Explicating, Understanding and Managing Technical Debt from Self-Driving Miniature Car Projects

Md Abdullah Al Mamun

Christian Berger

Jörgen Hansson

Presenter: Antonio Martini

Division of Software Engineering

Department of Computer Science & Engineering

Setting the scene

- Self-driving miniature cars
- *CaroloCup* international competition
- Student projects
  - Project in 2014 reused software from 2013

2013 Meili Project

Reuse

Maintenance

2014 Legendary Project

- Feature analyzed: lane-detection
  - Core feature

Objective & Research Questions

- **Objective:** Understand the evolution of the technical debt in the development of the cars so that proper actions can be planned to reduce the debt to have more reusable and maintainable software.

- Research Questions
  - RQ1: How did the technical debt evolve over time for a self-driving miniature car that competed twice in an international competition?
  - RQ2: How did the experience from a previous participation influence the technical debt of the recent software?
  - RQ3: What are the most important issues related to the development process that need to be adapted to reduce the technical debt for teams?

Methods

- Case Study

2013 Meili Project

2014 Legendary Project

- Questionnaire

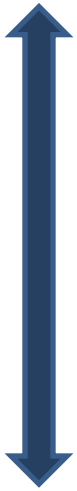
Case Study

- SonarQube tool
  - Code analysis tool for technical quality
  - Based on SQALE (Software Quality Assessment based on Lifecycle Expectations)
- Two versions of a core feature
- Git commits
- Data collection
  - Code size (LOC, lines, statements, files, functions),
  - Duplications (% of duplication, by lines, by blocks, by files),
  - Complexity (by functions, by files, total complexity),
  - Technical debt (in person hour),
  - # of identified issues of categories (blocker, critical, major, minor, info)

Case Study Result

Severity	2013 project # of detected issues	2014 project # of detected issues
Blocking	-	-
Critical	-	26
Major	37	451
Minor	29	208
Info	6	43
Total	72	728

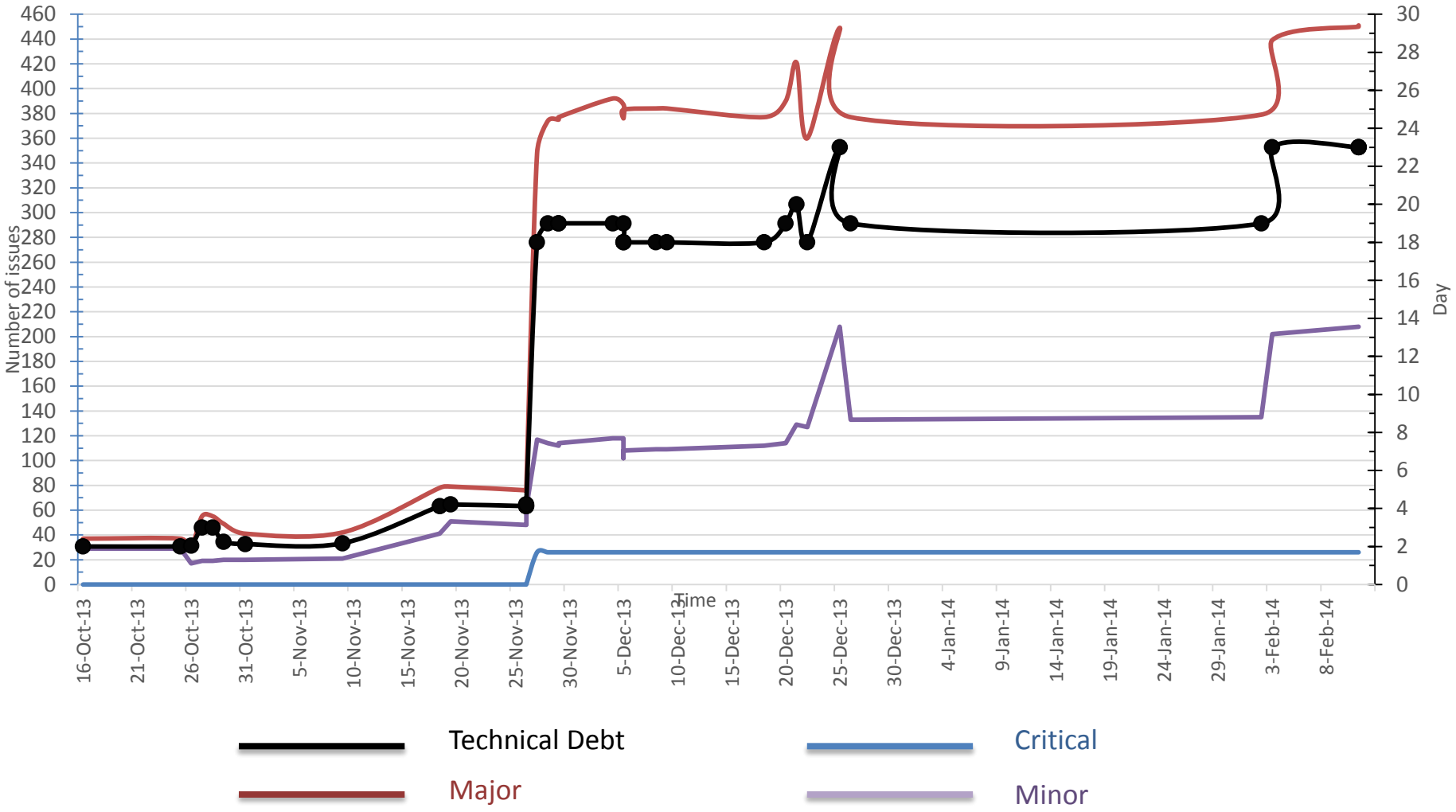
Higher Priority



- **Blocking**: Not found
- **Critical**: The right hand operand of a logical && or || operator shall not contain side effects
- **Major**: Insecure functions “strcpy”, “strcat”, and “sprintf” should not be used
- **Minor**: Magic numbers should not be used
- **Info**: Comments should not be located at the end of lines of code

Lower Priority

Case Study Result



(The 'Technical Debt' graph is connected to the vertical-right axis scale and the rest of the graphs are connected to the vertical-left axis scale)

Issue Selection for further Investigation

Severity	2013 project # of detected issues	2014 project # of detected issues
Blocking	-	-
Critical	-	26
Major	37	451
Minor	29	208
Info	6	43
Total	72	728

- Sorted according to frequency
- Top 3 issues are selected from both features under the 3 selected categories
- Selected issues are code smells
- Selected issues alone accumulate about half of the total debt in both projects
  - Higher number of occurrence than design debts

Occurrence	Issues
18	**CS1**: “new” and “delete” should be used
8	**CS2**: The right hand operand of a logical && or || operator shall not contain side effects
101	**CS3**: Sections of code should not be “commented out”
77	**CS4**: If-else statements must use braces
40	**CS5**: The statement forming the body of a switch, while, do . . . while, and for-statement shall be a compound statement
19	**CS6**: Nested code blocks should not be used
5	**CS7**: Insecure functions “strcpy”, “strcat”, and “sprintf” should not be used
126	**CS8**: Magic numbers should not be used
45	**CS9**: If statements should not be nested too deeply
39	**CS10**: An init-declarator-list or a member-declaratory-list shall consist of a single init-declarator or member-declarator, resp.
7	**CS11**: Switch statements should have at least three case clauses

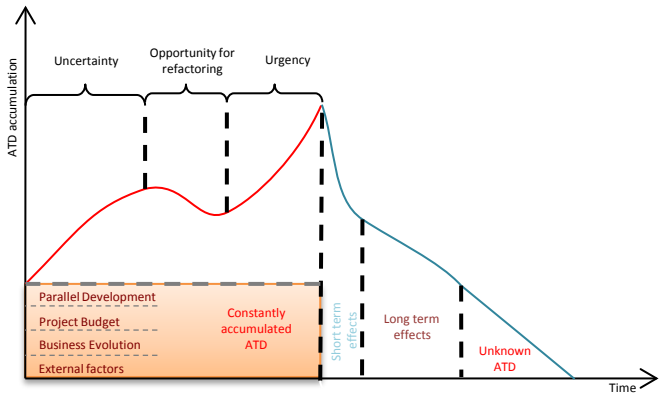
Follow-up questionnaires

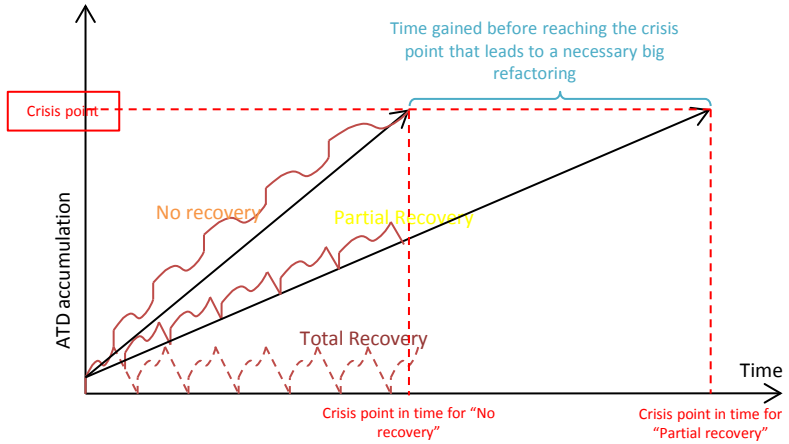
- Questionnaire 1
  - Selected TD issues
  - Factors for accumulation of TD
  - Developers perception of the severity of the issues
    - Checked against SonarQube levels

- Questionnaire 2
  - Cross-check of severity perception
  - Agreement of the developers with the severity proposed by SonarQube

Factors influencing TD accumulation

- Taxonomy of the factors influencing TD accumulation
  - Especially related to architectural technical debt
- Graphs of the trends of accumulation over time (hypotheses)





- Previous qualitative study in 7 large software organizations
  - Martini, A., Bosch, J., Chaudron, M., 2014. *Architecture Technical Debt: Understanding Causes and a Qualitative Model*, in: 40th Euromicro Conference on Software Engineering and Advanced Applications

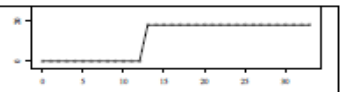
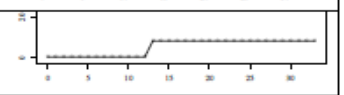
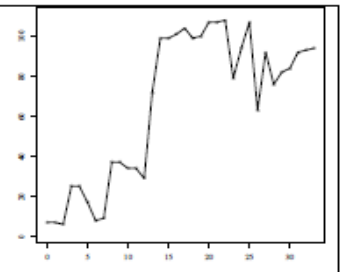
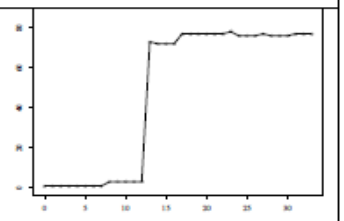
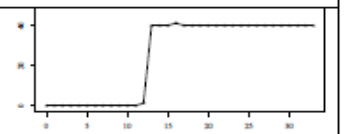
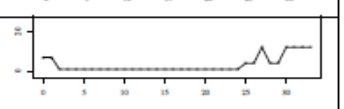
antonio.martini@chalmers.se

Questionnaire-1

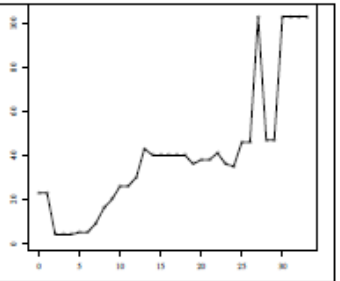
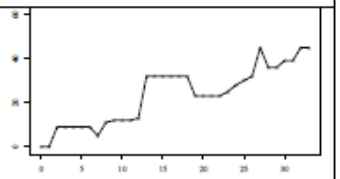
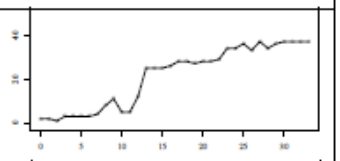
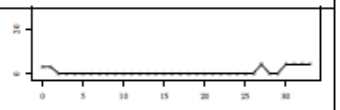
- Influencing factors ranked by the participants

Factors	2013 project		2014 project	Overall rating
	Developer I	Developer II	Developer III	
Time pressure: Deadlines with penalties	●●●○	●●●●	●●●●	●●●◕
Issues related to software/hardware integration	●●●●	●●●○	●●●○	●●●◔
Incomplete refactoring	●●●○	●●●●	●●●○	●●●◔
Human factors	●●●●	●●●○	●●○○	●●●○
Lack of experience with the middleware	●●●●	●●○○	●●●○	●●●○
Lack of domain experience	●●●●	●●○○	●●○○	●●◕○
Uncertainty of use cases in the beginning	●●●○	●●●○	●●○○	●●◕○
Reuse of Legacy / third party / open source	●○○○	●●●○	●●●○	●●◔○
Priority of features over product	●○○○	●●●●	●●○○	●●◔○
Lack of experience with the tools	●●●○	●●○○	●●○○	●●◔○
Technology evolution	●●○○	●●○○	●●●○	●●◔○
Malfunctioning hardware	●●●○	●●●○	●○○○	●●◔○
Parallel development	●●○○	●●●●	●○○○	●●◔○

Questionnaire-1

Severity	Top 3 code smells from the selected categories	SonarQube	2013 project		2014 project	Code review by SonarQube				Code smells over time for the 2014 project (X-axis: revision number, Y-axis: number of code smell occurrence)
						Latest code from 2013 project		Latest code from 2014 project		
			Developer I	Developer II	Developer III	Number of occur-rence	Fixing time (hour)	Number of occur-rence	Fixing time (hour)	
Critical	CS1	●●●●	●●●○ (lr)	●●●○ (lr)	●●○○ (dr)	-	-	18	6	
	CS2	●●●●	●●●○ (lr)	●●●○ (lr)	●●●● (gr)	-	-	8	2	
Major	CS3	●●●○	●●○○ (lr)	●●●● (lb)	●●○○ (lr)	7	2.33	94	31.33	
	CS4	●●●○	●●●○ (gr)	●●○○ (lr)	●●●○ (gr)	-	-	77	12.83	
	CS5	●●●○	●●●○ (gr)	●●○○ (lr)	●●●○ (gr)	-	-	40	6.67	
	CS6	●●●○	●●●● (lb)	●●●● (lb)	●●●○ (gr)	7	1.17	12	2	
	CS7	●●●○	●●●● (lb)	●●●○ (gr)	●●●○ (gr)	5	1.67	-	-	

Questionnaire-1

Severity	Top 3 code smells from the selected categories	SonarQube	2013 project		2014 project	Code review by SonarQube				Code smells over time for the 2014 project (X-axis: revision number, Y-axis: number of code smell occurrence)
						Latest code from 2013 project		Latest code from 2014 project		
			Developer I	Developer II	Developer III	Number of occur-rence	Fixing time (hour)	Number of occur-rence	Fixing time (hour)	
Minor	CS8	●●○○	●●●● (db)	●●●● (db)	●●○○ (gr)	23	1.92	103	8.58	
	CS9	●●○○	●●●● (db)	●●●○ (lb)	●●○○ (gr)	-	-	45	15	
	CS10	●●○○	●●●○ (lb)	●●●○ (lb)	●●●○ (lb)	2	0.17	37	3.08	
	CS11	●●○○	●●○○ (gr)	●●○○ (gr)	●●●○ (lb)	3	0.25	4	0.33	
Total hours to fix the selected code smells						1 day		11 days		
Total days to fix all issues			10 days	7 days	3 days	2 days		23 days		

Legend:
●●●●:Critical, ●●●○: Major, ●●○○: Minor, ●○○○: None
Indication of Color: *Green* (gr) - perfect match,
Light Blue (lb) - overrated by one notch, *Dark Blue* (db) - overrated by two notches,
Light Red (lr) - underrated by one notch, *Dark Red* (dr) - underrated by two notches.

Questionnaire 2

24.24% disagreement

Developers' agreement with the CS ranking

Severity	Code Smells	2013 project Developer I	2013 project Developer II	2014 project Developer III
Critical	CS1	●●●●	●●●○	●●●○
	CS2	●●●●	●●●○	●●○○
Major	CS3	●●●○	●●○○	●●○○
	CS4	●●●●	●●○○	●●●●
	CS5	●●●○	●●○○	●●●●
	CS6	●●●○	●●●○	●●○○
	CS7	●●○○	●●●○	●●●●
Minor	CS8	●●●●	●●●○	●●●○
	CS9	●●●○	●●●○	●●●○
	CS10	●●●○	●●●○	●●●○
	CS11	●●●○	●●●○	●●○○

- **CS1**: "new" and "delete" should be used
- **CS2**: The right hand operand of a logical && or || operator shall not contain side effects
- **CS3**: Sections of code should not be "commented out"
- **CS4**: If-else statements must use braces
- **CS5**: The statement forming the body of a switch, while, do . . . while, and for-statement shall be a compound statement
- **CS6**: Nested code blocks should not be used
- **CS7**: Insecure functions "strcpy", "strcat", and "sprintf" should not be used
- **CS8**: Magic numbers should not be used
- **CS9**: If statements should not be nested too deeply
- **CS10**: An init-declarator-list or a member-declaratory-list shall consist of a single init-declarator or member-declarator, respectively
- **CS11**: Switch statements should have at least three case clauses

21.21% Strongly Agree

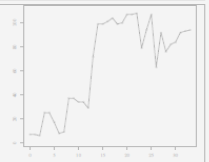
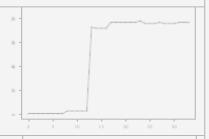
54.55% Agree

Analysis

Severity	Code Smells	2013 project Developer I	2013 project Developer II	2014 project Developer III
Critical	CS1	●●●●	●●●○	●●●○
	CS2	●●●●	●●●○	●●○○
Major	CS3	●●●○	●●○○	●●○○
	CS4	●●●○	●●○○	●●●●
	CS5	●●●○	●●○○	●●●●
	CS6	●●●○	●●○○	●●○○
	CS7	●●●○	●●●○	●●●●
Minor	CS8	●●●●	●●●○	●●●○
	CS9	●●●○	●●●○	●●●○
	CS10	●●●○	●●●○	●●●○
	CS11	●●●○	●●●○	●●○○

24.24% Disagreement

27% Underrating

Severity	Top 3 code smells from the selected categories	SonarQube	2013 project Developer I	2013 project Developer II	2014 project Developer III	Code review by SonarQube: Latest code from 2013 project Number of occurrence	Latest code from 2013 project Fixing time (hour)	Latest code from 2014 project Number of occurrence	Latest code from 2014 project Fixing time (hour)	Code smells over time for the 2014 project (X-axis: revision number, Y-axis: number of code smell occurrence)
Critical	CS1	●●●●	●●●○ (lr)	●●●○ (lr)	●●○○ (dr)	-	-	18	6	
	CS2	●●●●	●●●○ (lr)	●●●○ (lr)	●●●● (gr)	-	-	8	2	
Major	CS3	●●●○	●●○○ (lr)	●●●● (lb)	●●○○ (lr)	7	2.33	94	31.33	
	CS4	●●●○	●●●○ (gr)	●●○○ (lr)	●●●○ (gr)	-	-	77	12.83	
	CS5	●●●○	●●●○ (gr)	●●○○ (lr)	●●●○ (gr)	-	-	40	6.67	
	CS6	●●●○	●●●● (lb)	●●●● (lb)	●●●○ (gr)	7	1.17	12	2	
	CS7	●●●○	●●●● (lb)	●●●○ (gr)	●●●○ (gr)	5	1.67	-	-	
Minor	CS8	●●○○	●●●● (db)	●●●● (db)	●●○○ (gr)	23	1.92	103	8.58	
	CS9	●●○○	●●●● (db)	●●●○ (lb)	●●○○ (gr)	-	-	45	15	
	CS10	●●○○	●●●○ (lb)	●●●○ (lb)	●●●○ (lb)	2	0.17	37	3.08	
	CS11	●●○○	●●○○ (gr)	●●○○ (gr)	●●●○ (lb)	3	0.25	4	0.33	
Total hours to fix the selected code smells							7.5 hours		87.83 hours	
Total days to fix all issues			10 days	7 days	3 days		2 days		23 days	

Legend:
●●●●:Critical, ●●●○: Major, ●●○○: Minor, ●○○○: None
Indication of Color: *Green* (gr) - perfect match,
Light Blue (lb) - overrated by one notch, *Dark Blue* (db) - overrated by two notches,
Light Red (lr) - underrated by one notch, *Dark Red* (dr) - underrated by two notches.

Summary

- Lack of knowledge is not the foremost driving factor for TD accumulation

- The most important factors contributing to the TD are
  - Use of legacy/3rd Party/freely available code
  - Time pressure
  - Issues related to software-hardware integration
  - Incomplete refactoring

Thank You