서강대학교
SOGANG UNIVERSITY

오픈소스를 이용한 SW 개발실습

부제: DevOps 실습

SuJin Choi, PhD
Sogang University
Email: sujinchoi@sogang.ac.kr

Wall Street Journal on August 20, 2011.

Amazon,
Neflix,
Apple iTunes,
Disney,
Google,
Skype,
Linkedin

https://a16z.com/2011/08/20/why-software-is-eating-the-world/

시가총액 Top10 기업

2010년말

순위	기업명	시가총액 (억 달러)	Industry	국가
1	ExxonMobil	3,682	Energy	미국
2	PetroChina	3,291	Energy	중국
3	Apple	2,952	**Information Technology**	미국
4	Microsoft	2,385	**Information Technology**	미국
5	Walmart	1,934	Retail	미국
6	General Electric	1,922	Conglomerate	미국
7	Procter & Gamble	1,889	Consumer Goods	미국
8	Industrial and Commercial Bank of China	1,739	Financials	중국
9	China Mobile	1,968	**Telecommunications**	중국
10	Berkshire Hathaway	1,997	Financials	미국

2024년8월

순위	기업명	시가총액 (억 달러)	Industry	국가
1	Apple	34,000	**Information Technology**	미국
2	NVIDIA	32,000	**Information Technology**	미국
3	Microsoft	31,000	**Information Technology**	미국
4	Alphabet	21,000	**Information Technology**	미국
5	Amazon	19,000	**Consumer Discretionary**	미국
6	Saudi Aramco	18,000	Energy	사우디아라비아
7	Meta Platforms	13,000	**Information Technology**	미국
8	Berkshire Hathaway	9,670	Financials	미국
9	TSMC	9,100	**Information Technology**	대만
10	Eli Lilly	8,300	Healthcare	미국

출처:perflexity

UX DX

VS.

Gartner

How to Use Generative AI to Boost Developer Productivity

Use GenAI tools to improve developer satisfaction, collaboration and flow, to maximize gains.

GitHub의 설문 조사에 따르면, **GitHub Copilot**을 사용한 개발자의 60% 이상이 만족도와 웰빙 수준이 향상되었다고 보고했습니다.

...

개발자들이 즐기는 작업에 적용될 경우 개발자 만족도를 실제로 낮출 수 있습니다. 개발자들은 흥미로운 문제를 해결할 기회에서 동기를 얻습니다. 만약 GenAI가 코딩의 창의적인 측면을 대신하게 된다면, 개발자들은 덜 행복해할 것입니다. 소프트웨어 엔지니어링 리더들은 **개발자 만족도와 웰빙을 개선**하는 방식으로 **새로운 GenAI 도구**를 적용해야 합니다.

...

컨텍스트 스위칭(context switching)은 인지적 피로의 한 원인으로, 개발자가 현재 작업이 차단되거나 방해를 받아 다른 작업으로 전환될 때 발생합니다. GenAI는 개발자가 작업 간에 컨텍스트를 전환해야 하는 상황을 **최소화**하거나 심지어 없앨 수 있습니다.

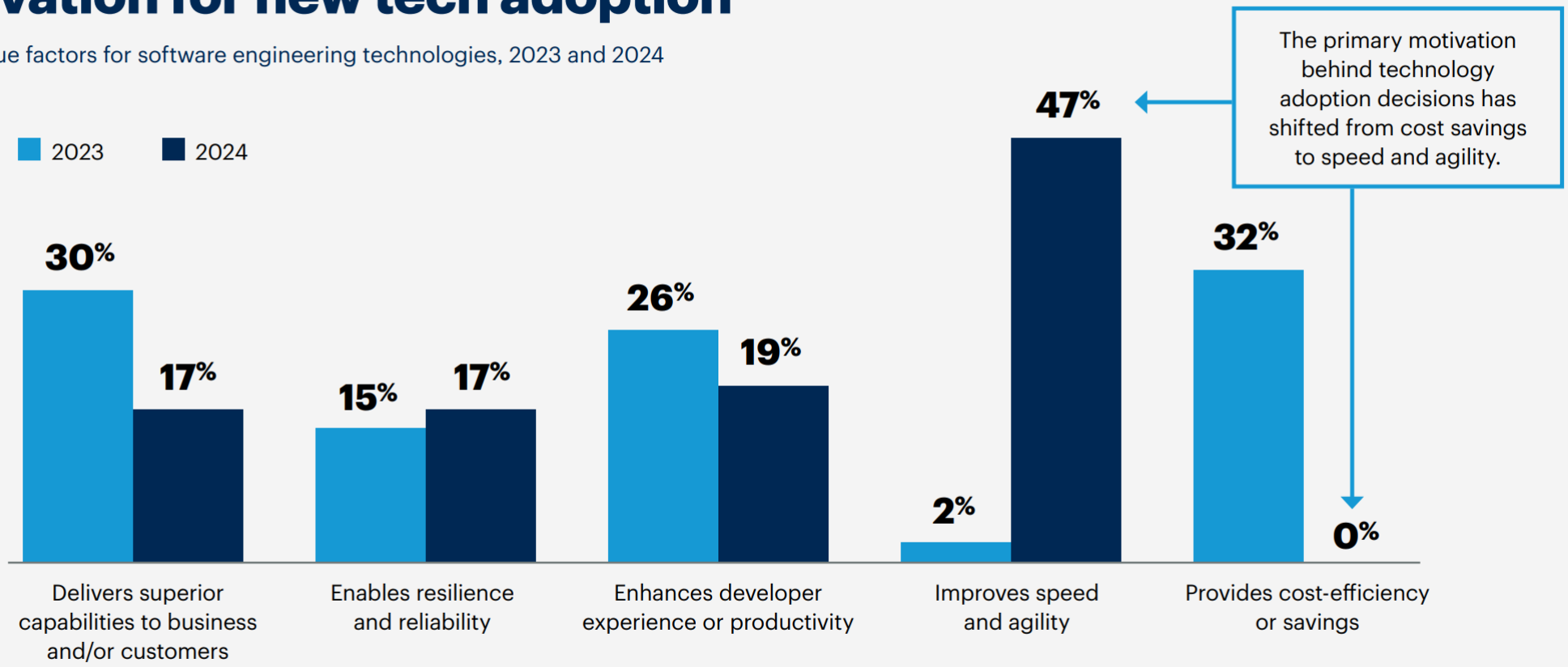
Gartner

Figure 1: The Gartner Top Strategic Technology Trends for Software Engineering for 2024

Gartner

Improving speed and agility is the top motivation for new tech adoption

Primary value factors for software engineering technologies, 2023 and 2024



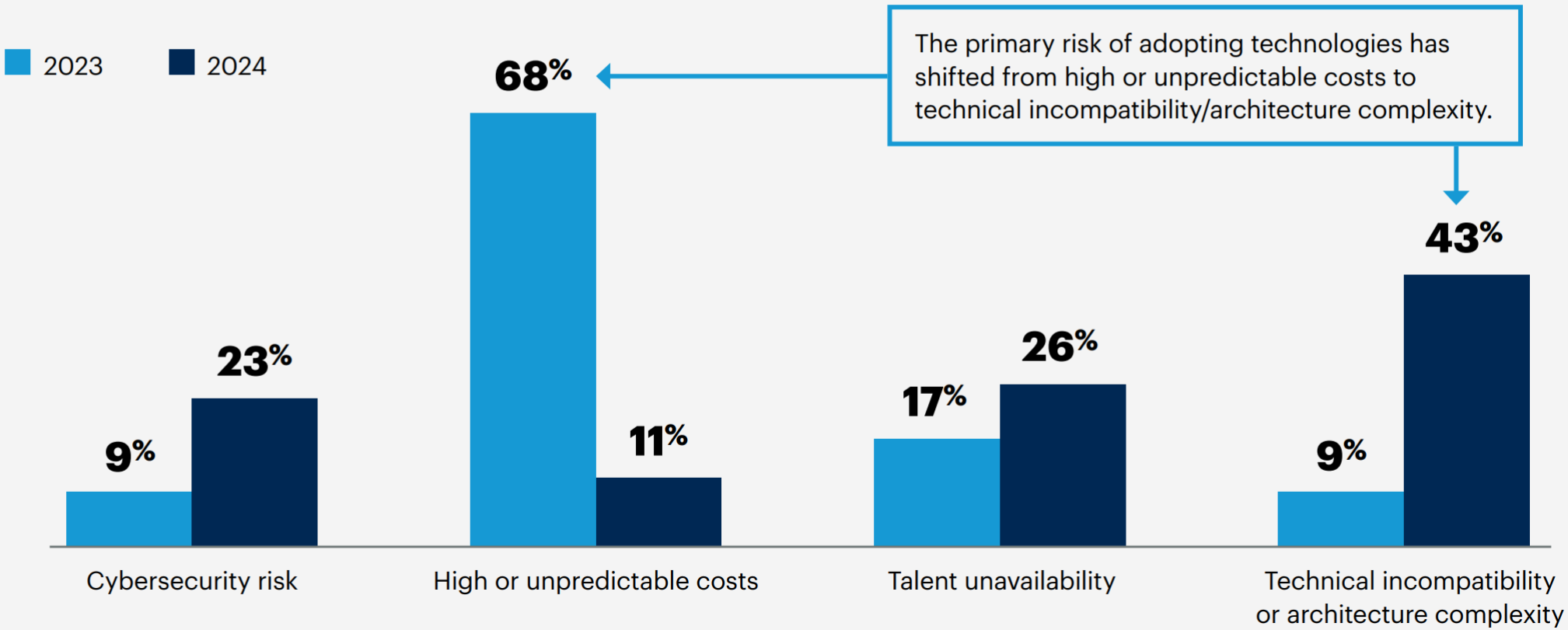
n (software engineering leaders from global organizations) = 147; n (technologies) = 47
Source: Gartner

Gartner, Software Engineering Technology Roadmap, 2024

Gartner

Technical incompatibility/architecture complexity is the primary risk factor

Primary risk factors for software engineering, 2023 and 2024

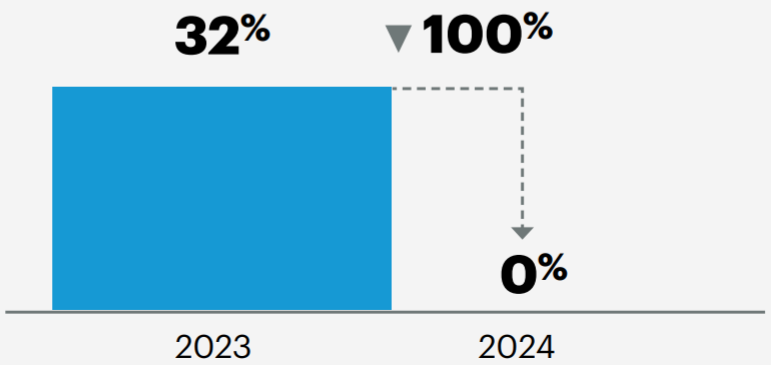


Gartner, Software Engineering Technology Roadmap, 2024

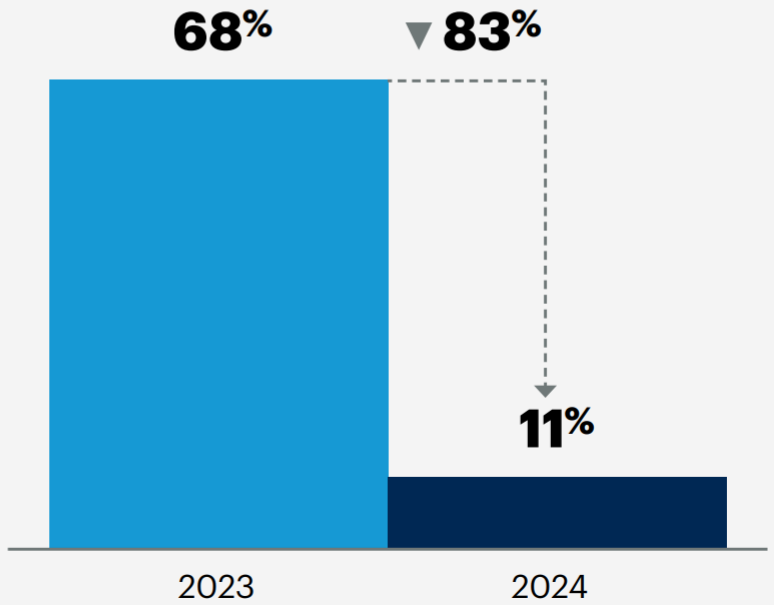
Gartner

Cost is no longer a primary driver

Percent of respondents indicating **cost-efficiency or savings** as a primary value factor



Percent of respondents indicating **high or unpredictable costs** as a primary risk factor



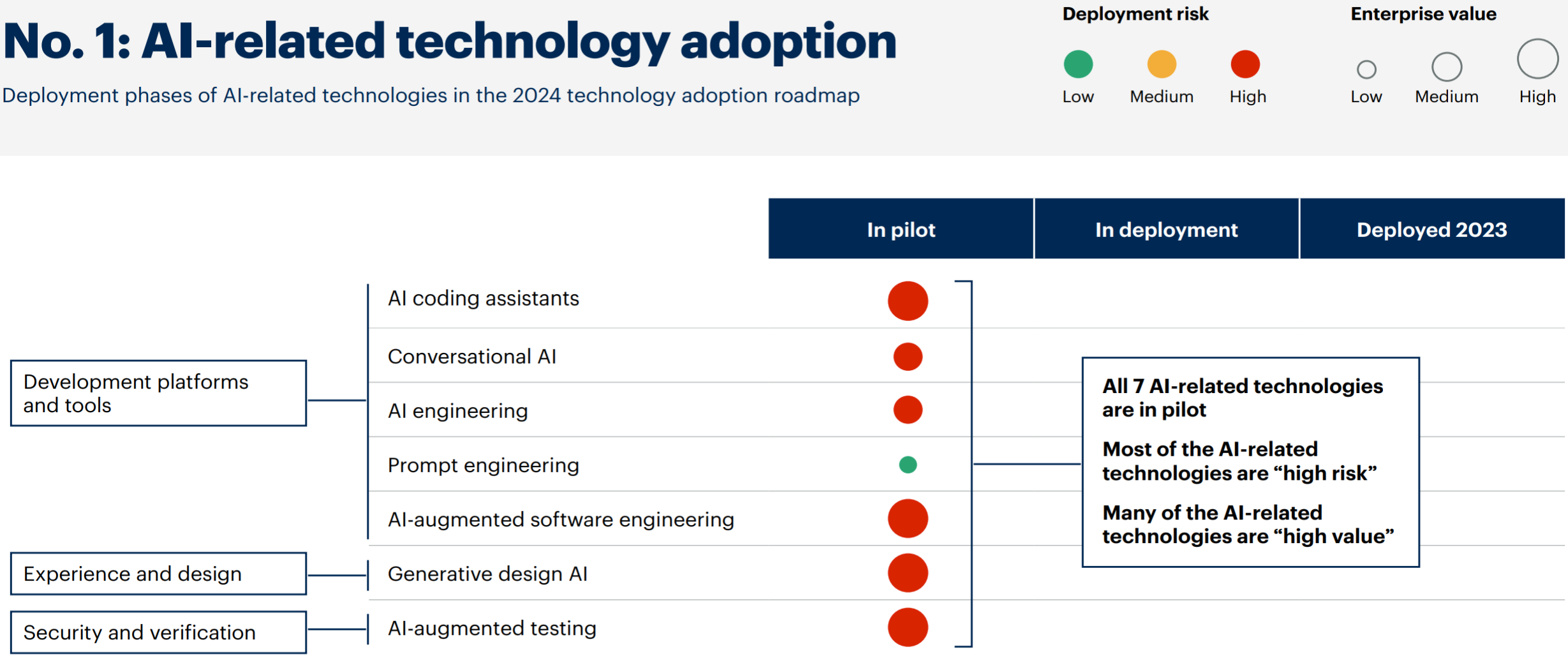
Gartner, Software Engineering Technology Roadmap, 2024

Gartner

2024 Key Technology Trends

No. 1: AI-related technology adoption

Deployment phases of AI-related technologies in the 2024 technology adoption roadmap



Gartner, Software Engineering Technology Roadmap, 2024

Gartner

2024 Key Technology Trends

No. 2: Developer experience as a critical priority

▲56%

There was a 56% increase in expert inquiries on the topic of developer experience (DX), from 2022 to 2023.

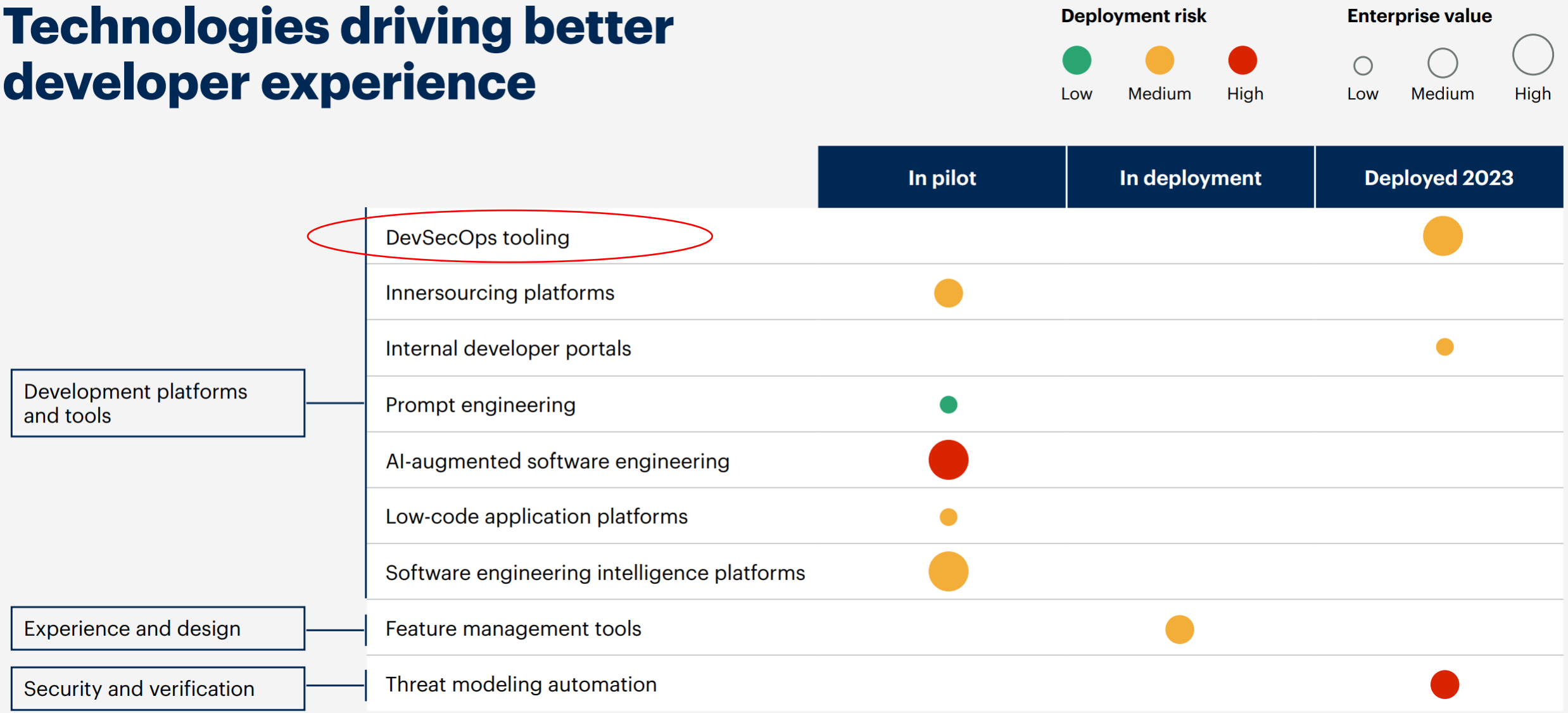
89%

of organizations are actively working to improve DX.

Gartner, Software Engineering Technology Roadmap, 2024

Gartner

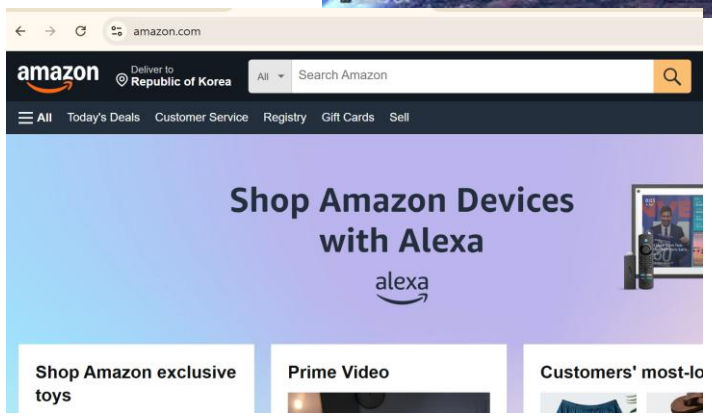
Technologies driving better developer experience



Gartner, Software Engineering Technology Roadmap, 2024

Life Cycle

image by Three musketeers from https://www.flaticon.com/



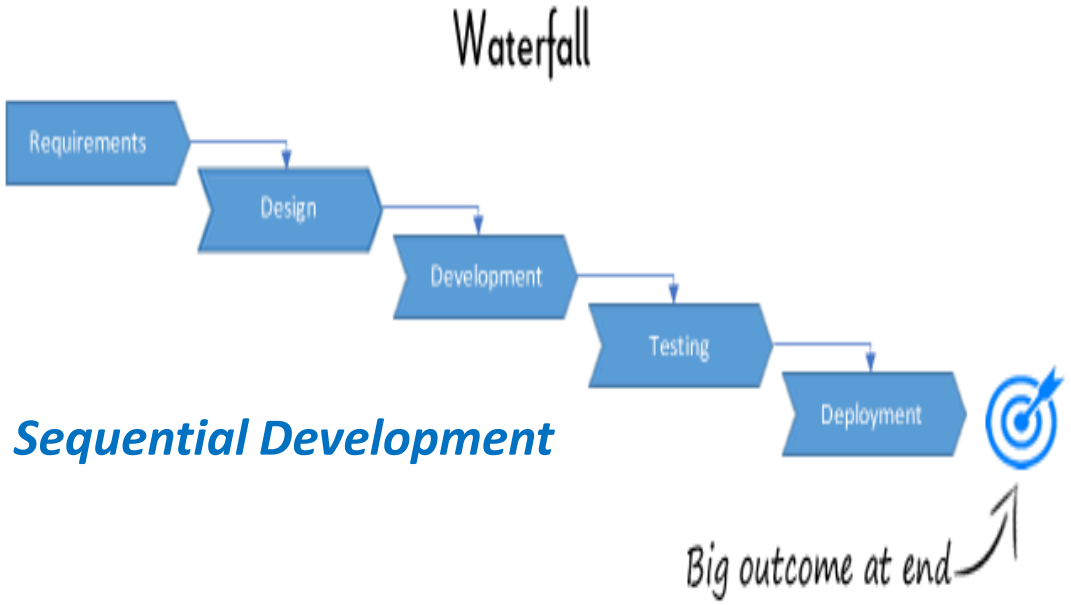
Software Life Cycle

- Concept exploration,
- Development
- Sustainment,
- Retirement

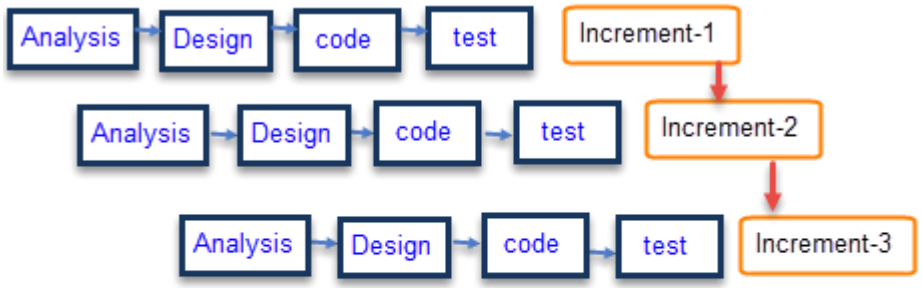
- Requirement Analysis
- Design
- Coding
- Testing

Traditional Software Life Cycle Models

- Waterfall Model, 1970, Winston W. Royce
- Prototyping Model
- Incremental Development Model, 1976, Harlan Mills



Sequential Development



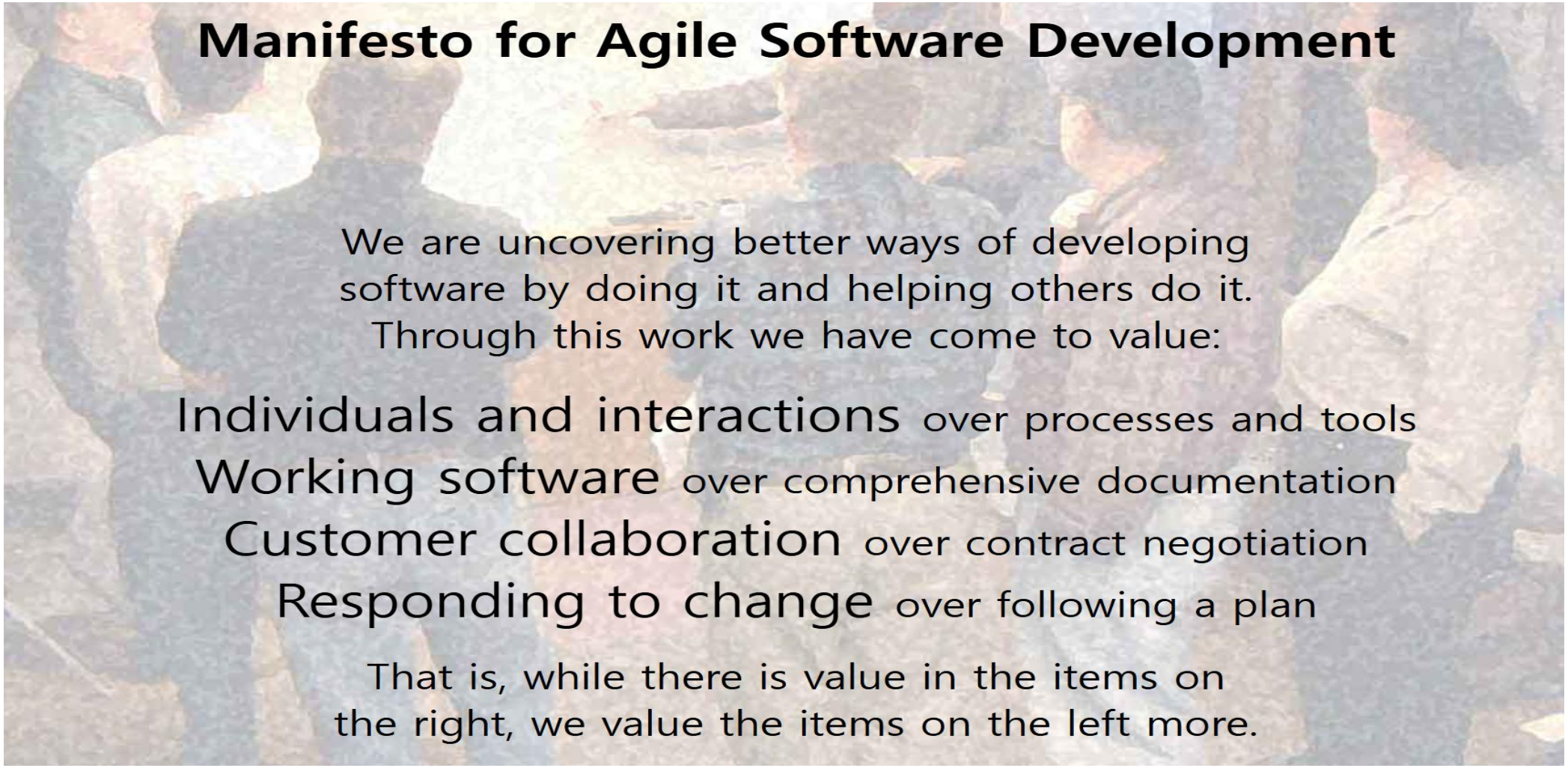
Incremental Model

https://www.guru99.com/

기존 모델의 한계

- 비즈니스 가치의 실현 시기가 …
- 재작업의 한계와 비용 이슈
- 고객의 니즈를 만족시키는가 …
- 요구사항의 변경은 …

Agile의 등장

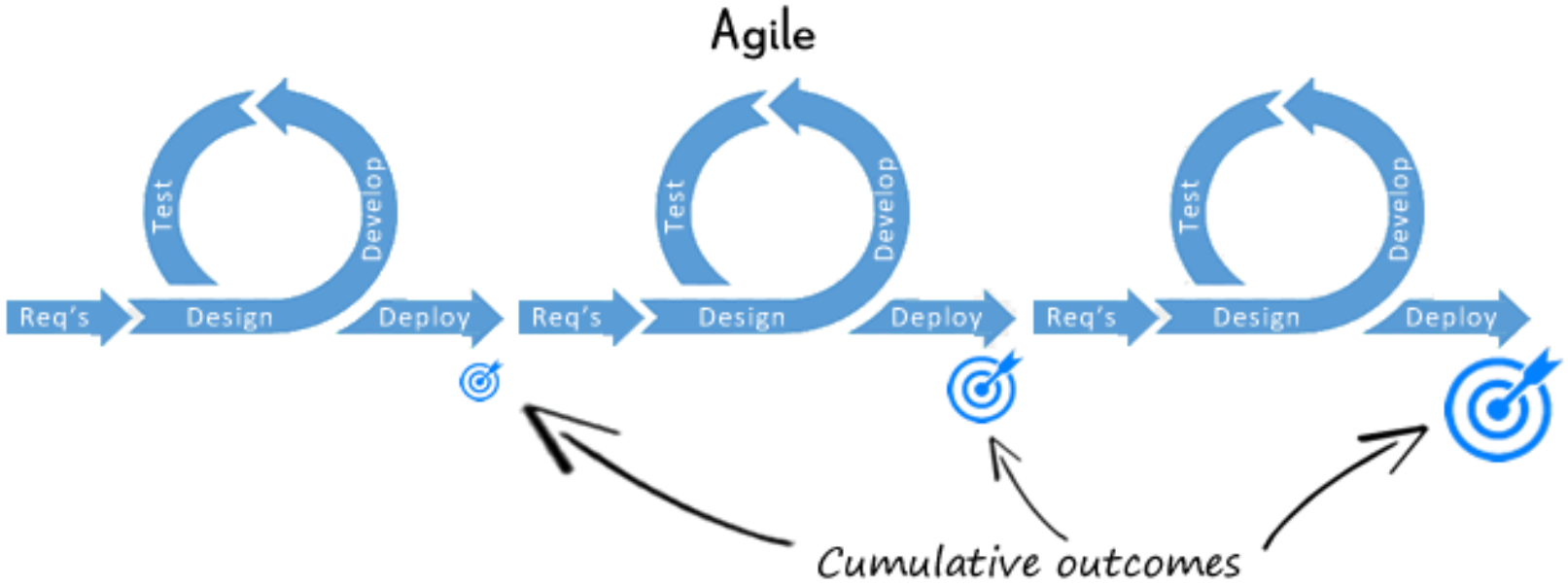


https://agilemanifesto.org/

Agile Software Development

- Iterative, Incremental, and Evolutionary approach
  - **Short cycle : early, concrete, and continuing feedback**
  - Incremental planning approach
  - Flexibly schedule the implementation, **responding to changing business needs**

Iterative and Incremental Development



Agile Methodology

- Scrum, 1995, Ken Schwaber and Jeff Sutherland
- eXtreme Programming(XP), 1996, Kent Beck
- Lean Software Development, 2003, Mary and Tom Poppendieck
- Kanban (for software), 2007, David J. Anderson

- Agile Manifesto, 2001

그러나, 현실은...

- 따로 국밥 ...

https://medium.com/@yevgeniy.zhitomirskiy/what-is-devops-9a1acc0c2a28

Digital 시대

소프트웨어 역할의 변화

“Software Service의 Quality, Speed, Availability, Security 가 비즈니스 성공의 핵심”

기술의 발전

클라우드 컴퓨팅 발전

협업과 공유의 문화

오픈소스 소프트웨어(OSS) 발전

DevOps

- Development + Operations
- DevOpsDay, 2009, Patric Debbois

How would you define DevOps in the shortest most concise definition possible?

I tried to summarize this once in a tweet: **DevOps is about removing the friction between silos**. All the rest is engineering.

Q&A: Patrick Debois on the Past, Present and Future of DevOps, Oct 24th, 2023 10:07am by Jennifer Riggins, https://thenewstack.io/

DevOps

- Silo
  - 원래 곡식과 목초를 쌓아두는 굴뚝 모양의 창고를 뜻함.

" 비유적으로 **조직 내 부서나 팀이 서로 단절되고 독립적으로 운영되는 현상** "

Image by rgaudet17 from Pixabay

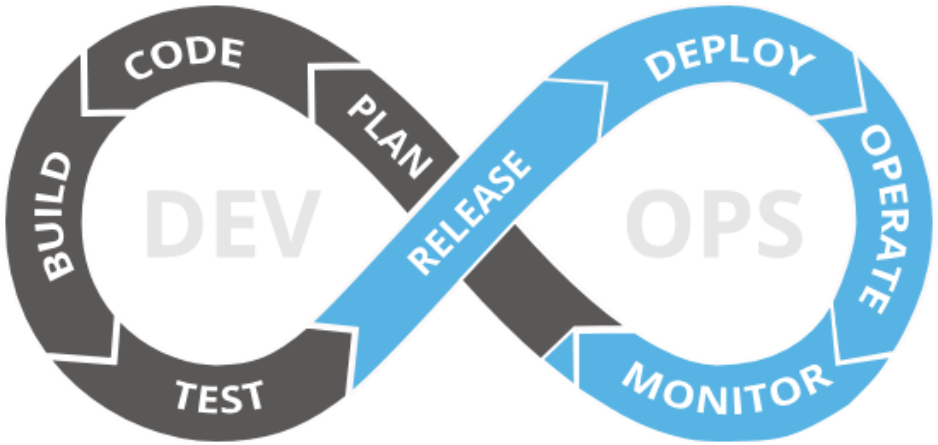
DevOps

- ***DevOps is a software engineering culture and practice that aims at unifying software development (Dev) and software operation (Ops).***

[출처: 미국 공군의 소프트웨어 개발 및 배포 플랫폼인 Platform One 홈페이지, https://software.af.mil/resources/devops/]

The main characteristic of the DevOps movement is to strongly advocate automation and monitoring at all steps of software development , from integration , testing , releasing to deployment and infrastructure management . DevOps aims at shorter development cycles, continuous delivery , and more dependable releases, in close alignment with business objectives.

DevOps is NOT ENOUGH! DevSecOps is what must be implemented with the cybersecurity stack built into the DevOps pipeline.



DevOps

- **ISO/IEC/IEEE 32675:2022 Information technology — DevOps — Building reliable and secure systems including application build, package and deployment**

- **소프트웨어 및 시스템 제품과 서비스의 명세, 개발, 운영을 목적으로 관련 이해관계자 간의 더 나은 소통과 협업을 가능하게 하고, 생명 주기 전반에서 모든 측면의 지속적인 개선을 지원하는 원칙과 실천의 집합** [ISO/IEC/IEEE 32675:2022]
  Set of principles and practices which enable better communication and collaboration between relevant stakeholders for the purpose of specifying, developing, and operating software and systems products and services, and continuous improvements in all aspects of the life cycle.

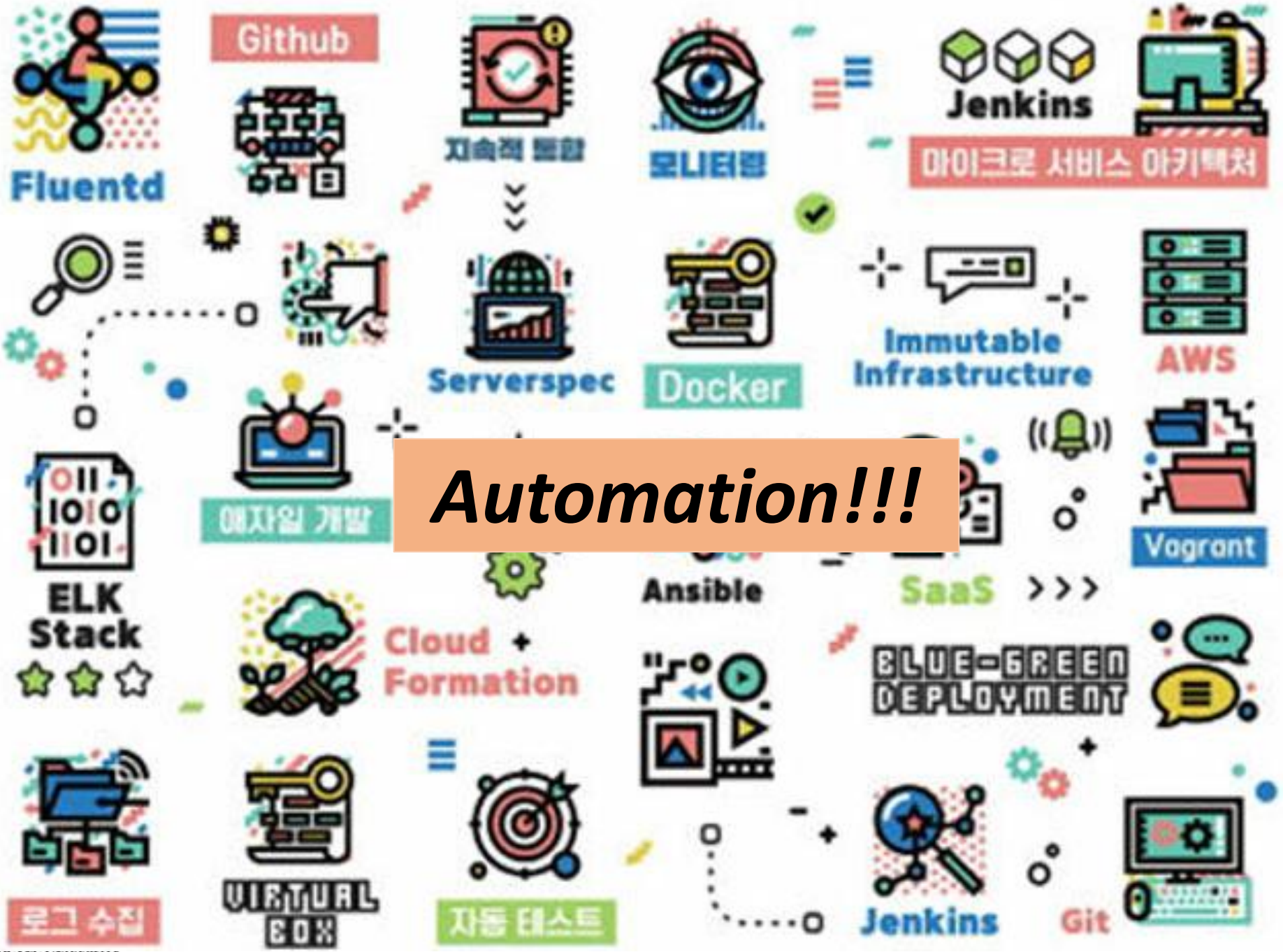
Why DevOps?

- US Air Force's Digital Journey

- 날짜: 2015년 10월 3일
- 장소: 아프가니스탄 쿤두즈
- 목표: 탈레반이 점령한 것으로 의심된 통신시설 건물
- 피해:
- 국경없는 의사회가 운영하는 병원 건물 완전 파괴
- 42명 사망 (환자, 의료진 포함), 수십 명 부상

- 원인: AC-130 건쉽(Gunship)**이 공격을 수행할 때 사용한 타겟팅 시스템에서 소프트웨어 오류

https://www.youtube.com/watch?v=oifWkFtmm_o&list=PLttCjInCgY7URq_Aq3kBxg1aHFsg8rP-Z&index=2


Github
Fluentd
지속적 통합
모니터링
Jenkins
마이크로 서비스 아키텍처
Serverspec
Docker
Immutable Infrastructure
AWS
애자일 개발
Automation!!!
Vagrant
ELK Stack
Ansible
SaaS
Cloud Formation
BLUE-GREEN DEPLOYMENT
로그 수집
VIRTUAL BOX
자동 테스트
Jenkins
Git


IT 운용 체제 변화를 위한
데브옵스 DevOps

DevOps 성능평가: DORA 메트릭

* DORA: DevOps Research and Assessment

1. **배포 빈도(Deployment Frequency)**:
   운영환경/End user에게 얼마나 자주 코드를 배포하는지
2. **변경 리드 타임(Lead Time for Changes)**:
   코드 commit이 운영환경에 반영되기까지 걸리는 시간
3. **변경 실패율(Change Failure Rate)**:
   배포된 코드 변경 중 실패한 비율.
4. **서비스 복구 시간(Time to Restore Service)**:
   서비스 장애 발생 시 복구하는 데 걸리는 시간
5. **재작업율 (Rework rate):**
   전체 배포 중 결함으로 인한 배포의 비율. 2024년 추가

Accelerate State of DevOps, Google Cloud

DevOps 성능평가: DORA 메트릭 (2024 실적)

Performance level	Change lead time	Deployment frequency	Change fail rate	Failed deployment recovery time	Percentage of respondents*
Elite	Less than one day	On demand (multiple deploys per day)	5%	Less than one hour	19% (18-20%)
High	Between one day and one week	Between once per day and once per week	20%	Less than one day	22% (21-23%)
Medium	Between one week and one month	Between once per week and once per month	10%	Less than one day	35% (33-36%)
Low	Between one month and six months	Between once per month and once every six months	40%	Between one week and one month	25% (23-26%)

2024 Accelerate State of DevOps, Google Cloud

Software delivery performance (DORA metrics)

- When compared to low performers, elite performers realize

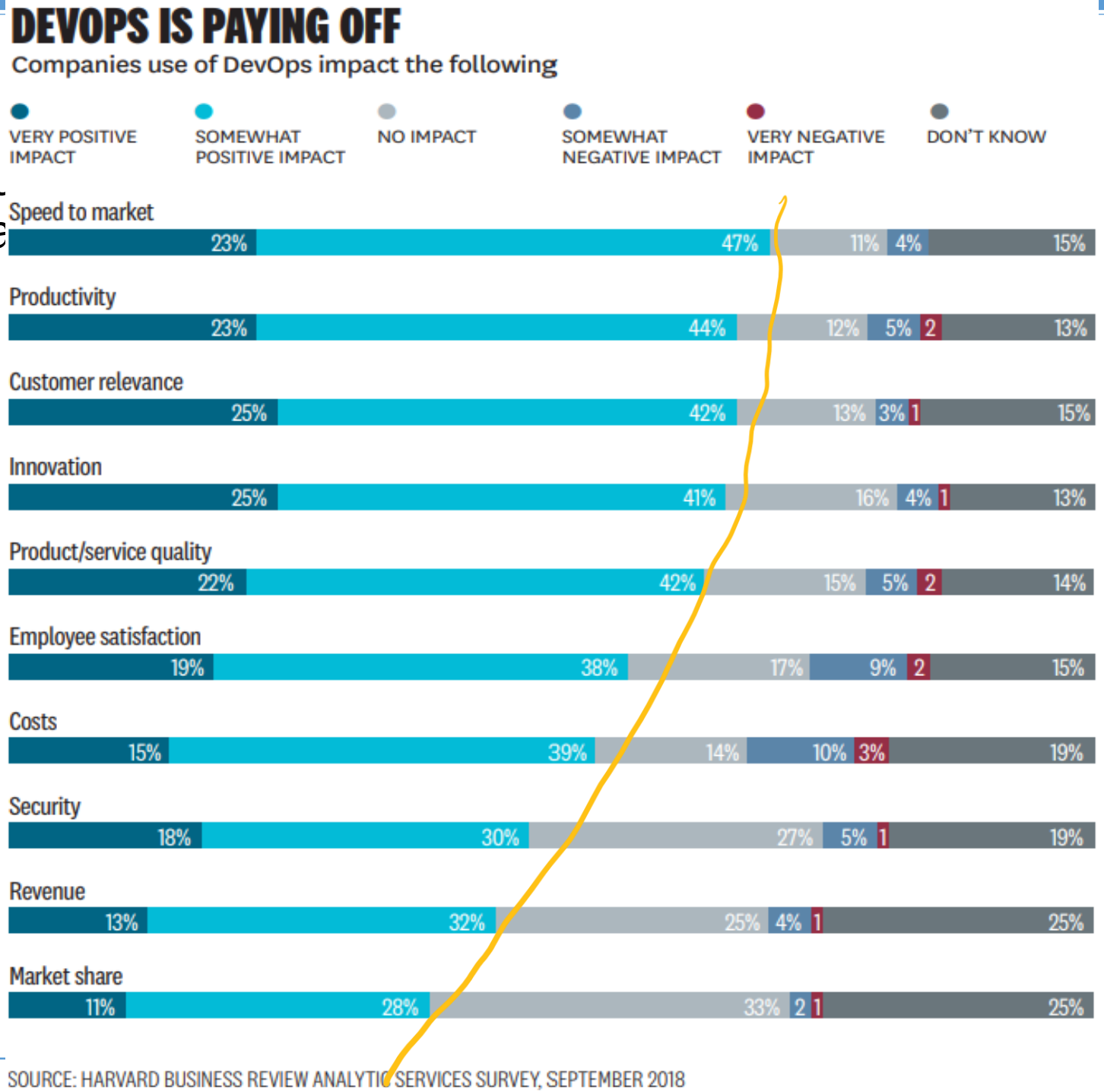
2024 Accelerate State of DevOps, Google Cloud

ROI of DevOps

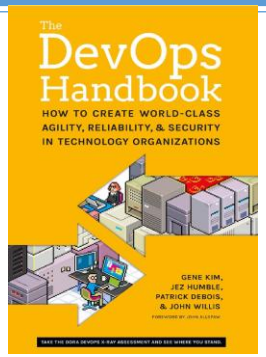
- Competitive Advantage throu
  Harvard Business Review Ana

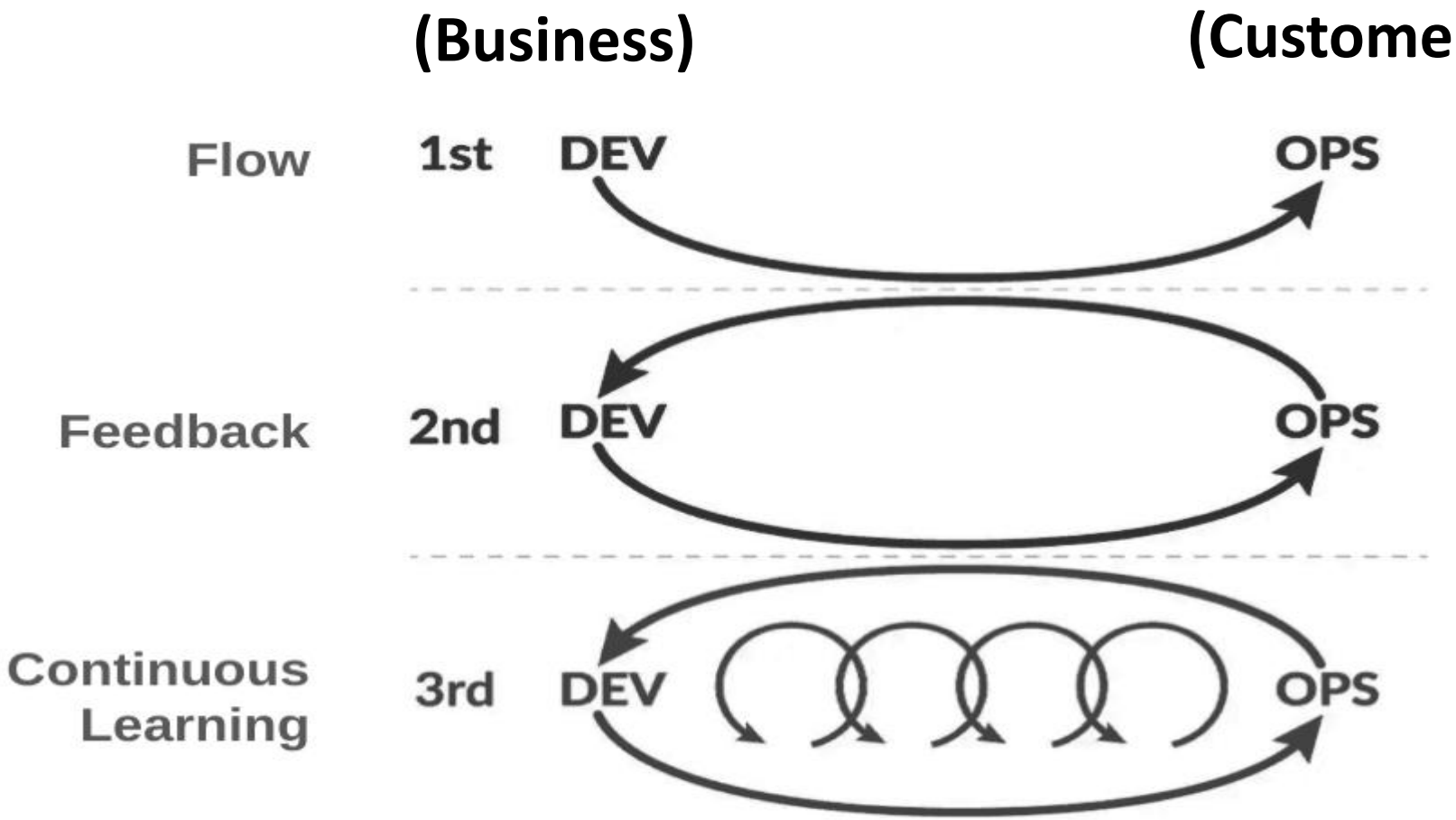
A total of 654 respondents

SIZE OF ORGANIZATION			
59%	30%	10%	0%
10,000 OR MORE EMPLOYEES	1,000-9,999 EMPLOYEES	500-999 EMPLOYEES	499 AND FEWER EMPLOYEES



Three Ways of DevOps:
The Principles Underpinning DevOps





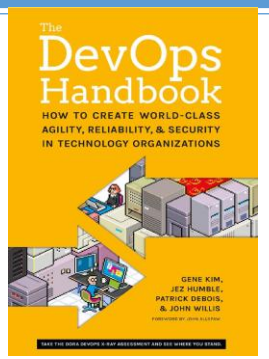
- emphasizes the performance of the entire system
- focus is on all business value streams

- shorten and amplify feedback loops so necessary corrections can be continually made

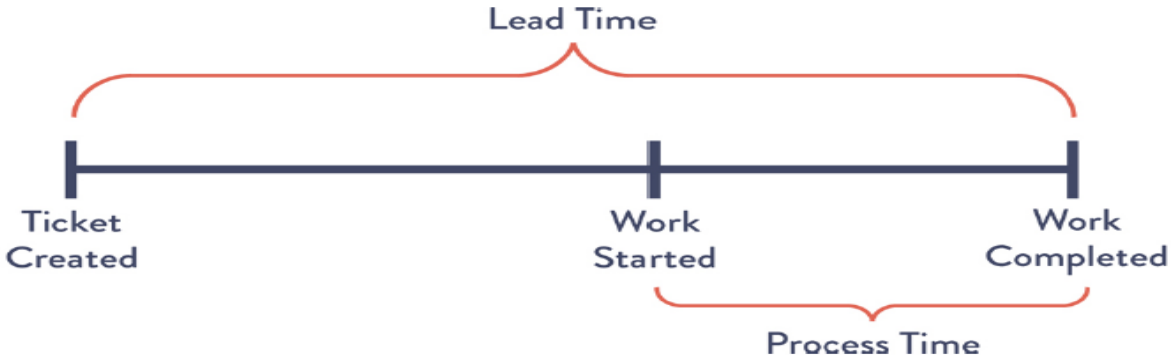
- continual experimentation
- taking risks and learning from failure

출처: Gene Kim, https://itrevolution.com/articles/the-three-ways-principles-underpinning-devops/

Deployment Lead Time



- Lead Time vs. Process Time



배포리드타임 3개월

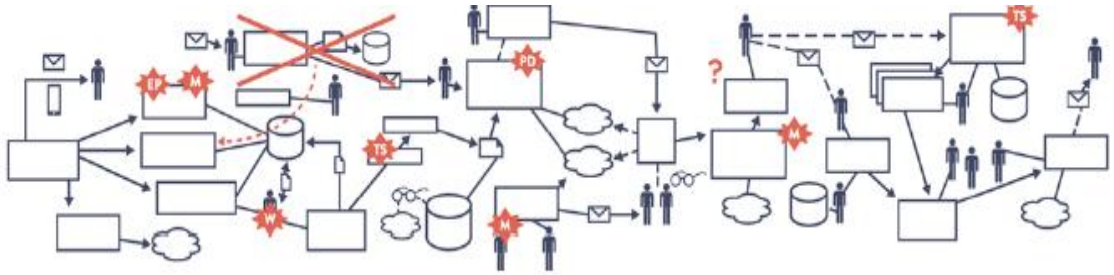


Figure 1.2: A Technology Value Stream with a Deployment Lead Time of Three Months

배포리드타임 25분

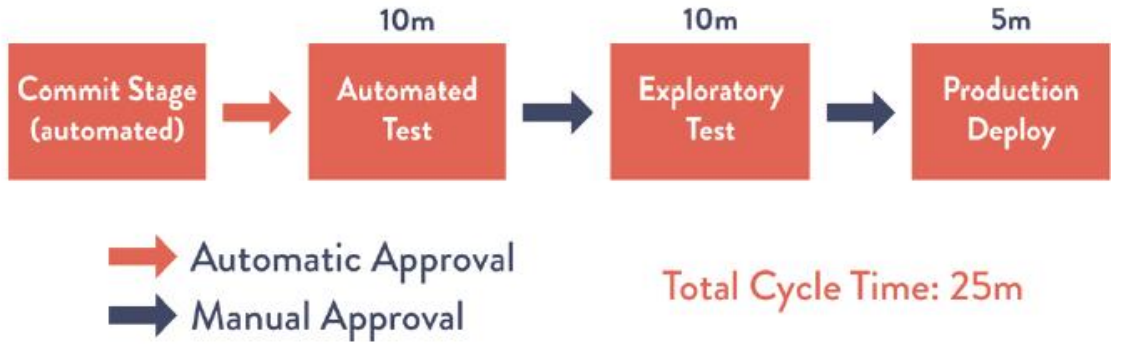
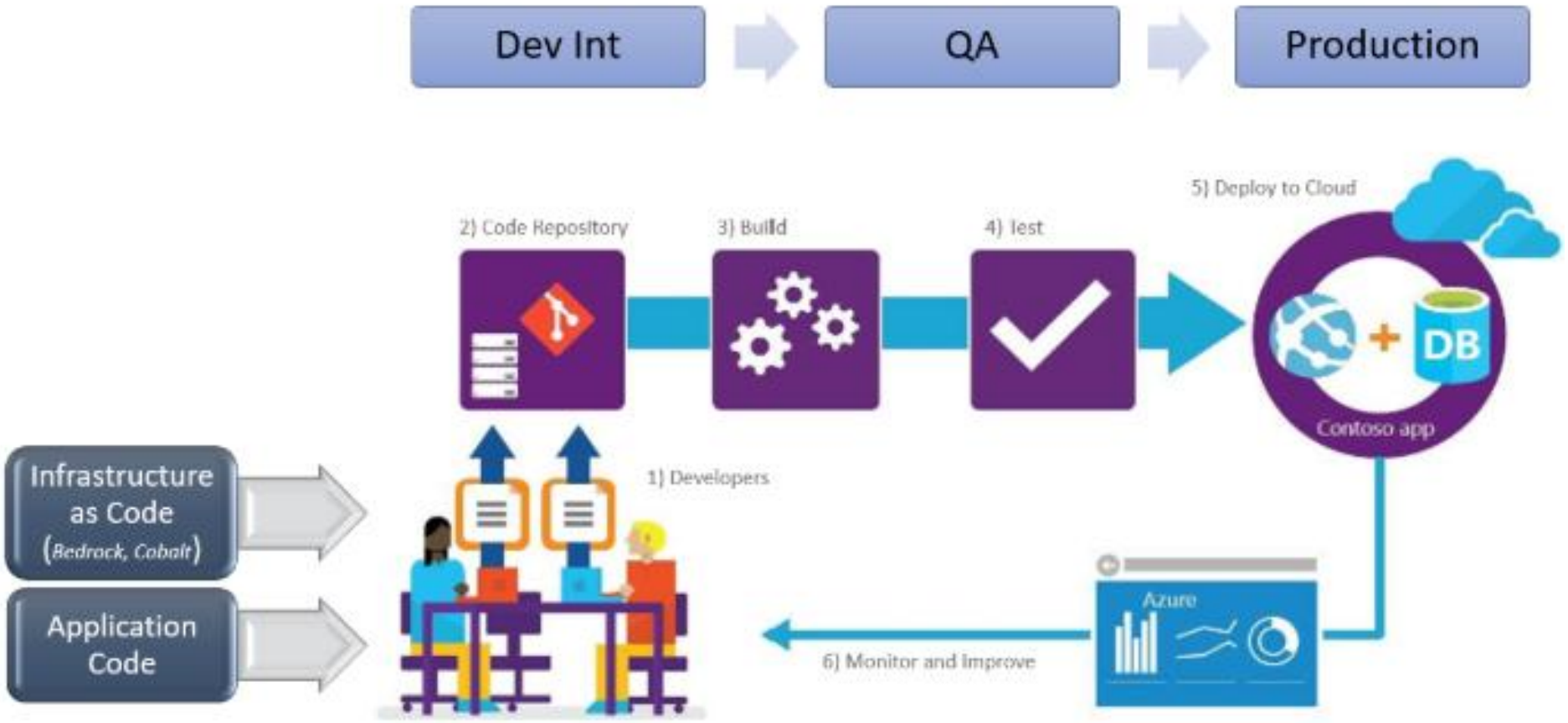


Figure 1.3: A Technology Value Stream with a Lead Time of Minutes

DevOps Pipeline



https://microsoft.github.io/code-with-engineering-playbook/

DevSecOps

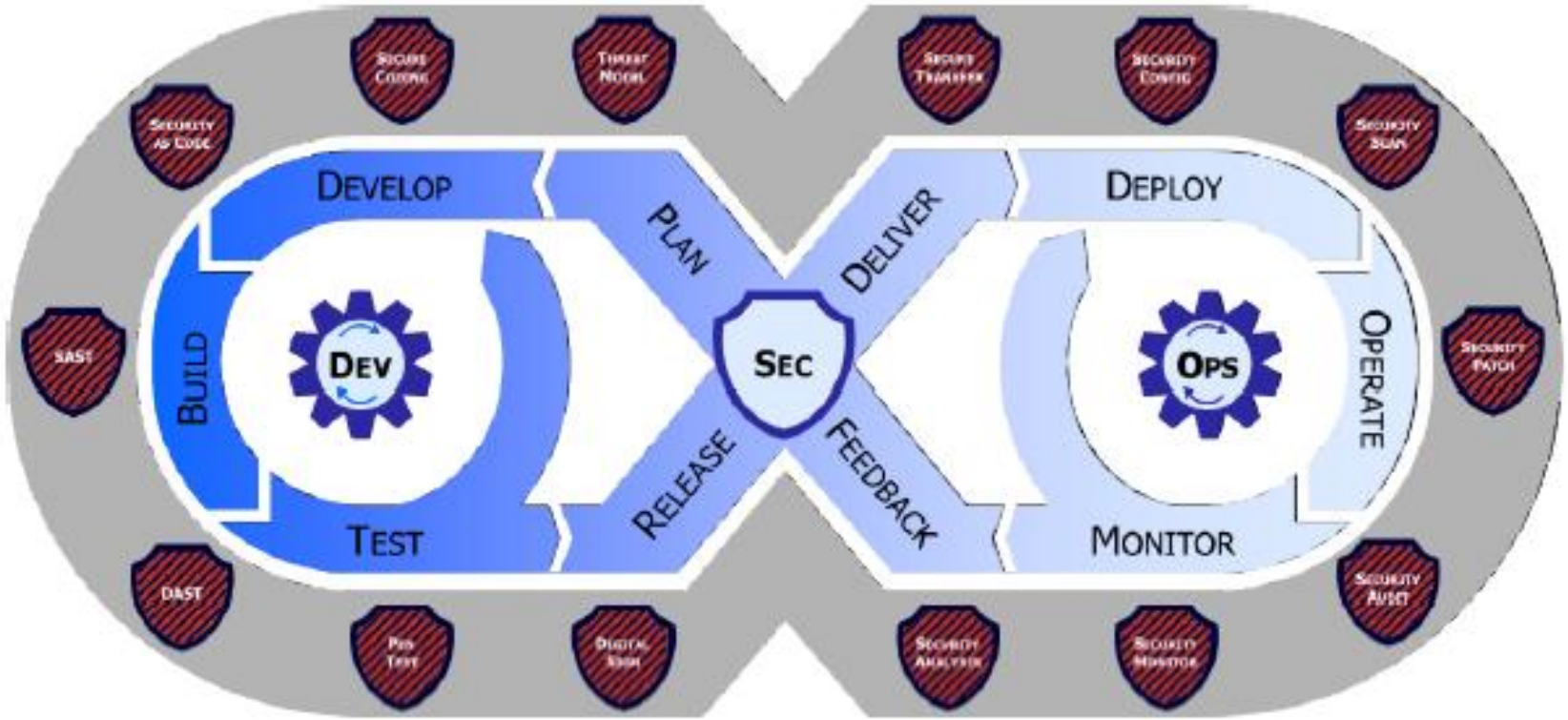


Figure 3 DevSecOps Distinct Lifecycle Phases and Philosophies

Department of Defence, DoD Enterprise DevSecOps Strategy Guide 2.0, March 2021

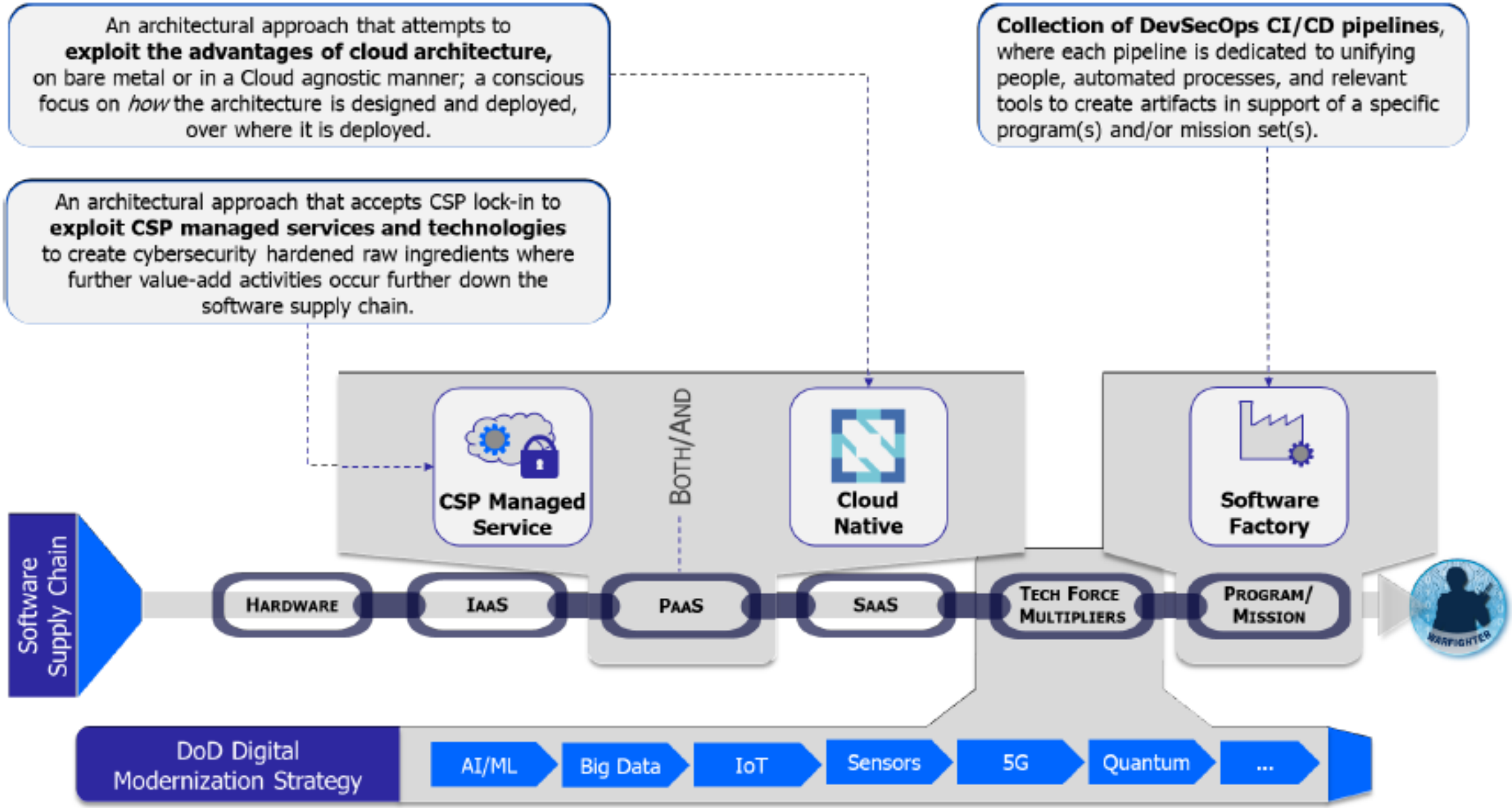


Figure 4 Notional Software Supply Chain

* CSP: cloud Service Provider

Department of Defence, DoD Enterprise DevSecOps Strategy Guide 2.0, March 2021

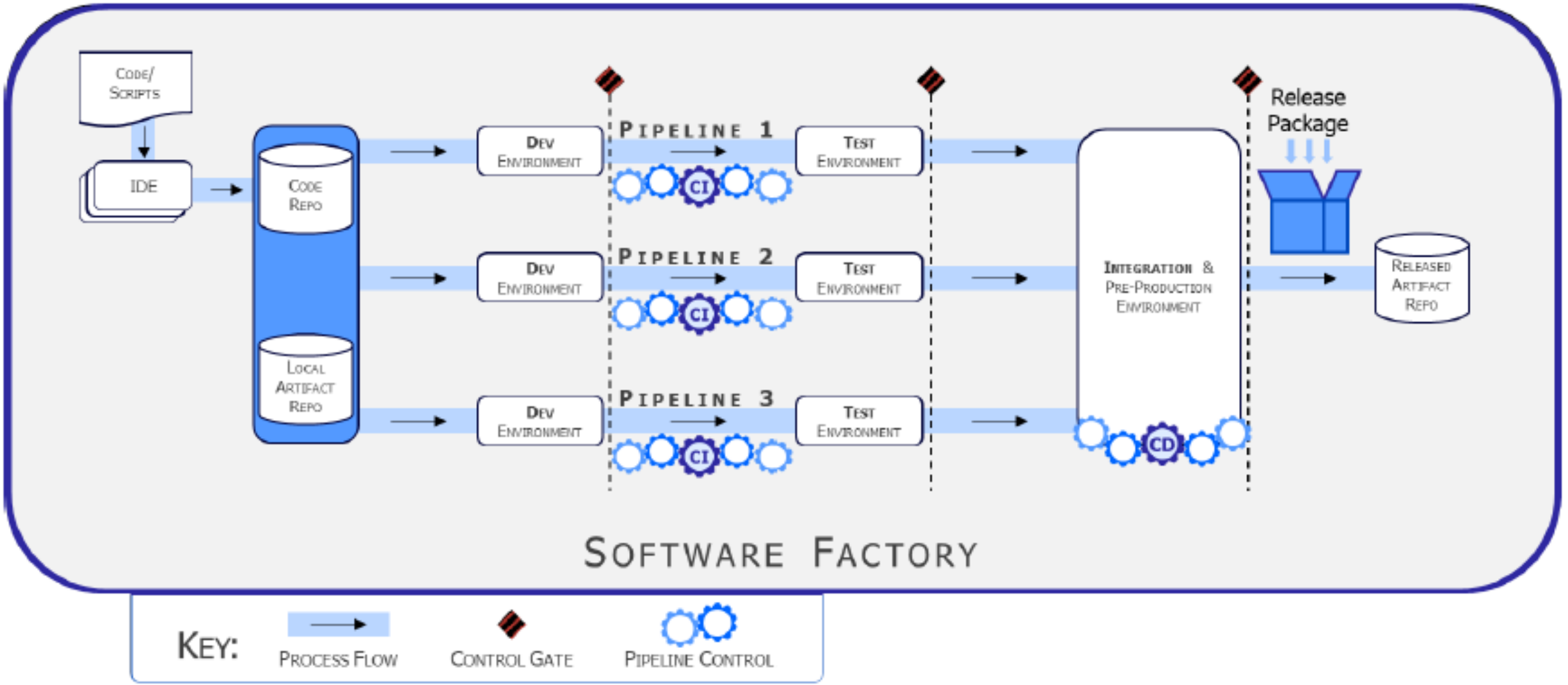


Figure 5 Normative Software Factory Construct

Department of Defence, DoD Enterprise DevSecOps Strategy Guide 2.0, March 2021

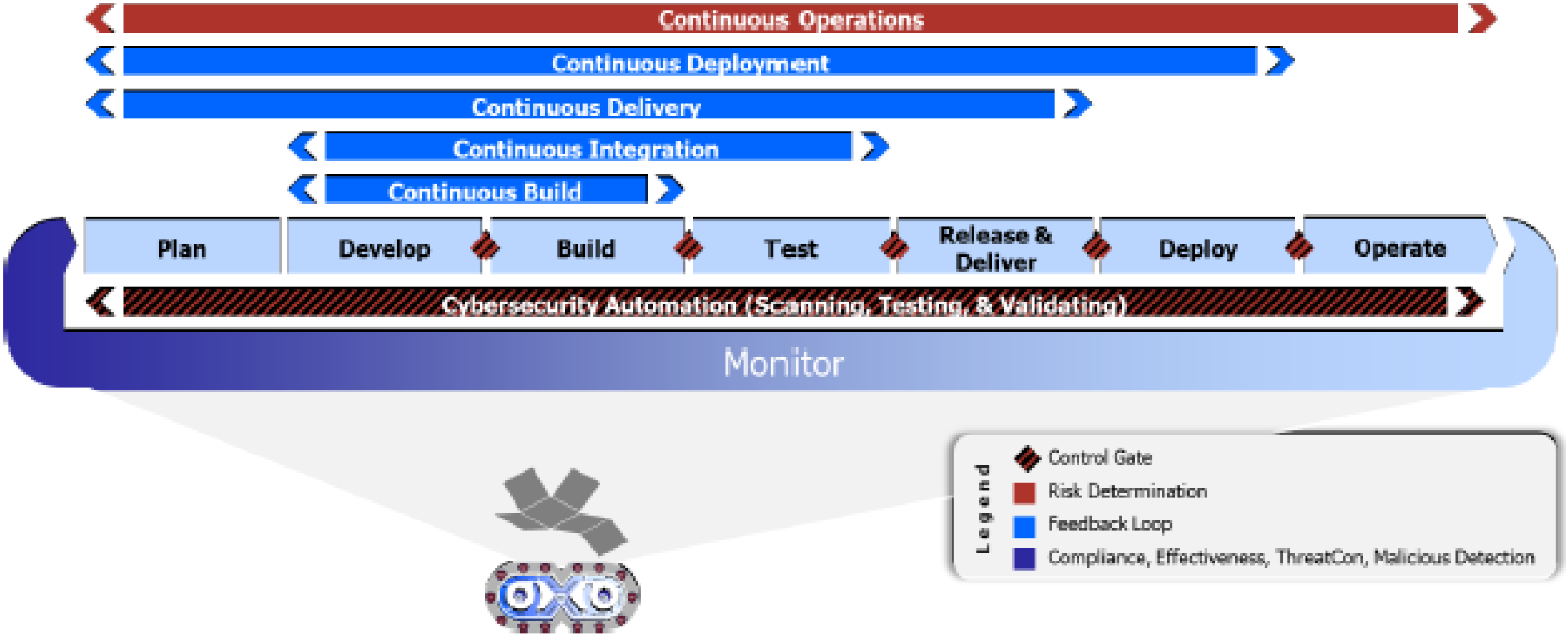
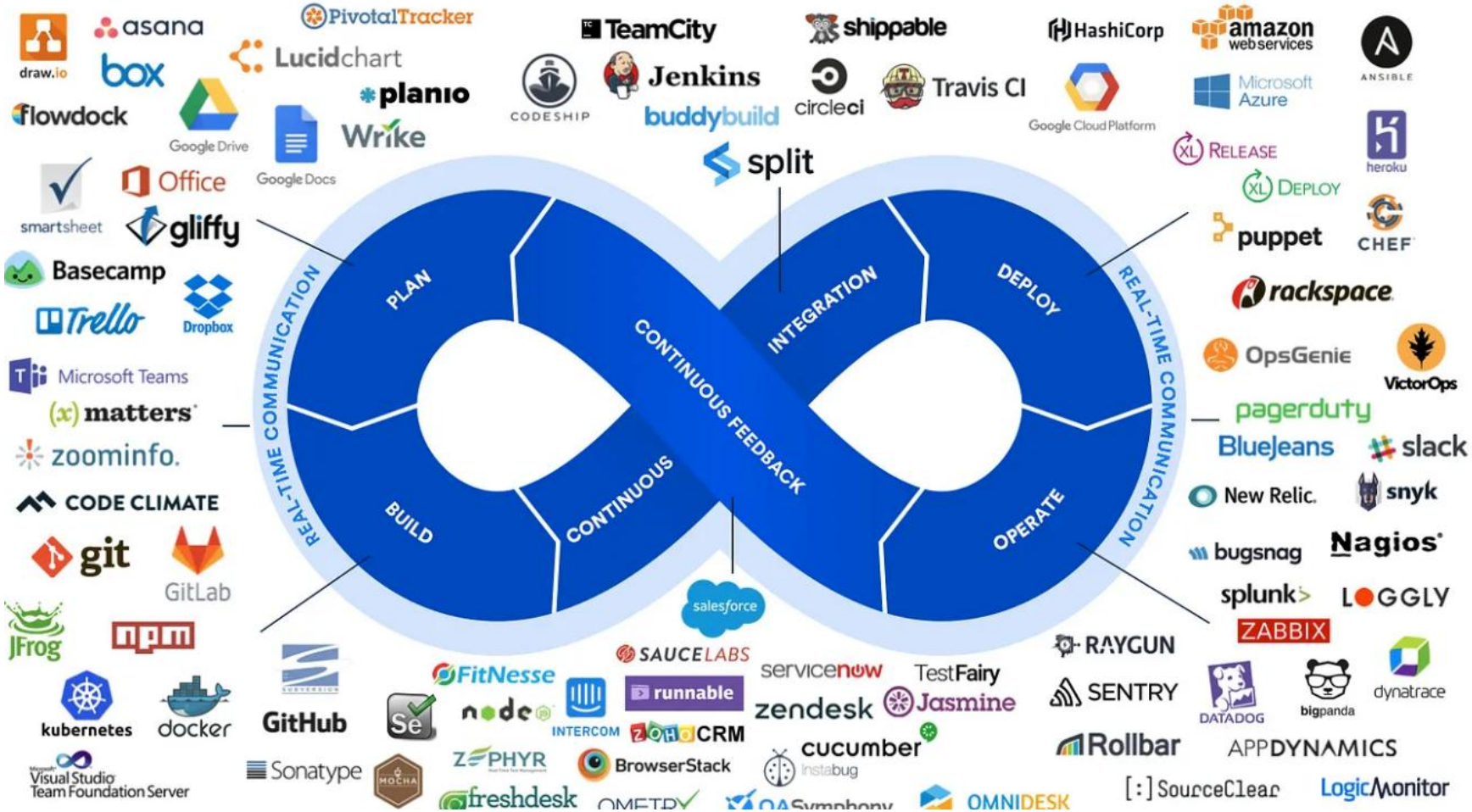


Figure 6 DevSecOps Lifecycle Phases, Continuous Feedback Loops, & Control Gates

Department of Defence, DoD Enterprise DevSecOps Strategy Guide 2.0, March 2021

DevOps Tool chain

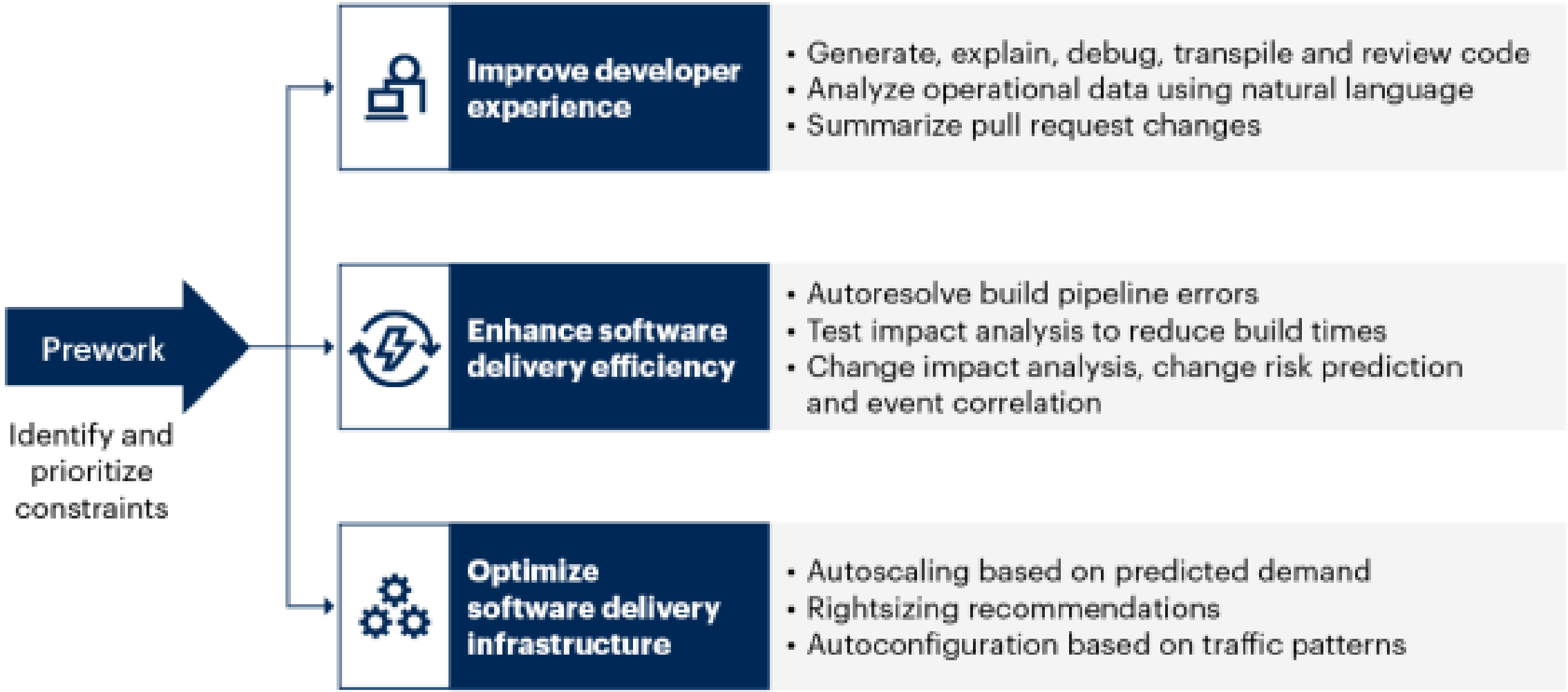
Platform Engineering



https://medium.com/cloud-native-daily/the-ultimate-guide-to-the-top-devops-tools-you-need-to-know-exploring-the-top-devops-tools-11e5f8aa9686

And…

Three Approaches to Augment DevOps With AI



Source: Gartner
801906_C

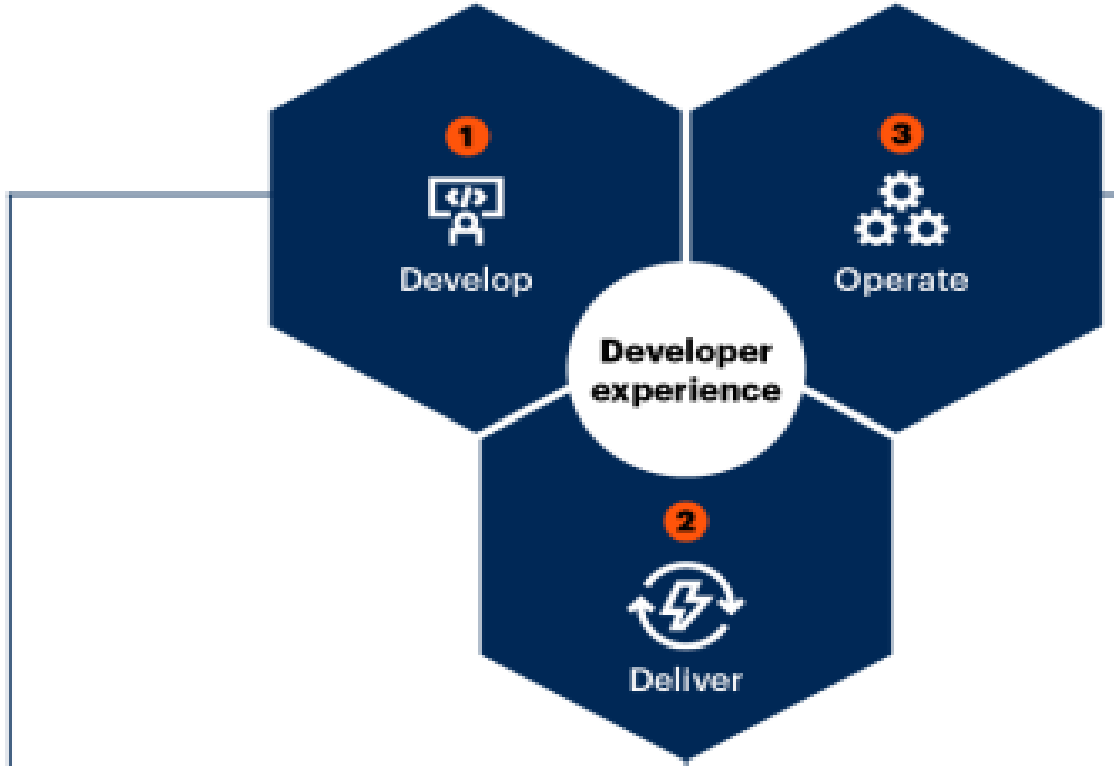
Iteratively Resolve The Greatest Constraint Within Each Phase of the SDLC

Plan	Create	Verify	Release	Configure	Operate
• Prioritize requirements • Ideate ⊙ Triage backlog • Refine requirement • Create user stories ⊙ Acceptance test creation • Create release plan ⊙ Threat modeling	• Architect • Design ⊙ Code ⊙ Search/ discover • Test ⊙ Debug • Refactor • Run static code analysis ⊙ Code review and merge changes	• Write build pipeline code ⊙ Run unit tests • Run security scans • Run "fitness functions" ⊙ Debug and fix build errors • Build artifacts • Provision test environments • Deploy artifacts • Runtime checks	• Feature management ⊙ Change impact analysis • Acceptance tests ⊙ Production readiness tests • Performance and chaos tests • Test backup and recovery • Deploy/verify changes ⊙ Approvals	⊙ Create config playbooks • Config policy automation • Configuration drift detection • Configuration drift correction ⊙ Implementing config controls	• Incident response • Monitoring service levels ⊙ Analyzing logs, metrics, traces ⊙ Root cause analysis • Triage alerts • Recover from failures ⊙ Cloud operations

Source: Gartner
801906_C

Gartner

Improve Developer Experience in DevOps With AI-Enabled Use Cases



- Write, explain, debug, refactor and review code
- Summarize pull request changes, review comments and discussion threads
- Perform semantic search and "chat" with the codebase and documentation

- Selective test runs to shorten build times (fast feedback loops)
- Aggregate and summarize build and deployment pipeline errors
- Early detection of change failures with anomaly detection
- Make it easier to analyze impact of a new change being deployed

- Analyze operational data using natural language ("Why did my service break?")
- Aid root-cause analysis by correlating alerts and change-related data
- Reduce alert noise by deduplicating and normalizing event data

Source: Gartner

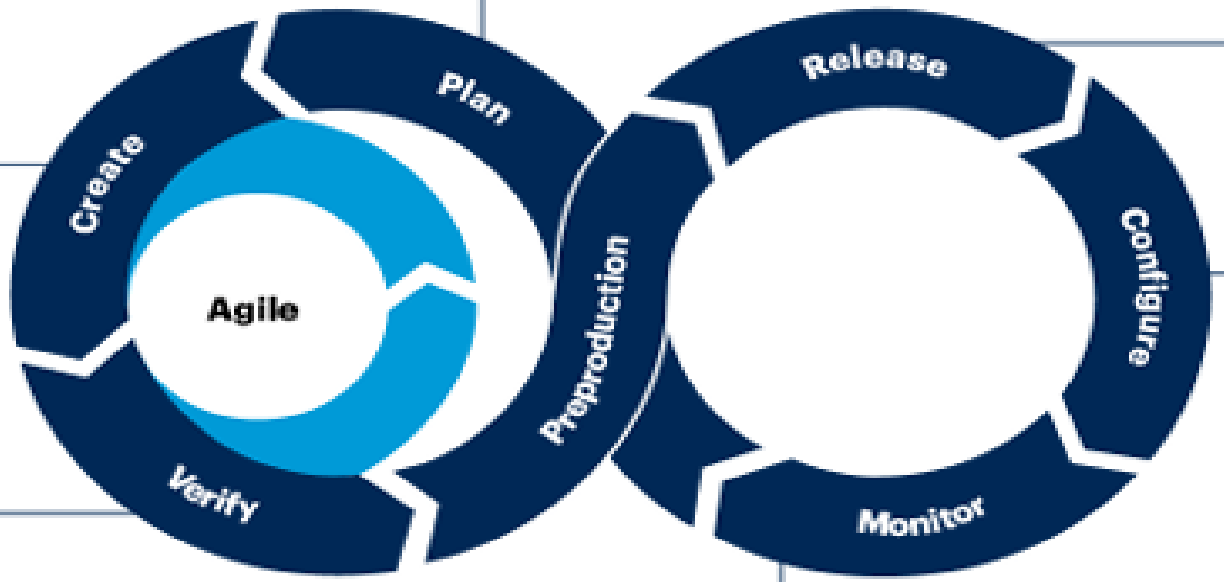
Gartner

AI-Augmented DevOps Use Cases to Optimize the SDLC

- Generate, refactor, debug, transpile, review and explain code
- Generate unit tests from code and natural language
- Convert UX designs to code
- Generate comments and documentation from code
- Explain the security vulnerability

- Summarize minutes of meeting
- Summarize issue comments
- Generate acceptance tests from user stories
- Automated generation of threat models
- Explain enterprise-specific terms
- Prioritize and label issues based on descriptions

- Continuous release verification
- Evaluate multiple hypotheses within a multivariate experiment
- Change impact analysis
- Predict change failures by assessing release readiness
- Autoenable/disable feature flags based on health checks



- Summarize PR changes and review comments
- Autoremediation of security vulnerabilities
- Intelligent test selection to reduce CI build times
- Test data generation (synthetic data)

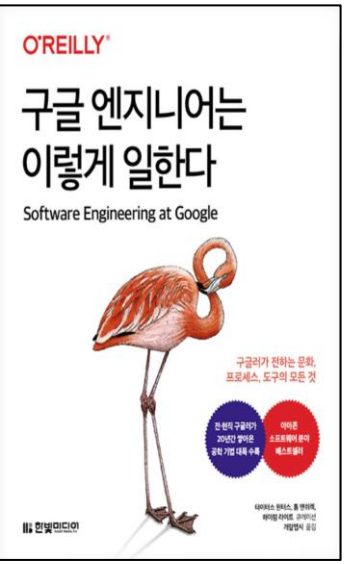
- Pattern recognition, anomaly detection, event correlation, root-cause analysis, self-healing systems
- Intelligent workload optimization (optimize cost, reliability and
- sustainability, and achieve trade-offs)
- Predictive and prescriptive analytics

- Generate configuration automation runbooks
- Continuously detect and fix configuration drift
- Recommend cloud configurations across application, infrastructure, security and data services

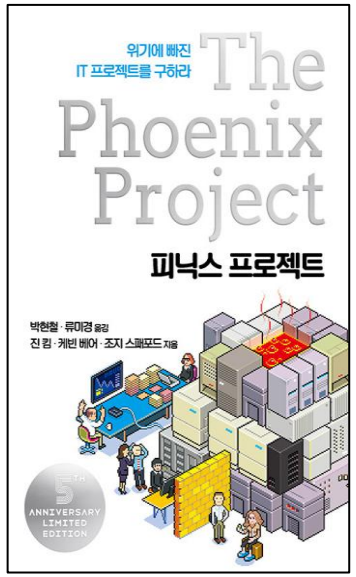
Source: Gartner

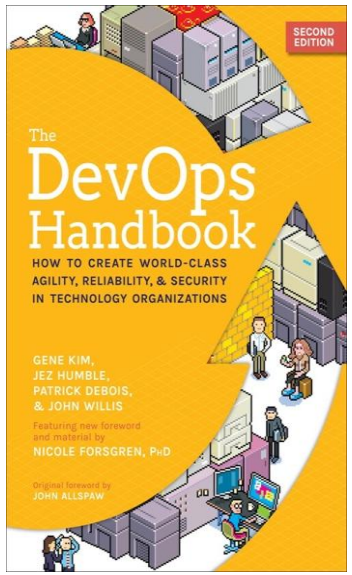
801906_C

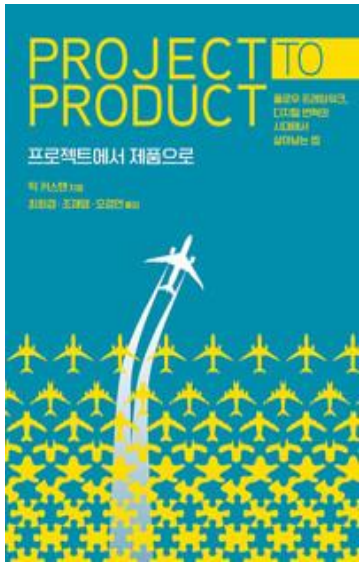
추천도서


O'REILLY
구글 엔지니어는 이렇게 일한다
Software Engineering at Google
한빛미디어


The DevOps Adoption Playbook
데브옵스 도입 전략

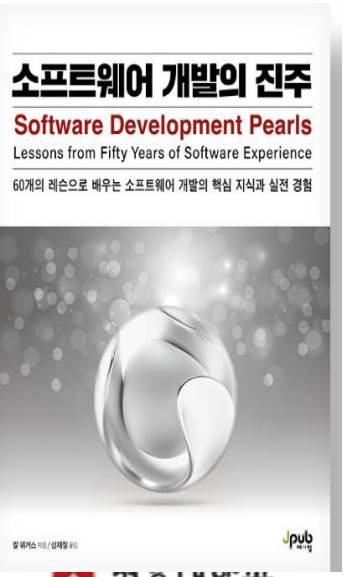

위기에 빠진 IT 프로젝트를 구하라
The Phoenix Project
피닉스 프로젝트
박현철 · 류미경 옮김
진 킴 · 케빈 베어 · 조지 스패포드 지음
5TH ANNIVERSARY LIMITED EDITION

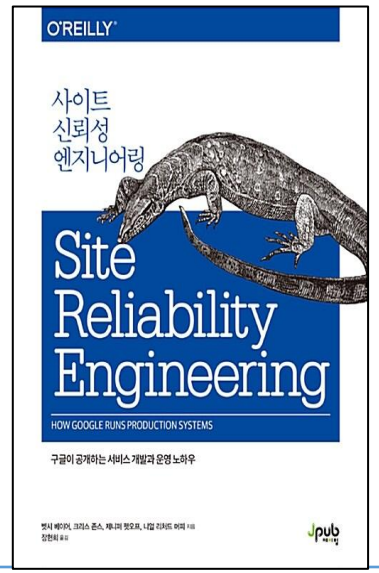

SECOND EDITION
The DevOps Handbook
HOW TO CREATE WORLD-CLASS AGILITY, RELIABILITY, & SECURITY IN TECHNOLOGY ORGANIZATIONS
GENE KIM, JEZ HUMBLE, PATRICK DEBOIS, & JOHN WILLIS
Featuring new foreword and material by NICOLE FORSGREN, PhD
Original foreword by JOHN ALLSPAW

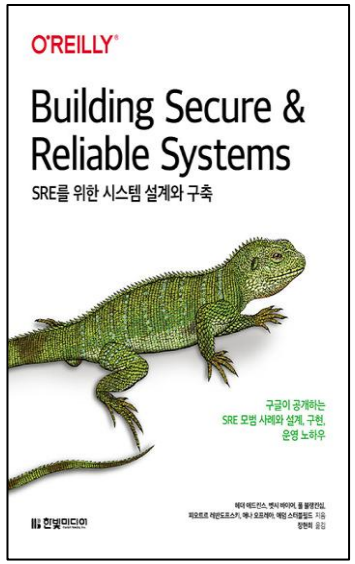

PROJECT TO PRODUCT
프로젝트에서 제품으로

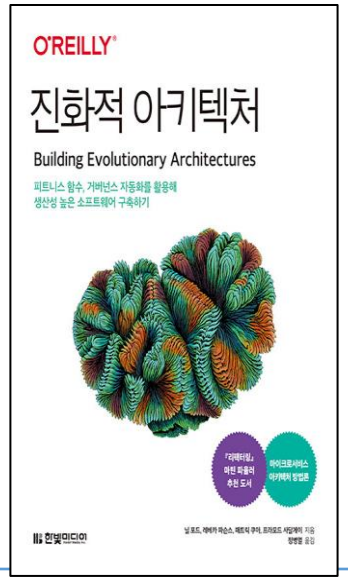

O'REILLY
데브옵스 엔지니어를 위한 실전 관찰 가능성 엔지니어링
Observability Engineering
한빛미디어

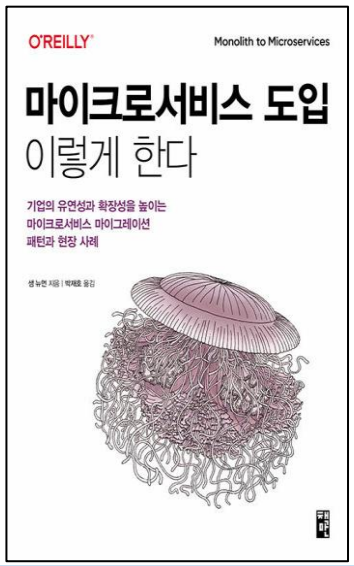

O'REILLY
Learning Chaos Engineering
카오스 공학
견고한 시스템을 만드는 카오스 공학의 이론과 실제


소프트웨어 개발의 진주
Software Development Pearls
Lessons from Fifty Years of Software Experience
60개의 레슨으로 배우는 소프트웨어 개발의 핵심 지식과 실전 경험
Jpub

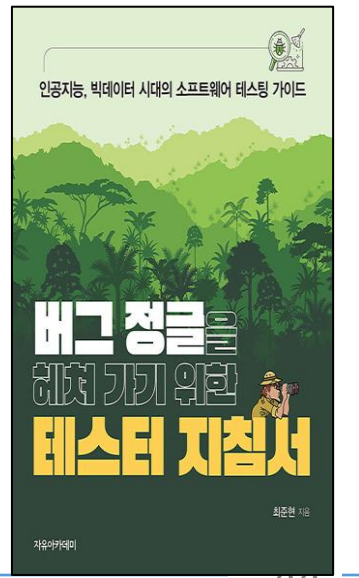

O'REILLY
사이트 신뢰성 엔지니어링
Site Reliability Engineering
HOW GOOGLE RUNS PRODUCTION SYSTEMS
구글이 공개하는 서비스 개발과 운영 노하우
Jpub


O'REILLY
Building Secure & Reliable Systems
SRE를 위한 시스템 설계와 구축
구글이 공개하는 SRE 모범 사례와 설계, 구현, 운영 노하우
한빛미디어


O'REILLY
진화적 아키텍처
Building Evolutionary Architectures
피트니스 함수, 거버넌스 자동화를 활용해 생산성 높은 소프트웨어 구축하기
한빛미디어


O'REILLY
Monolith to Microservices
마이크로서비스 도입 이렇게 한다
기업의 유연성과 확장성을 높이는 마이크로서비스 마이그레이션 패턴과 현장 사례


요즘 우아한 개발
기획하고 개발하고 회고하라
배달의민족을 만든 우아한형제들의 조직문화, 온보딩, 기획, 개발, 인프라 구축 이야기
우아한테크


인공지능, 빅데이터 시대의 소프트웨어 테스팅 가이드
버그 정글을 헤쳐 가기 위한
테스터 지침서
최준현 지음
자유아카데미

Learn from Failure

"원하는 결과가 나오지 않은 실험은

ㅇㅇㅇ실험이다!!!"

과정 안내

- 본 과정은 소프트웨어 개발 현장에서 필수적으로 자리 잡고 있는 DevOps의 기본 개념과 문화를 이해하고, 관련된 방법론과 기술을 체계적으로 학습하는 것을 목표로 합니다. 특히 클라우드 환경과 오픈소스 도구를 활용한 실습 및 팀 프로젝트를 통해 DevOps 프로세스를 직접 설계하고 구현하며, 실무 중심의 역량과 오픈소스 활용 능력을 강화합니다.

- 중간고사 25% 기말리포트 5% 개인과제 20% 팀과제 40% 참여도 10%
  * 시험은 플랫폼을 이용한 실기 시험 포함

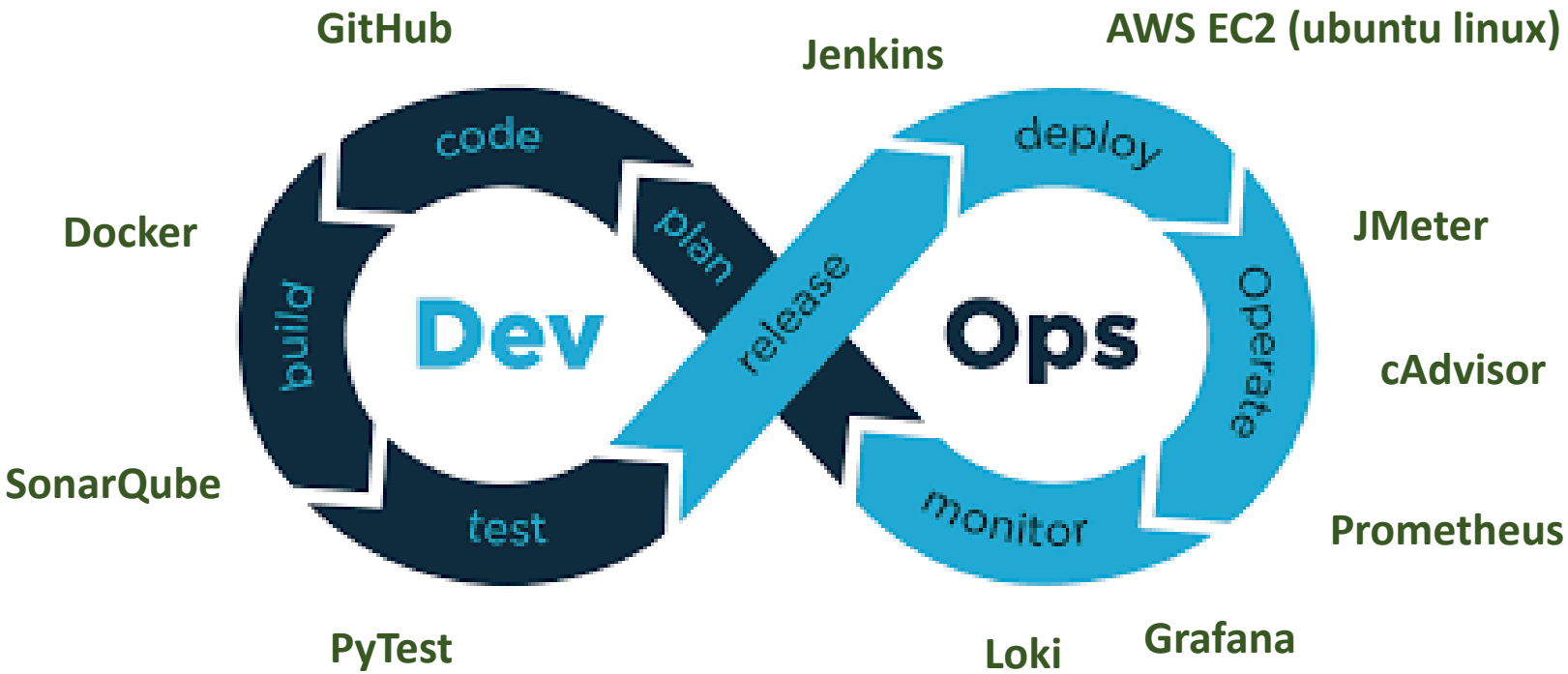
- Values
  - Team Effort
  - **협업과 공유**
  - **신뢰와 존중**
  - **다른 팀원, 다른 팀의 성공 돕기**
  - **실험과 실패로부터 배우기**
  - **Continuous Learning**

"Class Wiki 운영 – 활발한 정보 공유자에 대한 가점 부여"
- 오류 해결, 도구사용 팁 등 tech blog, 토론, Q&A 등
(링크 별도 공지)

"원하는 결과가 나오지 않은 실험은 성공한 실험이다!!!"
(삽질경험, Class에 꼭 공유하기)

"팀원간 상호평가 반영 "

DevOps Tools in class



과제물

- 개인과제: 단계별 과제물을 통해 기본적인 도구 활용 방법 습득
  - Jenkins, Docker, PyTest 등등

- 팀과제: DevOps를 지원하는 도구를 실제 팀별 서비스 구현 프로젝트에서 활용

  - 팀별 구현 서비스 정의 및 DevOps 파이프라인 적용하여 서비스 구현/배포
  - 개발방법 : LegacySW (오픈소스 포함) 개선, Develop from scratch
  - 개발언어/기술: 제한 없음 (단, UI + OpenAPI기반의 Backend)
  - **본 과목의 목적은 서비스 기획보다는 팀 협업 개발을 통한 DevOps 배포 실습임을 고려함**
  - 오픈소스 도구 조사 (별도 안내)

  - **다양한 DevOps Pipeline 구성 및 경험 공유 : 수업에서 다룬 tool에 대한 in-depth study, 다양한 형태의 응용과 활용, 유사목적 다른 도구의 검토와 적용**

주차별 수업계획

주차	일자	과제
1	09/03	#1_샘플 프로그램 작성
2	09/10	#2_서버_Git_IDE_연동
3	09/17	#3_젠킨스_빌드_배포
4	09/24	#4_도커빌드_배포
5	10/01	**팀과제 중간발표**
6	10/08 (추석)	휴강 (보강은 미정)
7	10/15	#5_자동화테스트(Pytest 등)
8	10/22	중간시험 (이론및실기)

주차	일자	과제
9	10/29	#6_정적분석 (SonarQube)
10	11/05	#7_모니터링(Prometheus, Grafana등)
11	11/12	**팀별 자료조사(튜토리얼) 발표**
12	11/19	#8_성능테스트(Jmeter)
13	11/26	#9_로그분석(Loki, Promtail, Grafana)
14	12/03	DevOps 실천방안
15	12/19	**팀과제 최종 발표**
16	12/17	기말 리포트 (추후공지)

* 진행내용 및 순서는 일부 조정될 수 있음.

주요 기술에 대한 동영상 교육 컨텐츠

seq.	과목명	소요시간	권장정도	비고
1	# 클라우드 컴퓨팅 첫 걸음 시작하기!	2h30m	*****	6개 실습
2	# 실무자가 알려주는 Git 입문	2h4m	*****	14개 실습
3	# 실무자가 알려주는 Git 활용한 프로젝트 관리	2h5m	*****	14개실습
4	# 생활코딩! 처음 배우는 리눅스	9h40m	***	
5	# Jenkins를 활용한 SW 통합 및 배포 관리	6h10m	*****	18개 실습
6	# Docker 를 이용한 Container 기술에 대한 이해	4h13m	*****	
7	# SW 품질 향상을 위한 코드 정적분석	2h53m	***	
8	# SW 유지보수성 향상을 위한 Clean Code	4h15m	***	
9	# 처음 시작하는 파이썬 프로그래밍	8h36m	**	42개 실습
10	# 마이크로서비스 아키텍처 : 패턴과 핵심 기술	8h7m	**	

http://codepresso.kr 에 회원가입 필요

개인과제 #1. 샘플앱 만들기

- 개인별 Toy Example 만들기
  - Python, FastAPI 이용한 간단한 나만의 'TodoList' 웹프로그램 만들기
  - 일단 DB 사용 없이 Jason 파일에 CRUD (Create, Read, Update, Delete) 하면 됨
  - 제출내역: 화면 캡처하여 보고서작성(pdf) + 프로그램 압축파일(zip)
  - 제공하는 가이드 참조 (링크 별도 공지)

- 제출(사캠): 9/9(화) 까지

차주 수업전까지

- 개인별 설문조사 참여 (9/9 까지)
  - 본인의 SW 도구 관련 경험
  - 수강신청 사유
  - 수업시간에 알고 싶은 내용 및 수업에 바라는 사항

- 팀 구성
  - 6개팀 구성 (팀당 5인 정도)
  - 모집 공고 등을 통해 자체구성 할 경우 차주 화요일(9/9)까지 조교에게 제출
  - 수업전까지 확정 예정

- 교육동영상 수강을 위한 회원가입 http://codepresso.kr
  - 차주 월요일까지 **가입완료 후 완료여부 Survey참여**

- 교육 동영상(codepresso) 수강
  - Linux, 클라우드, Git관련

Questions & Answers