Detecting Technical Debt Through Issue Trackers

Ke Dai
MASc Student

Supervised by
Philippe Kruchten
PhD, P.Eng, Professor
Department of Electrical and Computer Engineering
The University of British Columbia

What is Technical Debt?

"Shipping first time code is like going into debt. A little debt speeds development so long as it is paid back promptly with a rewrite... The danger occurs when the debt is not repaid. Every minute spent on not-quite-right code counts as interest on that debt. Entire engineering organizations can be brought to a stand-still under the debt load of an unconsolidated implementation, object-oriented or otherwise."

— *Ward Cunningham, 1992*

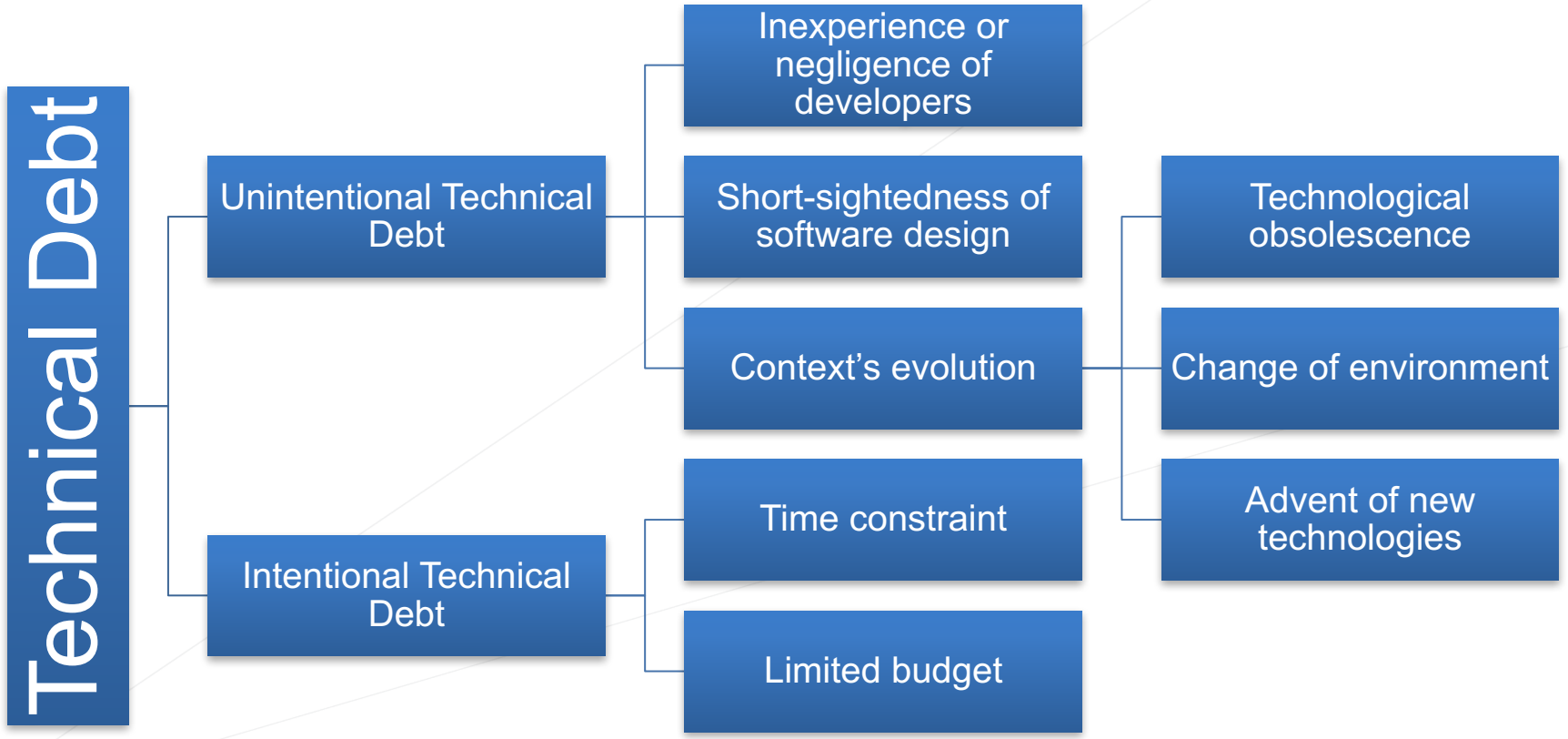
"*A design or construction approach that's expedient in the short term but that creates a technical context in which the same work will cost more to do later than it would cost to do now (including increased cost over time).*"

— *Steve McConnell, 2013*

"The term *technical debt* refers to delayed tasks and immature artifacts that constitute a 'debt' because they incur extra costs in the future in the form of increased cost of change during evolution and maintenance."

— Paris Avgeriou, Philippe Kruchten, Ipek Ozkaya, and Carolyn Seaman, 2016

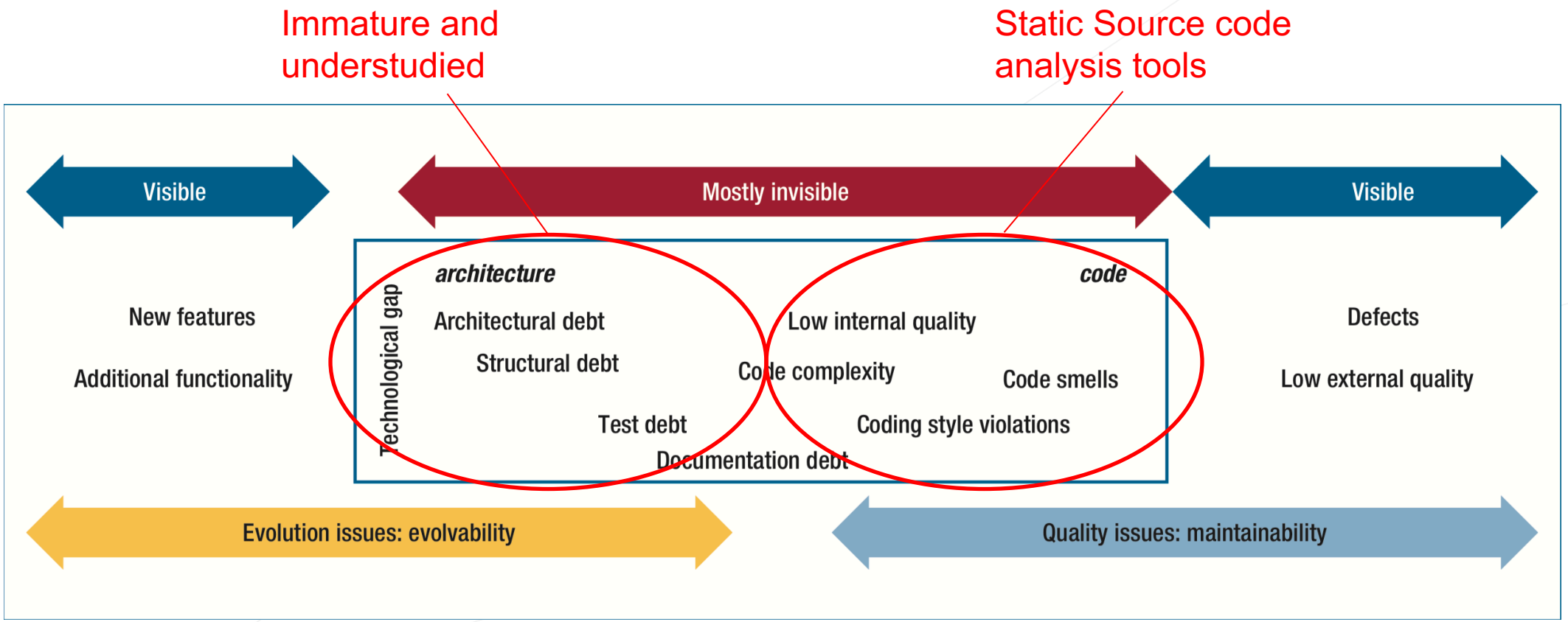
Causes of Technical Debt



Tradeoffs

Short-term Benefits	Long-term Costs
Delivering the product earlier	Increasing the cost of maintenance and evolution
Saving development costs	Reducing the productivity of development
Capturing the market	Increasing the risk of project abortion

The Scope of Technical Debt



My Research

A case study on a commercial software project

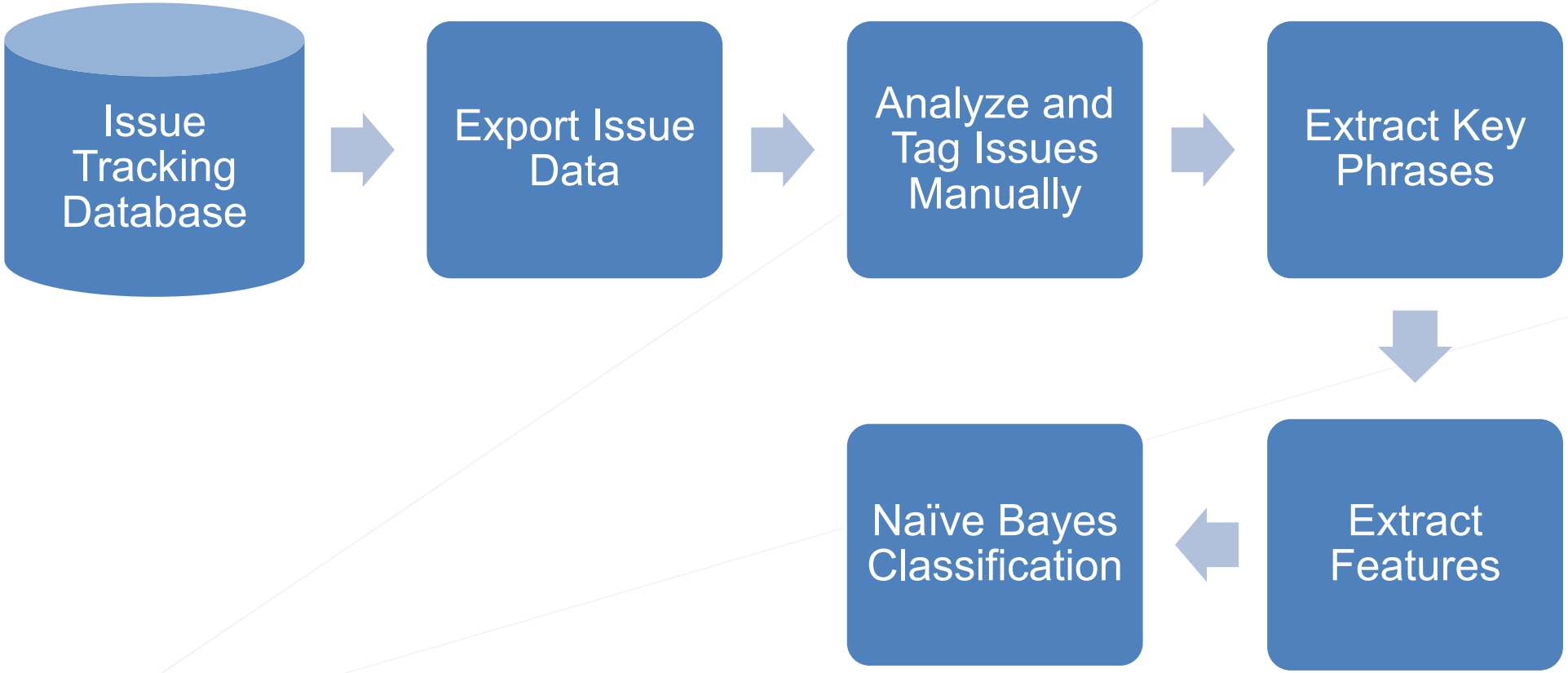
➢ Data Source

- An issue tracking data set
- Commercial software project
- Recorded in Chinese
- 8,194 samples

➢ Contributions

- A new approach to identifying technical debt
- Investigating how software developers communicate technical debt
- Automate the identification of technical debt

Approach Overview



Phase 0: Exporting issue data

	A	B	C	D	E	F	G
1	Id	Type	Priority	State	Summary	Description	Label
2	CS-15	Task	Normal	In Progress	协议细节描述，白名单协议细节描述要让用户能看懂（Plan）	协议细节描述，白名单协议细节描述要让用户看得懂， 出一个切实可行的计划书	
3	CS-16	Task	Normal	In Progress	告警，丢弃，阻断，需要一个更加容易懂的词(Doc)	Output: 一个简单文档	
4	CS-17	Task	Normal	In Progress	Signature从漏洞库导入 (Doc)	定义好接口文档，work flow	
5	CS-18	Task	Normal	To Do	描述清楚machine learning的接口（Doc）	Output： 形成一个文档，提供给restapi design用	
6	CS-20	Task	Normal	Review	POC： 用事件来验证从mw java restapi到UI新架构	Output： Demo	
7	CS-21	Task	Normal	Review	POC： 用topology来验证从UI新架构到mw java restapi	Output： Demo	
8	CS-35	Task	Normal	Review	DPI Architecture Review 结果action item整理，并转交给北京team.		
9	CS-54	Task	Normal	Review	V1.0 Feature development for 公安三所认证		
10	CS-64	Defect	Normal	Review	” 部署中 “ status is changed to "DEPLOYED" when deploying signature	See screenshot. 1. Go to signature page 2. Select any signature 3. Deploy them to the DPI boxes Result: When the system is deploying t	
11	CS-120	Task	Normal	Test In Queue	policy editing（编辑规则）(HTML/CSS)		
12	CS-125	Task	Normal	Review	Rule Template page(模板库)(HTML/CSS)		
13	CS-126	Task	Normal	Review	自定义库（HTML/CSS）		
14	CS-127	Task	Normal	Review	学习规则库剩下的页面（HTML/CSS）		
15	CS-153	Task	Normal	Review	UI: 实时监控->网络拓扑图显示		
16	CS-155	Task	Normal	Review	UI：安全设备手风琴缩略图		
17	CS-156	Task	Normal	Test In Queue	HTML/CSS：漏洞库->详情->预览规则的detail page		
18	CS-158	Task	Normal	Test In Queue	HTML/CSS：规则编辑update tools		
19	CS-160	Task	Normal	Review	HTML/CSS：规则管理		
20	CS-176	Defect	Normal	Review	UI中 “事情详情总览” 改为 “事件详情总览” 、 “设备详情总览”	UI中 “事情详情总览” 改为 “事件详情总览” 、 “设备详情总览”	
21	CS-177	Defect	Normal	Review	菜单子的颜色不对，不能都是灰色的	菜单子的颜色不对，不能都是灰色的	
22	CS-178	Defect	Normal	Review	网络拓扑图中的节点详情页信息有重叠bug	网络拓扑图中的节点详情页信息有重叠bug	
23	CS-179	Defect	Normal	Closed	设备详情页左侧的导航栏文字换行不正确	设备详情页左侧的导航栏文字换行不正确	
24	CS-182	Task	Normal	Review	UI:网络拓扑->进入，topology edit page显示但是没有编辑功能		

Phase 1: Tagging issues manually

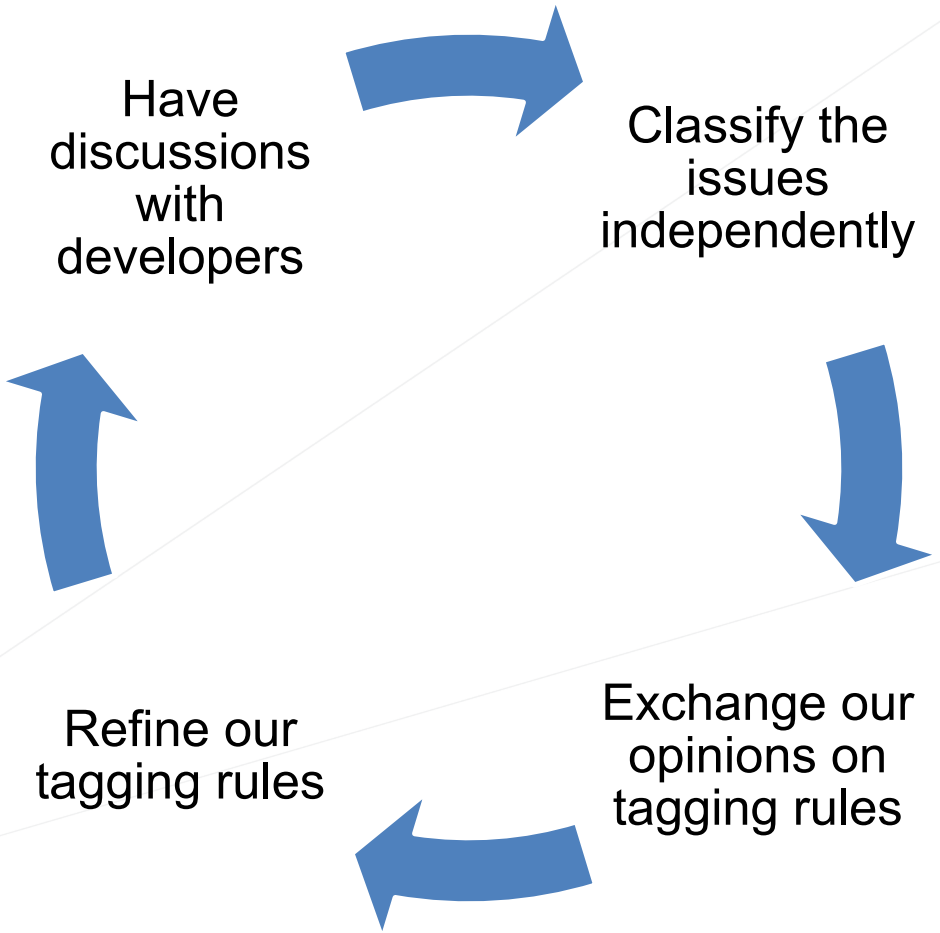
Label	Subtype	Description
Not Technical Debt	Requirement Change	The request for requirement change from the client
	New Features	Tasks to add new functions or introduce new features
	Insufficient Decription	The description is insufficient to make a decision
	Critical Defects	Critical functions or features are not implemented correctly
Technical Debt	Defect Debt	Temporarily tolerable defects that will be fixed in the future
	Requirement Debt	Requirements are not implemented accurately or implemented partially
	Design Debt	The violation of good object-oriented design principles such as god class and long method
	Code Debt	Bad coding practices such as dead code or no proper comments
	UI Debt	UI related issues such as inconsistent UI style or ugly UI elements
	Architecture Debt	Design limitation in architecture level such as the violation of modularity

Defects or Technical Debt?

- **Technical Debt**
  - Tolerable defects
  - Marginal negative impact
  - Not fixed immediately

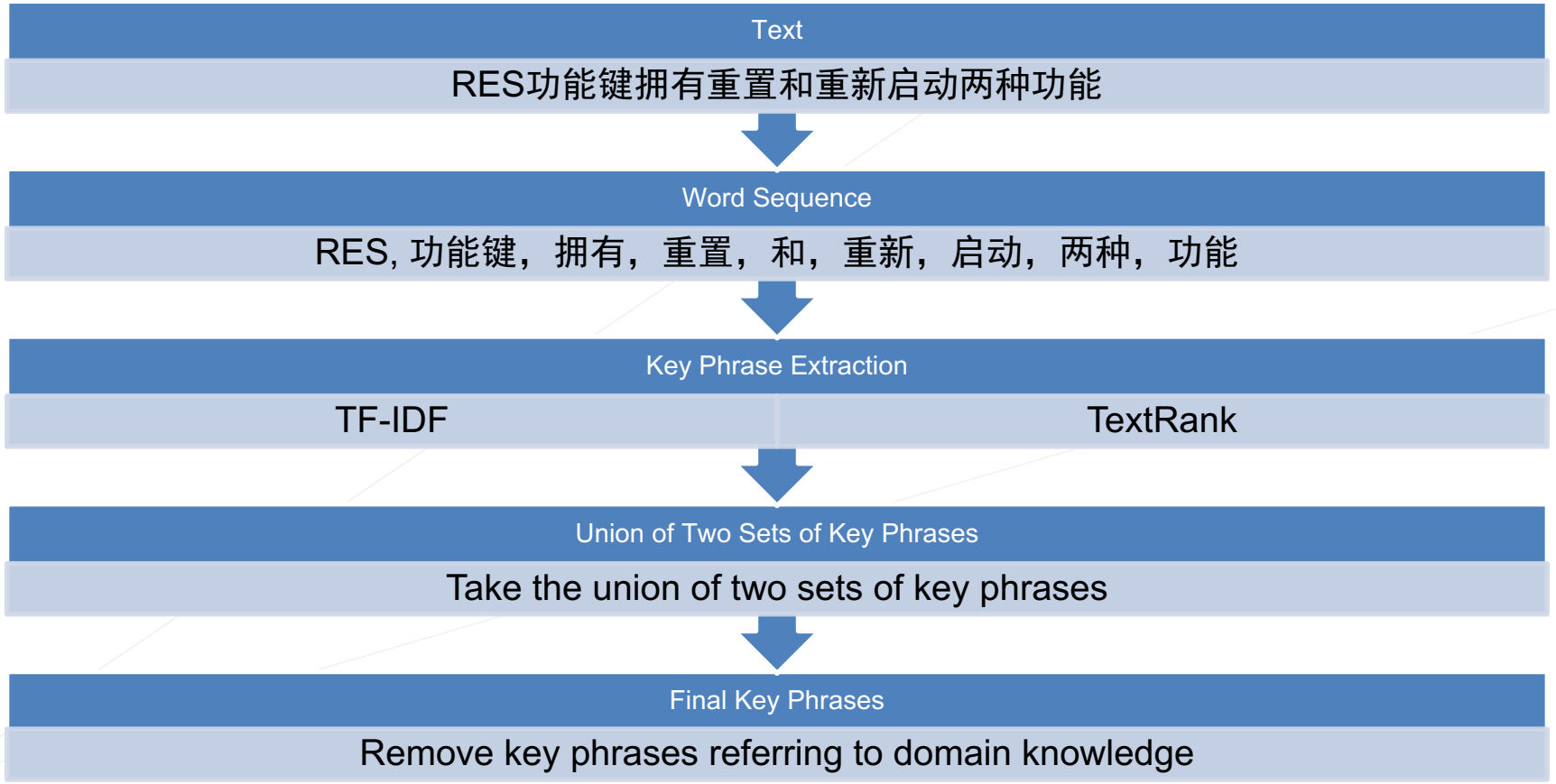
- **Not Technical Debt**
  - Critical defects
  - Fatal errors
  - Must be fixed immediately

Validation of Manual Tagging



Phase 2: Extracting key phrases

- **Tool:** Jieba (https://github.com/fxsjy/jieba/)



Final Key Phrases

114 in total, 104 in Chinese, 10 in English:

'目前', '当前', '现在', '现有', '前期', '过去', '将来', '时间', '实际', '现实', '用户', '客户', '增强', '修改', '修复', '更改', '整改', '改进', '改善', '改动', '改成', '改为', '取代', '替换', '变更', '删除', '取消', '建议', '优化', '简化', '完善', '提高', '重构', '解耦', '重新', '定义', '移植', '整合', '合并', '调整', '扩展', '期待', '计划', '管理', '维护', '功能', '需求', '设计', '规则', '理论', '策略', '机制', '算法', '数据结构', '逻辑', '代码', '结构', '架构', '构架', '风格', '样式', '格式', '性能', '效率', '充分', '安全性', '兼容性', '可扩展性', '可维护性', '稳定性', '通用性', '可用性', '可读性', '易读性', '实时性', '局限性', '更友好', '更专业', '更准确', '问题', '配置', '优先级', '不一致', '不合理', '不方便', '方便', '不清晰', '不准确', '不直观', '不美观', '不协调', '不流畅', '不符合', '不全', '异常', '缺陷', '限制', '影响', '体验', '习惯', '操作', '困难', '延迟', '卡顿', 'UI', 'risk', 'risks', 'design', 'code', 'optimise', 'optimize', 'refactor', 'refactoring', 'SonarQube'

Key Phrases

➢ Time (Accumulation)

“at present”, “now”, “current”, “previously”, “in the past”, “in the future”, “time”

➢ Modification

“strengthen”, “change”, “modify”, “replace”, “update”, “delete”, “cancel”, “optimize”, “simplify”, “perfect”, “improve”, “refactor”, “decouple”, “again”, “re-”, “replant”, “tidy”, “integrate”, “merge”, “adjust”, “extend”

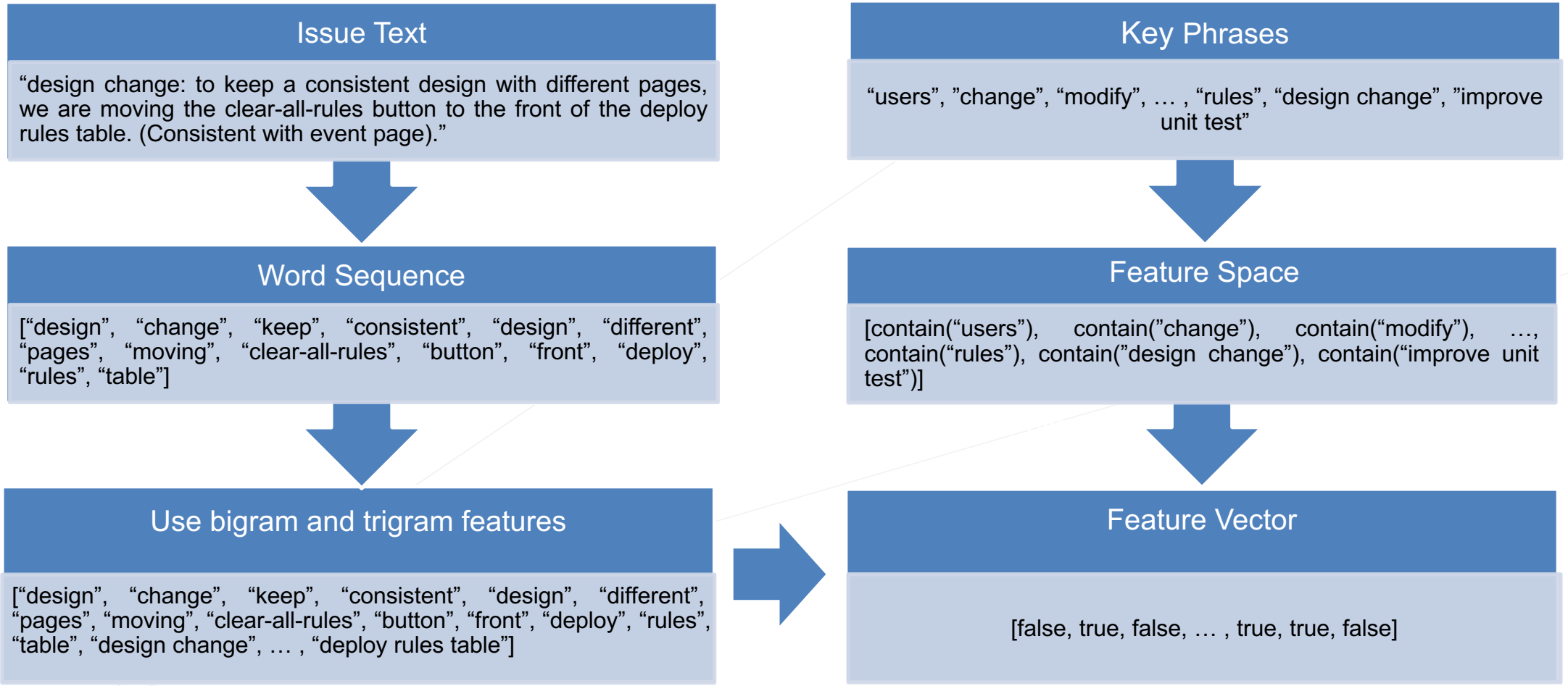
➢ Quality Attributes

“security”, “compatibility”, “scalability”, “maintainability”, “stability”, “generality”, “usability”, “readability”, “real-time”

➢ Defects or Design Limitation

“inconsistent”, “unreasonable”, “inconvenient”, “convenient”, “unclear”, “inaccurate”, 'not intuitive', “not pretty”, “incongruous”, “not smooth”, “inconformity”, “incomplete”, “abnormity”, “defect”, “limit”, “impact”, “experience”, “habit”, “operation”, “difficulty”, “delay”

Phase 3: Extracting features



Phase 4: Creating a binary Naïve Bayes Classifier

- **Naïve Bayes Algorithm**
  - based on an assumption that the features are conditionally independent of each other given the category
  - determines the category of a given sample with n-dimensional features ($x1,\ldots,xn$) by calculating the probability that the sample belongs to each category and then assigning the most probable category c to it
- **Tool: NLTK (http://www.nltk.org)**
- **Repeated random sub-sampling validation**
  - repeatedly splitting the full data set into 80/20% randomly distributed partitions
  - training and testing the classifier for each split
  - recording performance results

Conclusion

Category	Average Precision	Average Recall	Average F1-score
Technical Debt	0.72	0.81	0.76

- The term technical debt were found in the issue data set.
- All technical debt instances were expressed implicitly.
- Text patterns indicating technical debt exist.

20 Most Informative Features for Detecting Technical Debt	
Features	Likelihood Ratio (Technical Debt : not Technical Debt)
协议识别优化(protocol identification optimization) = 1	155.2 : 1.0
增强 (strengthen) = 1	128.2 : 1.0
不方便 (inconvenient) = 1	128.2 : 1.0
提高 (improve) = 1	117.4 : 1.0
优化 (optimize) = 1	90.8 : 1.0
整改 (change or modify) = 1	87.7 : 1.0
风格 (style) = 1	65.2 : 1.0
体验 (experience) = 1	64.4 : 1.0
改进 (improve) = 1	60.7 : 1.0
不容易 (not easy) = 1	47.2 : 1.0
改善 (improve) = 1	44.5 : 1.0
效率 (efficiency) = 1	44.5 : 1.0
简化(simplify) = 1	38.2 : 1.0
解决方案(strategy) = 1	35.8 : 1.0
困难(difficulty) = 1	33.7 : 1.0
前期(previously) = 1	33.7 : 1.0
不美观(not pretty) = 1	33.7 : 1.0
risk = 1	33.7 : 1.0
算法(algorithm) = 1	31.8 : 1.0
习惯(habit) = 1	31.8 : 1.0

Limitation and Future Work

- **Limitation**
  - Limited issue data set
  - One classification algorithm
  - Simple feature extraction method

- **Future work**
  - Multi-classifier
  - Sophisticated feature extraction methods
  - Other classification algorithms: random forest, deep learning

Thank you!

谢谢！