table. Good Practices to prevent Smells in Software Requirements.

No.	Practice	Arrangement / Prevention	Ref.	%
1	Automatic reviews of requirements artifacts.	Prevention	A1, A3, A27	10.71%
2	Validation of Bad Smells in requirements before defining test cases.	Prevention	A6	3.57%
3	Defining measurable requirements in the requirements specification.	Prevention	A7	3.57%
4	Taking into account the positive and negative case in automatically generated test cases.	Prevention	A18	3.57%
5	Preprocessing requirements written in Natural Language before moving to the design phase.	Prevention	A22	3.57%

Abbreviations: Practice Identifier (No.), Debt arrangement practice (Arrangement), Debt prevention practice (Prevention), Studies suggesting the practice (Ref.), percentage of occurrence with respect to Primary Studies (%).

Table. Software Tools for Smell Detection in Requirements.

No.	Name	Description	Ref.
1	Smella	Web-based tool to detect bad smells in requirements artifacts using part-of-speech (POS) tagging, morphological analysis and dictionaries.	A1
2	Automatic Quality User Story Artisan (AQUSA)	Software tool that employs natural language processing techniques to automatically detect quality defects in user stories and suggests possible corrections. Defect detection is performed based on violations of the quality criteria in the QUS (Quality User Story) framework, a collection of 13 criteria for user story quality.	A15
3	Ambi Detect	Tool (prototype) for automatic detection of ambiguous software requirement specifications.	A23
4	Ambiguity detection tool	A tool capable of detecting the following types of ambiguities: Lexical, syntactic, semantic, pragmatic, vague words or phrases and language errors, by using regular expressions.	A25
5	RENDEX	Measurement-based method for automated review and classification of software requirements based on needed improvements.	A27
6	ARM	Tool to automatically analyze a requirements document and produce a detailed quality report. The resulting report is based on the statistical analysis of word frequencies at various structural levels of the input document.	A28

Abbreviations: Software Tool Number (No.), Studies suggesting the tool (Ref.).

Table. Other Important Contributions on Bad Smells in Software Development.

No.	Name	Description	Ref.
1	NLP Approach	A natural language processing approach to identify ambiguous terms in the context of requirements elicitation interviews and group meetings involving different domains. Given a set of domains, the approach produces a ranked list of terms, scored by their degree of ambiguity across domains.	A11
2	Why, what and how?	Three steps are proposed to improve the quality of the requirements. The first one is the "Why", which looks to define why a requirement should be improved. The second one, the "What", looks to define the favorable characteristics that a requirement should have. And in the "How", metrics, rules, restrictions or smells to be avoided are defined which indicate how to write requirements.	A16
3	PURE	Requirements dataset for natural language processing and result generalization. The dataset consists of 79 documents in various formats and 12 documents that have been ported	A19

		to a common XML format. The documents cover multiple domains, have varying degrees of abstraction and range from product standards to public company and university project documents.	
4	State-of-the-art NLP tools	State-of-the-art of available tools and techniques for automatic ambiguity detection in software requirements written in natural language.	A22

Abbreviations: Contribution Number (No.), Studies making the proposal (Ref.).