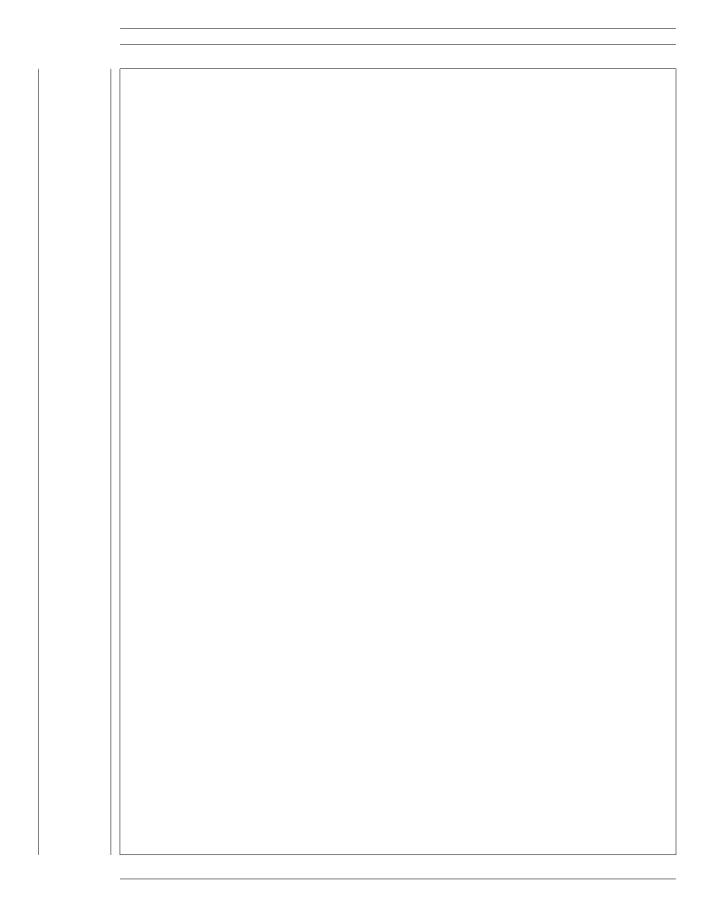
C	CNC	TENTS		
1	Intro	duction: comple	te overview	וי
2	Fou	ndations		3
	2.1			
	2.1	Developer and op	peration teams converge and both use software engineering practices	3
	2.1	2.1.1 Devops:	a culture that drives team efficiency and software quality (historical	
	2.1	2.1.1 Devops: evolution		
		2.1.1 Devops: evolution 2.1.2 Agile	a culture that drives team efficiency and software quality (historical n of the devops, loads of refs, how do i define devops)	3
	2.2	2.1.1 Devops: evolution 2.1.2 Agile Velocity and cycl	a culture that drives team efficiency and software quality (historical n of the devops, loads of refs, how do i define devops) etime as efficiency metrics of an agile team	3 3 4
		2.1.1 Devops: evolutio 2.1.2 Agile Velocity and cycl MTTR and Error	a culture that drives team efficiency and software quality (historical n of the devops, loads of refs, how do i define devops) etime as efficiency metrics of an agile team Rate measures the quality of a software	3
	2.2	2.1.1 Devops: evolution 2.1.2 Agile Velocity and cycl MTTR and Error 2.3.1 mainten	a culture that drives team efficiency and software quality (historical n of the devops, loads of refs, how do i define devops) etime as efficiency metrics of an agile team	3 3 4 4
	2.2	2.1.1 Devops: evolution 2.1.2 Agile Velocity and cycl MTTR and Error 2.3.1 mainten 2.3.1.1 b	a culture that drives team efficiency and software quality (historical n of the devops, loads of refs, how do i define devops) etime as efficiency metrics of an agile team Rate measures the quality of a software ance: (software lifecycle management)	3 4 4 4
	2.2	2.1.1 Devops: evolution 2.1.2 Agile Velocity and cycl MTTR and Error 2.3.1 mainten 2.3.1.1 to 2.3.1.2 i	a culture that drives team efficiency and software quality (historical n of the devops, loads of refs, how do i define devops) etime as efficiency metrics of an agile team Rate measures the quality of a software ance: (software lifecycle management) ougs + debugging	3 4 4 4 4
	2.2	2.1.1 Devops: evolution 2.1.2 Agile Velocity and cycl MTTR and Error 2.3.1 mainten 2.3.1.1 to 2.3.1.2 is 2.3.1.3 so 2.3.1.4 r	a culture that drives team efficiency and software quality (historical n of the devops, loads of refs, how do i define devops) etime as efficiency metrics of an agile team Rate measures the quality of a software ance: (software lifecycle management) ougs + debugging lities security monitoring	3 3 4 4 4 4 4 4 4
	2.2	2.1.1 Devops: evolution 2.1.2 Agile Velocity and cycl MTTR and Error 2.3.1 mainten 2.3.1.1 to 2.3.1.2 is 2.3.1.3 so 2.3.1.4 r	a culture that drives team efficiency and software quality (historical n of the devops, loads of refs, how do i define devops) etime as efficiency metrics of an agile team Rate measures the quality of a software ance: (software lifecycle management) ougs + debugging lities security	3 3 4 4 4 4 4 4
3	2.2 2.3	2.1.1 Devops: evolution 2.1.2 Agile Velocity and cycl MTTR and Error 2.3.1 mainten 2.3.1.1 to 2.3.1.2 is 2.3.1.3 so 2.3.1.4 r	a culture that drives team efficiency and software quality (historical n of the devops, loads of refs, how do i define devops) etime as efficiency metrics of an agile team Rate measures the quality of a software ance: (software lifecycle management) ougs + debugging lities security monitoring	3 3 4 4 4 4 4 4 4
3	2.2 2.3	2.1.1 Devops: evolution 2.1.2 Agile Velocity and cycle MTTR and Error 2.3.1 mainten 2.3.1.2 i 2.3.1.2 i 2.3.1.3 s 2.3.1.4 r 2.3.2 the apath	a culture that drives team efficiency and software quality (historical n of the devops, loads of refs, how do i define devops) etime as efficiency metrics of an agile team Rate measures the quality of a software ance: (software lifecycle management) ougs + debugging lities security monitoring	3 3 4 4 4 4 4 4 4 4 5
3	2.2 2.3	2.1.1 Devops: evolution 2.1.2 Agile Velocity and cycl MTTR and Error 2.3.1 mainten 2.3.1.1 to 2.3.1.2 i 2.3.1.3 s 2.3.1.4 r 2.3.2 the apath tions reducing mainten	a culture that drives team efficiency and software quality (historical in of the devops, loads of refs, how do i define devops) etime as efficiency metrics of an agile team Rate measures the quality of a software ance: (software lifecycle management) bugs + debugging lities security monitoring hic agile devops team	3 3 4 4 4 4 4 4 4 4 5
3	2.2 2.3 Solu 3.1	2.1.1 Devops: evolution 2.1.2 Agile Velocity and cycl MTTR and Error 2.3.1 mainten 2.3.1.2 i 2.3.1.2 i 2.3.1.3 s 2.3.1.4 r 2.3.2 the apath tions reducing mainten Automated post of the deployer	a culture that drives team efficiency and software quality (historical n of the devops, loads of refs, how do i define devops) etime as efficiency metrics of an agile team Rate measures the quality of a software ance: (software lifecycle management) ougs + debugging lities recurity monitoring nic agile devops team	3 3 4 4 4 4 4 4 4 4 4

ii CONTENTS

4	Eval	luation	7
	4.1	GapFish a startup company	7
	4.2	GapFish's traditional toolchain and teams	7
	4.3	team agility metrics	7
	4.4	software quality metrics	7 7 7
	4.5	Results	7
	4.6	Lessons learned and future	7
5		clusion	9
	5.1	theoretical/practical conclusion	9
	5.2	for Gapfish	9

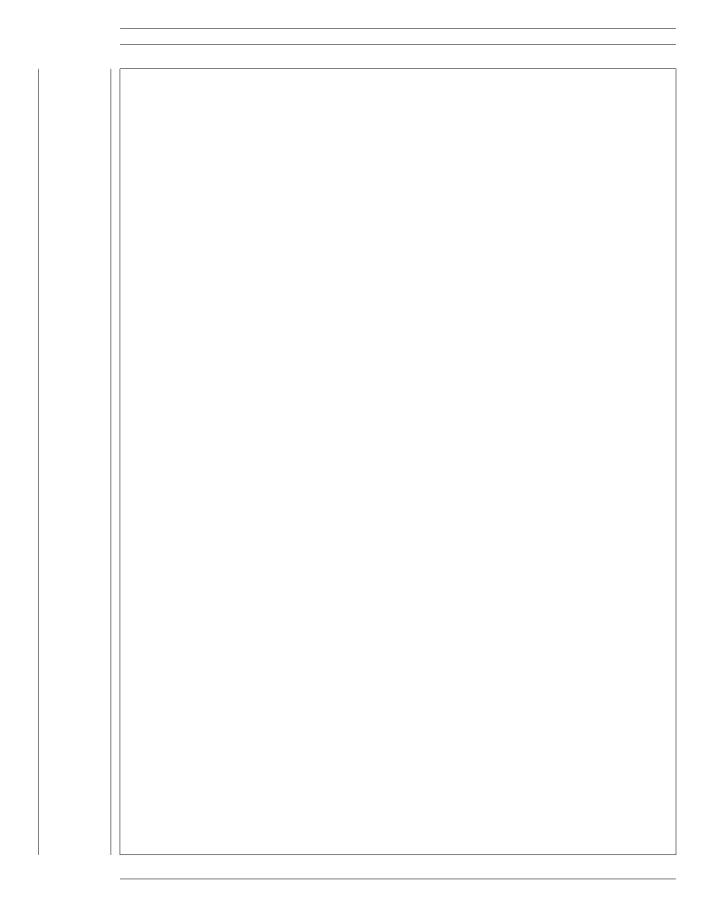
CHAPTER 1
INTRODUCTION: COMPLETE OVERVIEW



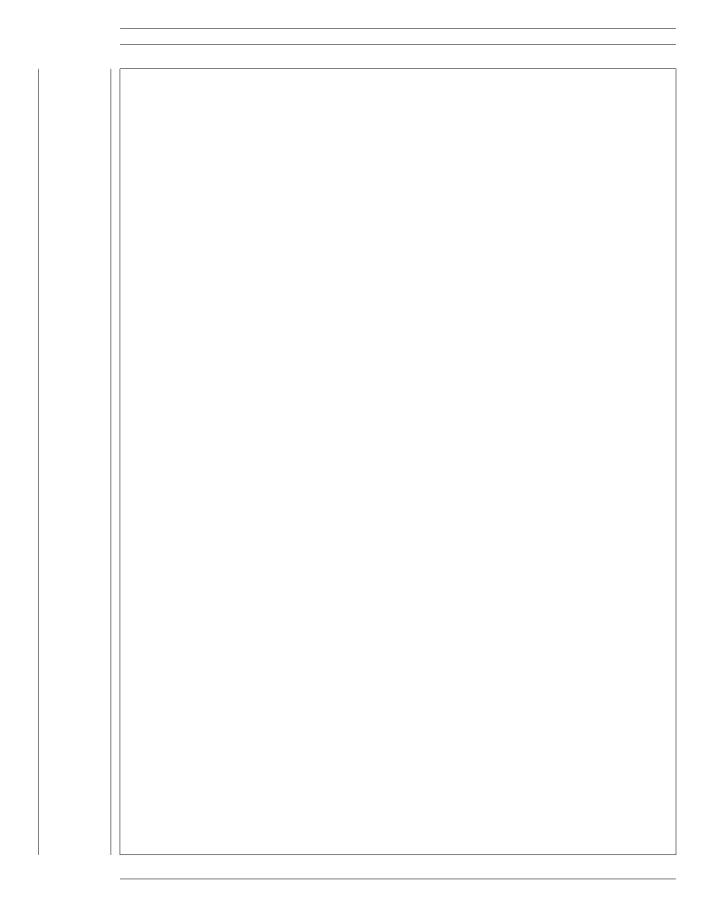
CHAPTER 2
FOUNDATIONS
2.1 Developer and operation teams converge and both use software engineering practices
2.1.1 Devops: a culture that drives team efficiency and software quality (historical evolution of the devops, loads of refs, how do i define devops)
subsubsectionOperation uses typical developer methods subsubsectionToolchains of operation and developers subsubsectionIntegration of developer and operation teams subsubsectionCluster management systems automate developer operations
2.1.2 Agile
subsubsectionContinuous Delivery Pipeline
overview: commitment, continuous integration/testing, deployment
The evolution of deployments/continuous deployment

4	FOUNDATIONS
2.2	Velocity and cycletime as efficiency metrics of an agile team
2.3	MTTR and ErrorRate measures the quality of a software
2.3.	1 maintenance: (software lifecycle management)
2.3.1	1.1 bugs + debugging
2.3.1	1.2 ilities
2.3.1	1.3 security
2.3.1	1.4 monitoring
2.3.2	2 the apathic agile devops team

CHAPTER 3	
SOLUTIONS	
3.1 reducing maintenance with continuous production testing	
3.2 Automated post deployment testing	
3.3 the deployer	
3.3.1 architecture and how it integrates in the pipeline	
subsectionimportance for maintenance and feature deploys subsectioncycle time vs. notifications	



CHAPTER 4	
EVALUATION	
4.1 GapFish a startup company	
4.2 GapFish's traditional toolchain and teams	
4.3 team agility metrics	
subsectioncycle time measures quality of delivery engine subsectionlocs/deploy measures risk subsectiondeploys per day measures agility	
4.4 software quality metrics	
subsectionErrorRate as monitoring measure for automation subsectionproblems in error rate measure defect and failure subsectionsolution a secific heuristic	
4.5 Results	
subsectiontraditional vs. new	
4.6 Lessons learned and future	



CH	APTER 5
CC	ONCLUSION
5.1	theoretical/practical conclusion
5.2	for Gapfish