# **Development plan**

Zohour ABOUAKIL Sofia BOUTAHAR David COURTINOT Xiaowen JI Fabien SAUCE

Recherche de motifs dans un code C++ à l'aide de la logique temporelle

# Sommaire

Ι	Pro	ject de	escription and objectives test test	<b>2</b>
	I.1	Surrou	undings of the project	2
	I.2		et description	2
		I.2.1	Main idea	2
		I.2.2	Related technologies	2
		I.2.3	Project parts	2
		I.2.4	To conclude	3
	I.3	project	3	
		I.3.1	Define priorities	3
		I.3.2	Deliverable documents	3
II	Pro	ject or	ganization	4
		-	lefinition	4
	II.2	Develo	opment organisation	5
		II.2.1	Usage of Scrum method	5
		II.2.2	Team repartition approach	5
	II.3	Tasks	organisation	5
		II.3.1	Tasks definition	5
		II.3.2	Planning	6
III	Risl	k mana	agement	8
IV	Cod	le man	agement	10
	IV.1	Qualit	y management	10
		IV.1.1	Automated coding style checks	10
	IV.2	Test st	trategy	10
	IV.3	Config	guration management	10
$\mathbf{v}$	App	endice	es	11

### Partie I

# Project description and objectives test test

#### I.1 Surroundings of the project

Le projet long à l'ENSEEIHT Organisation du projet

Le client c est qui?? Les noms, leurs fonctions, les motivations du projet

Nos motivations – pas sur

### I.2 Project description

#### I.2.1 Main idea

#### I.2.2 Related technologies

- Coccinelle
- Clang

#### I.2.3 Project parts

- Parser
- CTL
- Model checking

- I.2.4 To conclude
- I.3 Final project
- I.3.1 Define priorities
- I.3.2 Deliverable documents

### Partie II

# Project organization

### II.1 Role definition

Project manager

Quality manager

Test manager

Test manager

Configuration manager

Documentation manager

#### Chain development

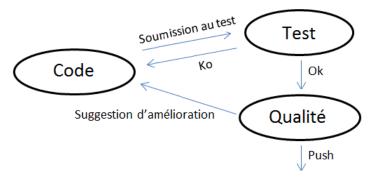


Figure II.1 - Schéma descriptif de la chaîne de développement

#### II.2 Development organisation

To secure our evolution we can use:

#### II.2.1 Usage of Scrum method

We will try to use Scrum method, which is actually widely used, and recognised for its effectiveness. At first, we will define a product backlog containing all desired functionalities in the final product. In fact, this report is also a part of product backlog. Next, we will divide the project into three sprints (which means iterations). A sprint backlog is defined for each sprint, including all we need to realise at the end of an iteration. Each sprint lasts two weeks and lies in improve the software incrementally, so that it is close to product backlog.

At the end of each *sprint*, we will organise a meeting, in order to review the progress and propose improvements or modifications of planning, but in the process of a *sprint*, we cannot modify the *sprint backlog*. At last, each day starts with a *scrum meeting*, on the meeting, each team member present his objective of the day and his actual difficulties.

#### II.2.2 Team repartition approach

We will use an approach inspired by the XP (extreme programming) method. In fact, we found it unnecessary that the team members work separately, and we found it excellent to work in pairs, in order to prevent errors and bias of the program structure, so that we can save times of testing and debugging. So four of us work in pairs and the last one works individually. The group will changes as the tasks are completed.

### II.3 Tasks organisation

#### II.3.1 Tasks definition

Sprint 1 backlog:

- AST parsing of procedure C++ code
- CFG conversion from parsed AST  $\,$
- Model checking with simple properties

\_

#### Sprint 2 backlog :

- AST parsing of object oriented C++ code
- CFG conversion from parsed AST  $\,$
- Model checking with simple criteria

\_\_

Improved CFG conversion from parsed AST; Model checking with complex criteria;

#### II.3.2 Planning

Task list	21-Jan	22-Jan	23-Jan	П	26-Jan	27-Jan	28-Jan	29-Jan	30-Jan	П	2-Feb	3-Feb	4-Feb	5-Feb	6-Feb	П	9-Feb	10 Fob	11-Feb	12 Fob	12 Eob			
	Z1-JdII	ZZ-JdII	25-JdII	11111	J			29-3411	30-Jan	111111	2-гев	3-гер	4-гер	3-reb	0-reb	11111	9-гев	10-reb	11-reb	12-reb	13-гев			
Redaction of specificaitons				-1111	ZDX	ZDX	ZDX			-1111						-1111								
				-1111	-					11111							1							
Training:				-1111	§					11111							<u> </u>							
- Scala	ZXFS	ZXFS	ZXFS		ZXFS	ZXFS	ZXFS	ZXFS	ZXFS															
- CTL and model checking	D	D	ZDXFS		ZDXFS	ZDXFS											<u> </u>							
					§												1							
Recurring tasks:				1111																				
- Design review				1111	ZDXF	ZDXF	ZDXF																	
- Code review				THE STATE OF						M							ZDXFS							
code renem					§											- iiii								
				-1111						Sept.						1111								
Sprint 1:					§																			
- Implementation Model Checking 1				-1111	§		Z F	Z F	Z F	11111	ZDF	ZDF	ZDF	Z D	Z D	11111	ZD	Z D	Z F					
- Implementation Parsing AST 1					§		DX	DX	DX		Х	Х	Х	X	Х		X	Х	Х					
- Implementation conversion AST to GFC 1				-99	\$					99	D S	DS	D S	D S	D S	-11111	DS	D S	D S					
- Validation:					£																			
- jeu de tests MC 1					į												1			Z F	Z F			
- jeu de tests AST/GFC 1				1111																DXS	DXS			
				1						Techen						1								
				+						+						+								
	+		+	+	16-Feb	17-Feb	18-Feb	19-Feb	20-Feb	+	23-Feb	24-Feb	25_Eoh	26-Feb	27 Eak	+								+
Recurring tasks:				1111	10-LED	17-160	10-LGD	12-160	20-760	min	23-760	24-160	23-FED	20-760	27-160	+								+
- Design review			1		ZDXFS	ZDXES			1						1	+								+
- Code review			-						1		ZDXFS					+								+
- Code review			-	-1111						-886	ZUAFS					++								
				1111						100						-								
Sprint 2:																								
- Implementation Model Checking 2					<u> </u>																			
- Implementation Parsing AST 2 (VF)					§																			
- Implementation conversion AST to GFC 2				1111																				
- Validation:																								
- jeu de tests MC 2					-											+								
- jeu de tests AST/GFC 2				1111	}					-886						++								
				-1111	}																			
- Validation du produit mergé																								
				Ш.						1						4								
Legend:				Ш.						1						4								
				Ш						1														
Member	_		Responsi	ibility																				
Zohour Abouakil	Project m	anagement	t																					
David Courtinot	Quality m	anagement	t																					
Xiaowen Ji		tion manag														+								
Fabien Sauce	Documen									+						+								
		tation								-						-								
Sofia Boutahar	Testing									++						+								
	-		-	+	-	-			1	+					1	+								
				$\perp$						1						$\perp$								
				$\perp$																				
				$\top$	2-Mar	3-Mar	4-Mar	5-Mar	6-Mar	$\top$	9-Mar	10-Mar	11-Mar	12-Mar	13-Mar	$\top$								
Recurring tasks:	1	1	1	dill	\$	<u> </u>	<u> </u>	·	<u> </u>	dilli		<u> </u>		1				1						+
- Design review			1		ZDXFS	7 D V 5 C			1							+								+
			-		ZUXFS	ZDXFS			-	dilli						+								+
- Code review				- Hill					-	IIII	ZDXFS					$\perp$								
								L	L	000														
Sprint 3:										OH)														
- Implementation Model Checking 3 (VF)																								
- Implementation conversion AST to GFC 3 (VF)				IIII	§					Hill														
- Validation:				1111	<b>.</b>					MI														
- Test MC 3					\$																			
- Test AST/GFC 3	+		-		<b>:</b>																			+
				-91111	1											$\vdash$								
- Validation of product	-		-			-										$\vdash$								+
				1111	§					9111														

# Partie III

# Risk management

Date	Risk description	Consequences	Type of risk	Probability (1-5)	Impact level (1-5)	Weight	Preventive mesure
27th, January 2015	Communication problems : lack of communication, misunderstanding, etc	Unproductive group, non-respect of the interfaces necessary to compatibility	Human resources	5	5	25	Be sure we agreed with our teammates before starting a part
27th, January 2016	Underestimation of the development time	Deadline exceeded / late delivery	Scope of project	4	5	20	Supervisor able to switch from one task to another and have a global vision
27th, January 2017	Wrong or unappropriate assumptions during the analysis	Unexpected edge cases difficult to handle with our model	Development method	5	4	20	Validate the conception by the client
27th, January 2018	Customer's requirements not respected	Product not accepted by the client	Client requirements	4	4	16	Having some meetings with the clients every weeks and making them validate our steps
27th, January 2019	Bad design choices at the beginning, issues to make the model evolve, corner cases	Problem to make the project evolve, waste of time to readapt the conception to the new requirements	Quality	3	5	15	Allocate several days to conception and ensure everyone is convinced by the design
27th, January 2020	Health problems : a member of the team getting sick, etc	In the best case, redefine the other team member role. Otherwise, the product will be late.	Scope of project	2	5	10	Flexible schedule
27th, January 2021	Underestimation of the learning curve, different time learning among the team	Delays, different rhythms for the various parts of the project	Scope of project	3	3	9	Create balanced teams (people better trained with people less trained)
27th, January 2022	Appearance of bugs that we cannot fix	Unable to meet certain requirements	Quality	2	4	8	Restart the task with another approachs and change the people affected to this task

### Partie IV

# Code management

### IV.1 Quality management

#### IV.1.1 Automated coding style checks

For ensuring that our coding rules are respected and evaluate the quality of our sources, we have used a tool called *Scalastyle* that enables, using an easy-to-use xml configuration file, to check some properties on a Scala code. Combined with a specific pulgin, this can be use to generate warnings or errors in the IDE the developer is using. Our settings can be found in appendix A.

### IV.2 Test strategy

### IV.3 Configuration management

# Partie V

# Appendices