



생성AI가 촉발하는 SW생산방식의 변화

소프트웨어정책연구소 **유호석** 산업정책연구실장/공학박사

< SPRI 포럼 >

2023. 6. 27.

판교 코사이어티 타운홀

생성AI가 촉발하는 SW생산방식의 변화

Contents

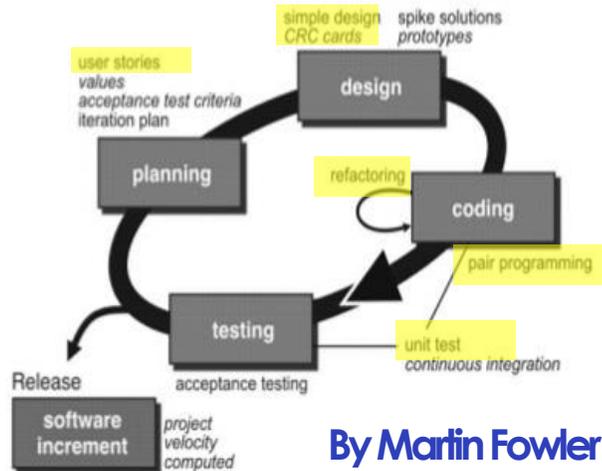
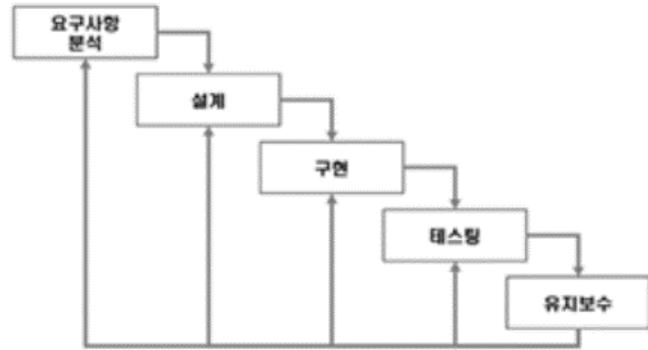


- 1 SDLC (SW Development Lifecycle) 의 과거와 현재
- 2 SW Delivery 가속 도구, 생성형AI
- 3 AI가 만드는 Software 2.0
- 4 AI 활용의 고려사항

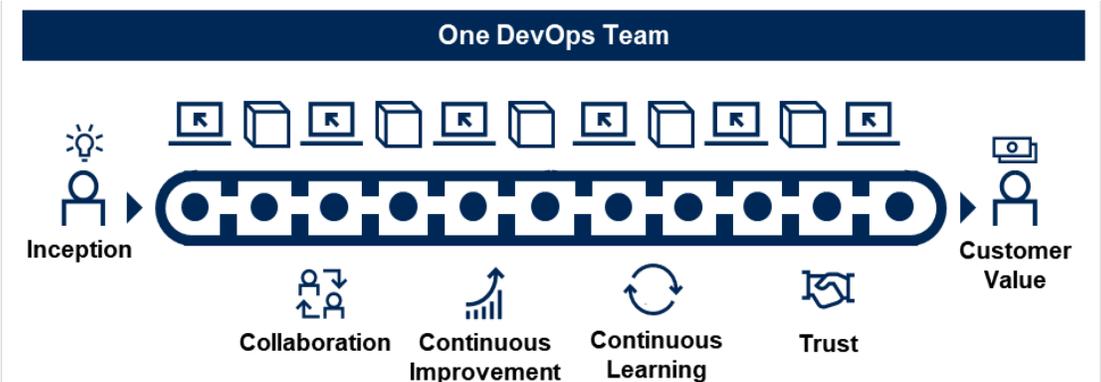
1. SDLC (SW Development Lifecycle) 의 과거와 현재

◆ Waterfall 에서 Agile 을 거쳐 DevOps 까지

< 과거 : Waterfall 에서 Agile 로 >

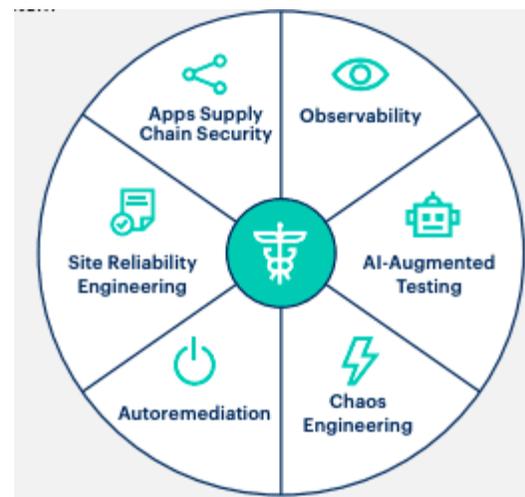
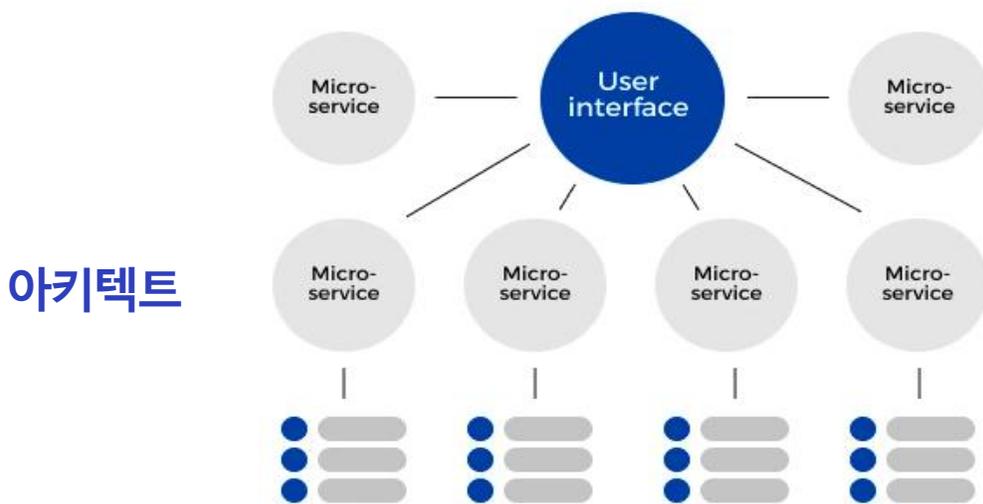


< 현재 : DevOps >



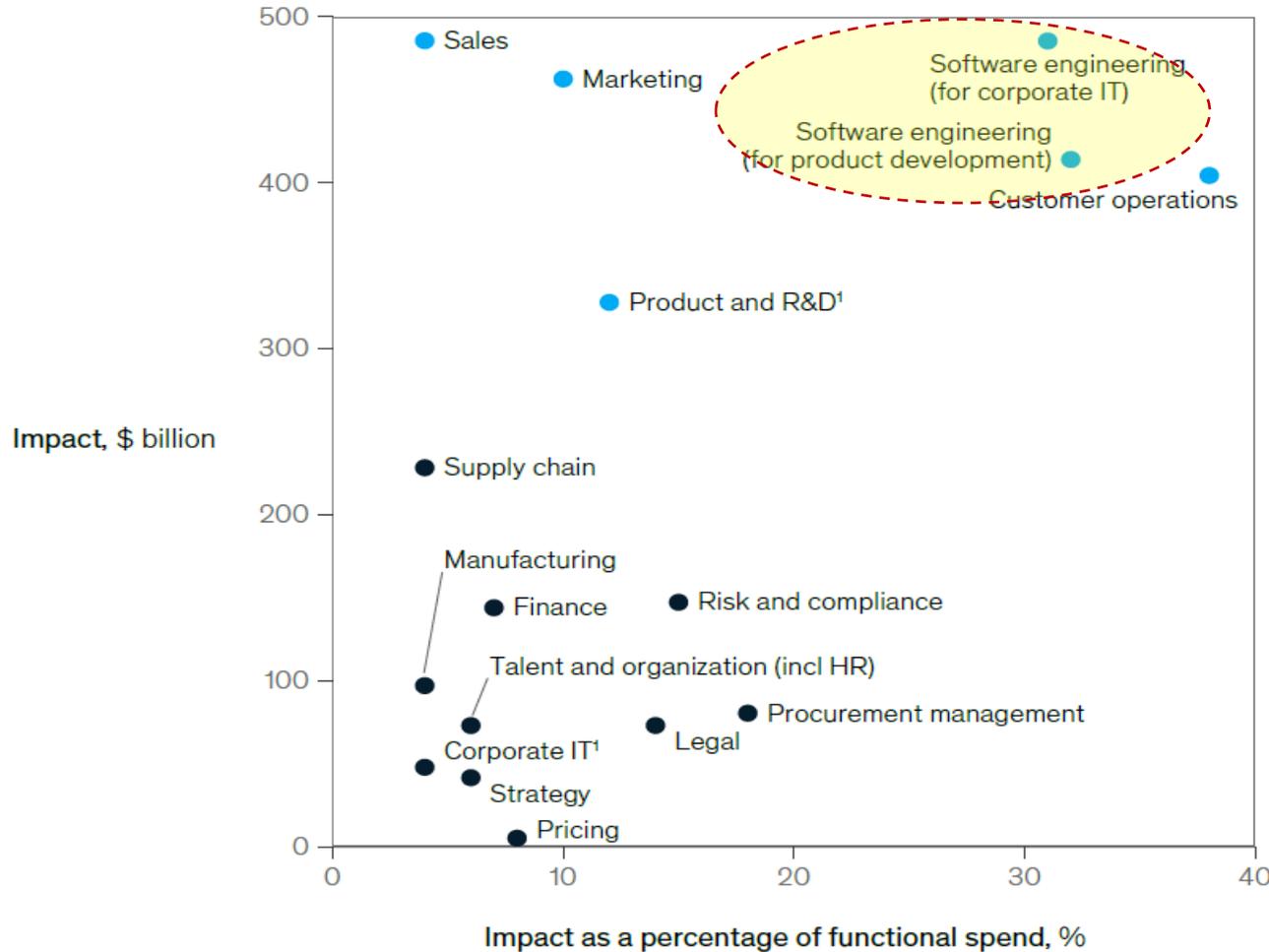
1.SDLC의 과거와 현재

◆ DevOps 개발자는 SDLC 상에서 아키텍트, 운영자, 품질 등 다양한 역할을 수행해야 함



2. SW Delivery 의 가속 도구, 생성형AI

◆ SW엔지니어링 부문이 생성형AI의 경제적 영향을 가장 많이 받을 전망 (McKinsey '23.6월)



Inception and planning

Software engineers and product managers use generative AI to assist in analyzing, cleaning, and labeling large volumes of data, such as user feedback, market, trends, and existing system logs.

System design

Engineers use generative AI to create multiple IT architecture designs and iterate on the potential configurations, accelerating system design, and allowing faster time to market.

Coding

Engineers are assisted by AI tools that can code, reducing development time by assisting with drafts, rapidly finding prompts, and serving as an easily navigable knowledge base.

Testing

Engineers employ algorithms that can enhance functional and performance testing to ensure quality and can generate test cases and test data automatically.

Maintenance

Engineers use AI insights on system logs, user feedback, and performance data to help diagnose issues, suggest fixes, and predict other high-priority areas of improvement.

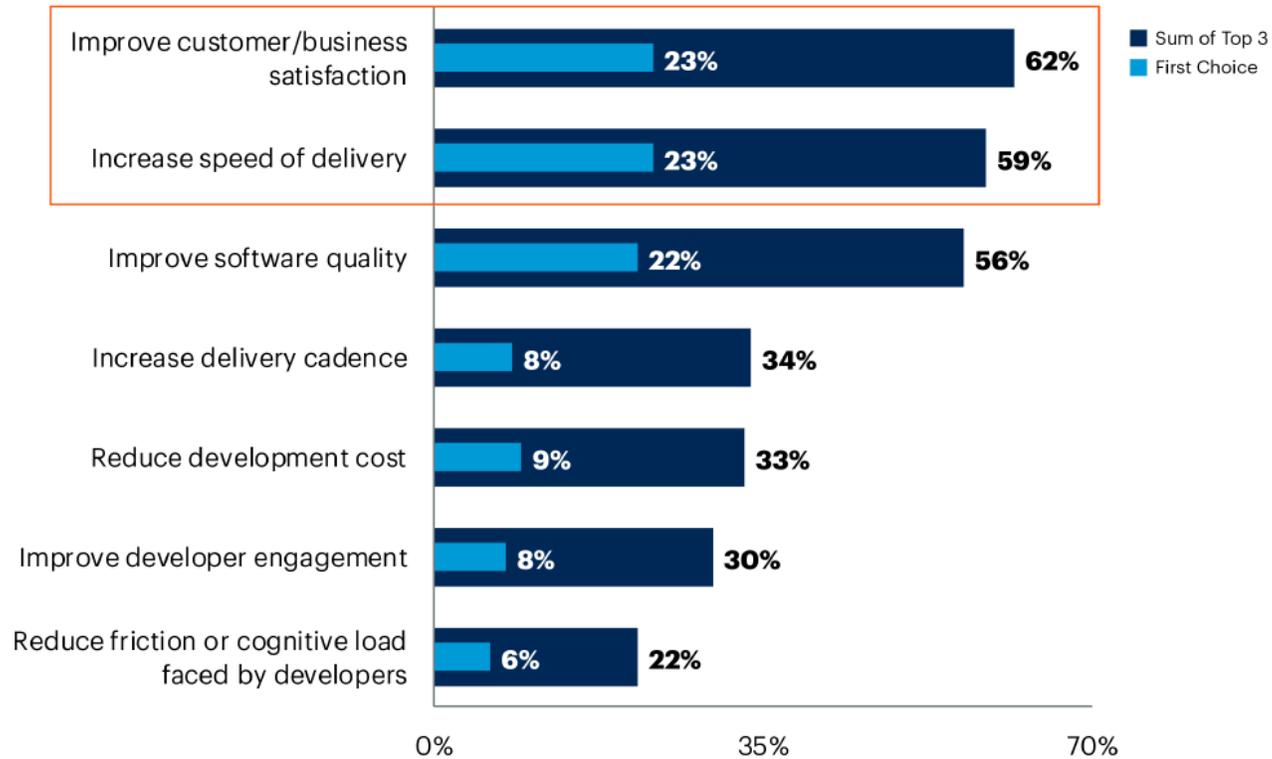
* McKinsey(June 23), <The economic potential of generative AI>

2. SW Delivery 의 가속 도구, 생성형AI

◆ 북미 개발리더 절반 이상이 생성형AI를 활용할 것으로 응답*

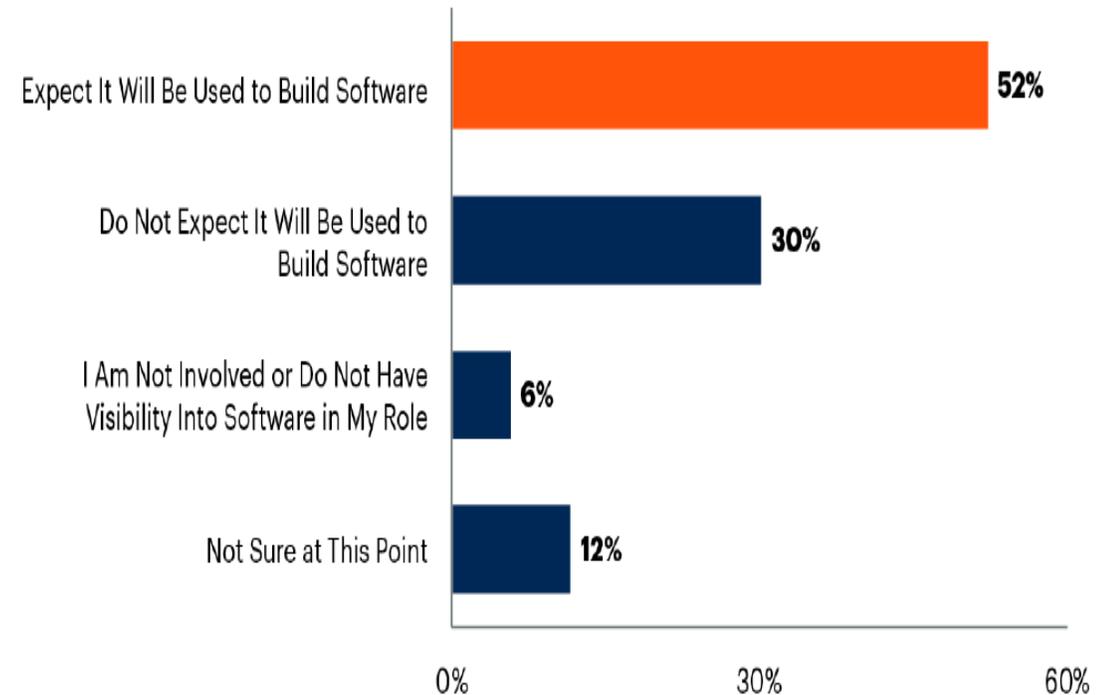
Top Goals for Improving Developer Productivity

Top 3 Rank



n = 300, engineering leaders with teams focused on application development/software engineering,

More Than Half of IT Leaders Expect Their Organization Will Use Generative AI to Build Software



n = 86; IT Leaders with knowledge of Generative AI at their organizations

* 가트너의 2022년 12월 북미지역 패널조사 결과 < Quick Answer: How Can Generative AI Tools Speed Up Software Delivery? >

2. SW Delivery 의 가속 도구, 생성형AI

◇ (예시 시나리오) SPRI 주차관리 시스템

< SPRI 주차관리 시스템 >

아이디	<input type="text"/>	<input checked="" type="checkbox"/> 아이디 저장	로그인
비밀번호	<input type="password"/>		



할인등록 | 할인내역 |

차량 번호로 검색해주세요.

차량 번호

검색

적용년월	기본(무료)한도		
	총한도금액(원)	사용금액(원)	잔여금액(원)
2023-06	무제한	344,900	-

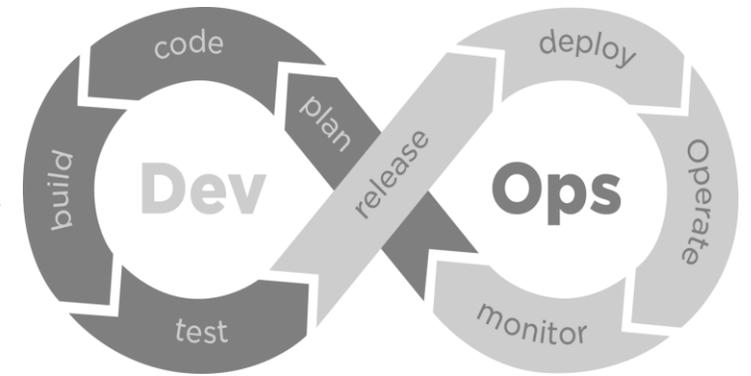
차량번호	입차시각	주차시간(분)	할인키	등록자
검색된 차량 정보가 없습니다.				

월할인 차량 방문객 2시간 입주사 확인무료

< 목표 >

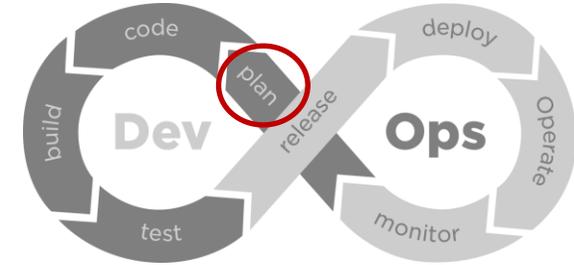
총무 직원이 매일 약 50명의 직원들의 차량번호를 주차관리 시스템에 수동으로 등록하고 있어서 이 업무를 자동화하려고 함

< DevOps + 생성AI 적용 >



2. SW Delivery 의 가속 도구, 생성형AI

◆ (Plan) 요구사항 작성 및 분석, 사용자 시나리오 작성



프롬프트

나의 회사에는 주차장이 있는데 Employees 가 해당 주차장을 무료로 사용하려면 특정 웹주소 게시된 웹할인 시스템에 로그인하여 차량번호를 등록해야 해.

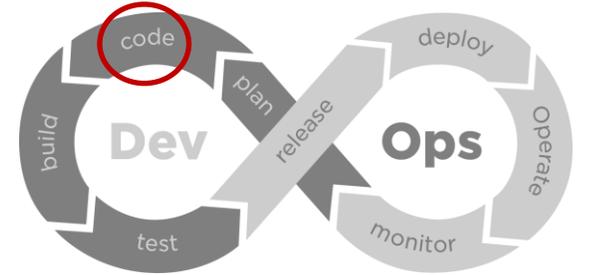
웹할인 시스템에 로그인하여 다수의 차량번호를 자동으로 등록하는 프로그램을 만들기 위한 **할 일 목록을 정의해 줘. 아직 코드를 만들지는 말고**

응답

1. 웹할인 시스템의 구조와 행태를 분석합니다.
2. 필요한 데이터를 식별합니다: 시스템 URL, 로그인 자격 증명, 번호판 입력 필드
3. 웹 자동화 라이브러리를 선택합니다.
4. 필요한 라이브러리를 설치합니다.
5. 자동화를 위해 웹 드라이버를 설정합니다.
6. 로그인 프로세스를 구현합니다.
7. 번호판 데이터를 준비합니다.
8. 번호판 등록 프로세스를 자동화합니다.
9. 오류 조건을 처리합니다.
10. 프로그램의 기능을 테스트하고 확인합니다.

2. SW Delivery 의 가속 도구, 생성형AI

◆ (Code) Python 라이브러리를 식별하고, 예시코드를 작성



프롬프트

웹할인 시스템에 로그인하여 다수의 차량번호를 자동으로 등록하는 python 프로그램을 만들기 위한 할일 목록은 다음과 같아. 단계별로 사용하는 **Python 코드의 예시를 보여줘**

1. 모든 vehicle number 가 저장된 파일을 읽기
2. 시스템에 로그인 : 접속주소는 "http://<myparking>", id를 입력하는 필드명은 "loginForm.user_id", password를 입력하는 필드명은 "loginForm.password", id값은 "<user_id>"이고 password 값은 "<user_pwd>"
3. vehicle number 조회 페이지로 이동
4. 1번에서 읽은 파일에서 vehicle number 필드를 하나씩 조회 : 파일에서의 필드명은 "vehicle_number", 웹페이지에서의 차량번호 필드명은 모두 "search_form.license_plate_number"
5. 차량번호 조회 페이지에서 한국어로 "검색된 차량 정보가 없습니다." 라는 메시지가 있는 경우 다음 차량번호로 4번 단계를 실행
6. 차량번호 조회 페이지에서 한국어로 "검색된 차량 정보가 없습니다." 라는 메시지가 없는 경우 해당 차량번호로 양식 제출

모든 차량번호에 대해 상기 4번과 5번을 반복하고, 반복이 끝나면 "모든 차량번호가 등록되었습니다." 라는 메시지를 출력 해.

응답

```
bash
pip install selenium beautifulsoup4

Here is a simplified script to demonstrate the general process:

python
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from getpass import getpass
from bs4 import BeautifulSoup

# You'll need to download a driver for selenium.
# This example uses the Firefox driver (geckodriver),
# but other browsers have their own respective drivers.
driver_path = "/path/to/your/geckodriver"
driver = webdriver.Firefox(executable_path=driver_path)

url = 'http://...'
driver.get(url)

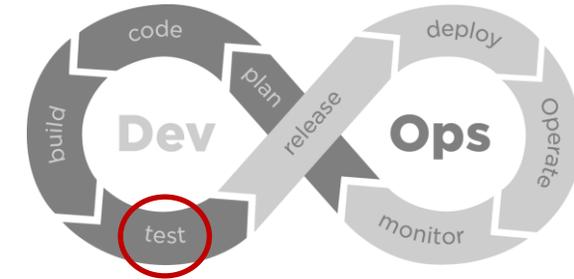
# This is a simplified example. In the real situation,
# you may need to interact with more elements or even deal with CAPTCHA.

# Log in
username = driver.find_element_by_name('username')
password = driver.find_element_by_name('password')

# You should use secure ways to store and retrieve credentials.
# For simplicity, this example asks them from the terminal.
username.send_keys(input("Enter your username: "))
password.send_keys(getpass("Enter your password: "))
password.send_keys(Keys.RETURN)
```

2. SW Delivery 의 가속 도구, 생성형AI

◆ (Test) 테스트케이스와 테스트 코드를 작성



프롬프트

위의 단계를 Test Driven Development 방식으로 코드를 작성하고 싶어. Unit 테스트를 위한 python **테스트 케이스를 작성**하고, 해당 **테스트 케이스를 실행하는 코드**도 함께 만들어줘

응답

```
python
import unittest
from myparking_program_test import MyParkingProgramTestCase

def run_tests():
    test_suite = unittest.TestLoader().loadTestsFromTestCase(MyParkingProgramTestCase)
    unittest.TextTestRunner(verbosity=2).run(test_suite)

if __name__ == '__main__':
    run_tests()
```

```
python
import unittest
from myparking_program import MyParkingProgram

class MyParkingProgramTestCase(unittest.TestCase):

    def test_read_vehicle_numbers(self):
        program = MyParkingProgram()
        vehicle_numbers = program.read_vehicle_numbers('vehicle_numbers.txt')
        self.assertTrue(len(vehicle_numbers) > 0)

    def test_login_to_system(self):
        program = MyParkingProgram()
        result = program.login_to_system('<myparking>', '<user_id>', '<user_pwd>')
        self.assertTrue(result)

    def test_search_and_register_license_plates(self):
        program = MyParkingProgram()
        program.login_to_system('<myparking>', '<user_id>', '<user_pwd>')
        license_plates = ['ABC123', 'DEF456', 'GHI789']

        for license_plate in license_plates:
```

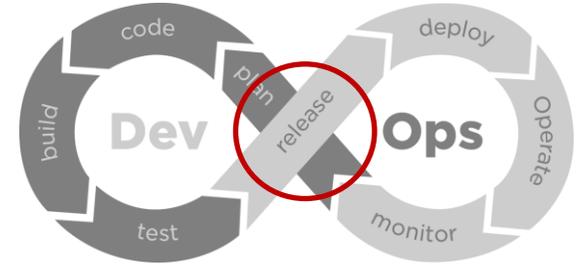
2. SW Delivery 의 가속 도구, 생성형AI

◆ (Release) 예시코드를 Auto Programming 도구에서 완성하여 출시



```
JS demo.js
JS demo.js
1 // create a function that establishes a connection to an
  azure cosmos db database
2
3 function connectToCosmosDB() {
  // create a new instance of the CosmosClient class
  const client = new CosmosClient({
    endpoint: process.env.COSMOS_ENDPOINT,
    key: process.env.COSMOS_KEY,
  });

  // return the client
  return client;
}
```



2. SW Delivery 의 가속 도구, 생성형AI

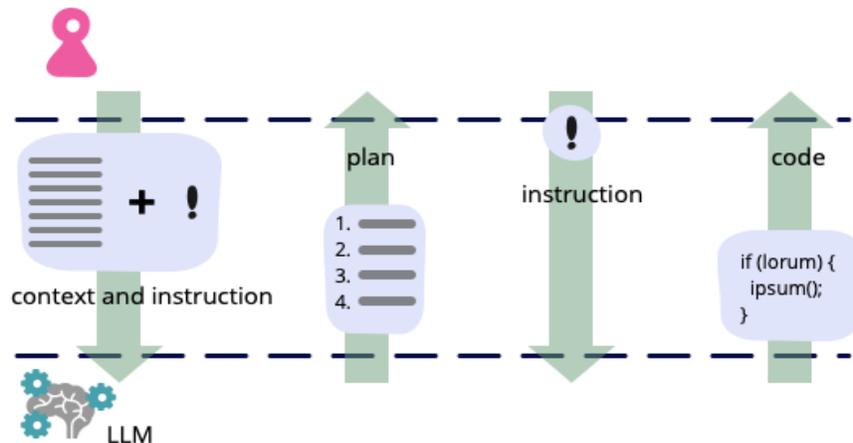
◆ Martin Fowler's Insights from a Programming perspective

1. Generated Knowledge

- 계획을 가지고 단계를 나누어 접근하는 방법
 - ① 문제를 풀기 위한 필수 단계와 정보 요청
 - ② GPT 생성 정보에 대한 피드백
 - ③ GPT의 코드 생성 유도

3. Names for the program elements

프로그램 요소에 대해 이름을 부여한 후 목록 재생성



2. Chain of Thoughts

- 지시 내리기 프롬프트
 - 따라야 할 기본 지침 제공
- 꼬리 물기 프롬프트
 - 추론 설명 유도
 - 참조할 작업 목록 구축
 - 검토하고 피드백 수정 및 재작성

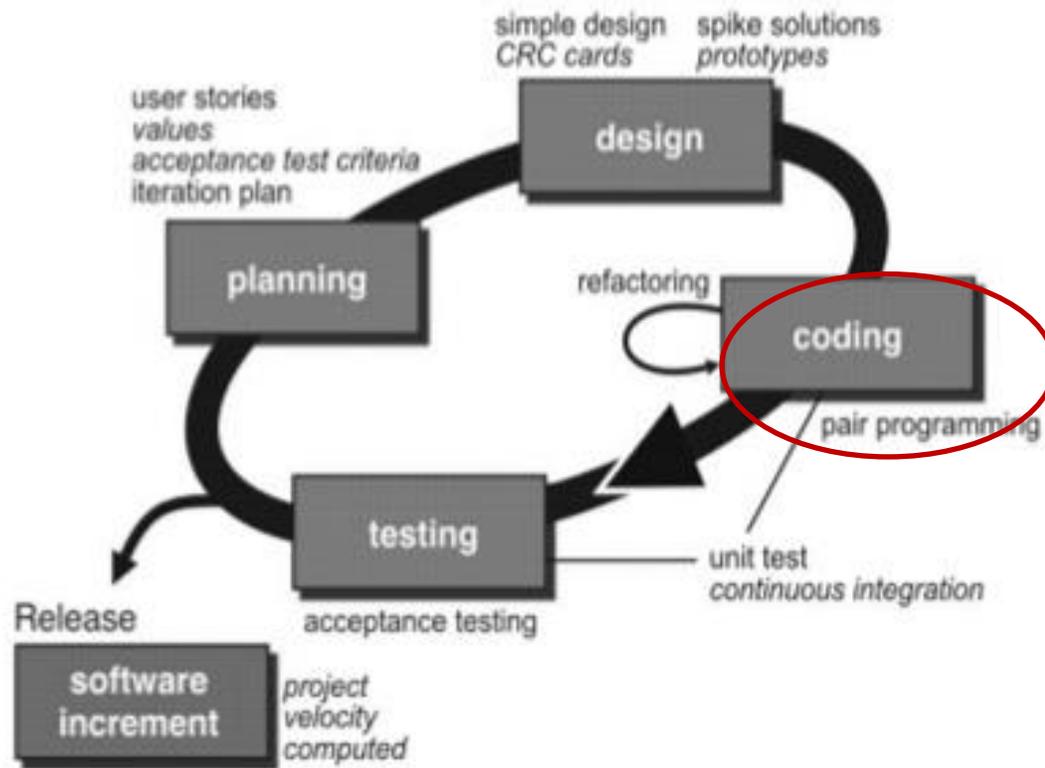
4. Code for one of the steps

- ① 단계별 코드 생성
- ② 점진적인 코드 조정 피드백

* 자료 : Martin Fowler 의 블로그 원문을 김도균 MS MVP 가 정리

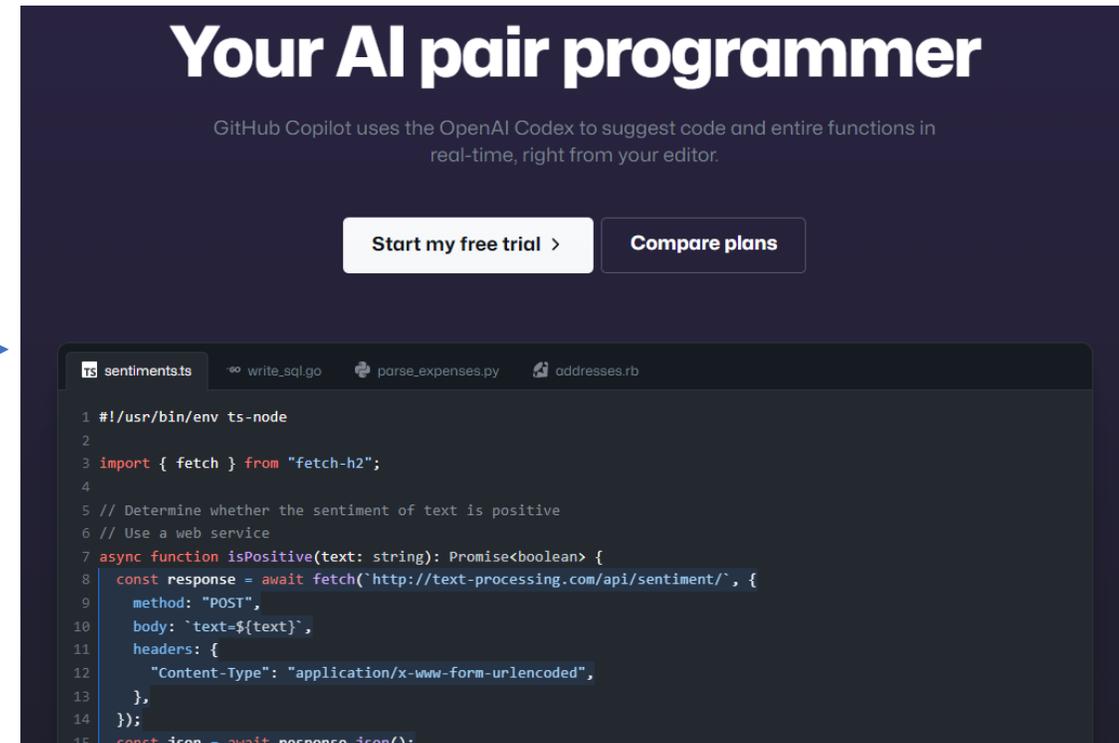
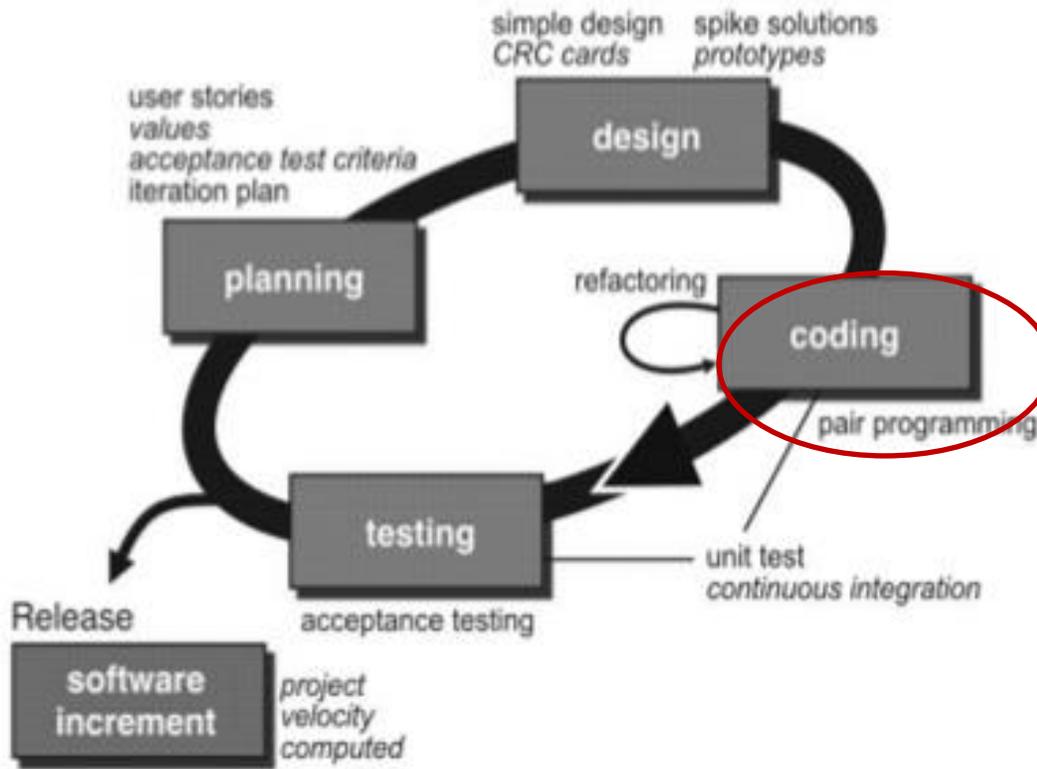
2. SW Delivery 의 가속 도구, 생성형AI

◆ Pair Programming 실현 도구



2. SW Delivery 의 가속 도구, 생성형AI

◆ Pair Programming 실현 도구 : 지치지 않는 AI



2. SW Delivery 의 가속 도구, 생성형AI

◆ Use Persona as a DevOps Team member

- ① 역할 만들기
 - 생성AI의 창의력을 끌어내기 위해 역할 부여
- ② 역할 이해하기
 - 맥락내에 이미 역할의 성격이 메모리에 올려져 있다.
 - 역할을 부르며 특성을 부여한다.
- ③ 사용 사례에 맞는 프롬프트 만들기
 - 특정 역할을 전제로 대화

프롬프트

아키텍트로서 상기 프로그램을 마이크로서비스로 분해해 줘

응답

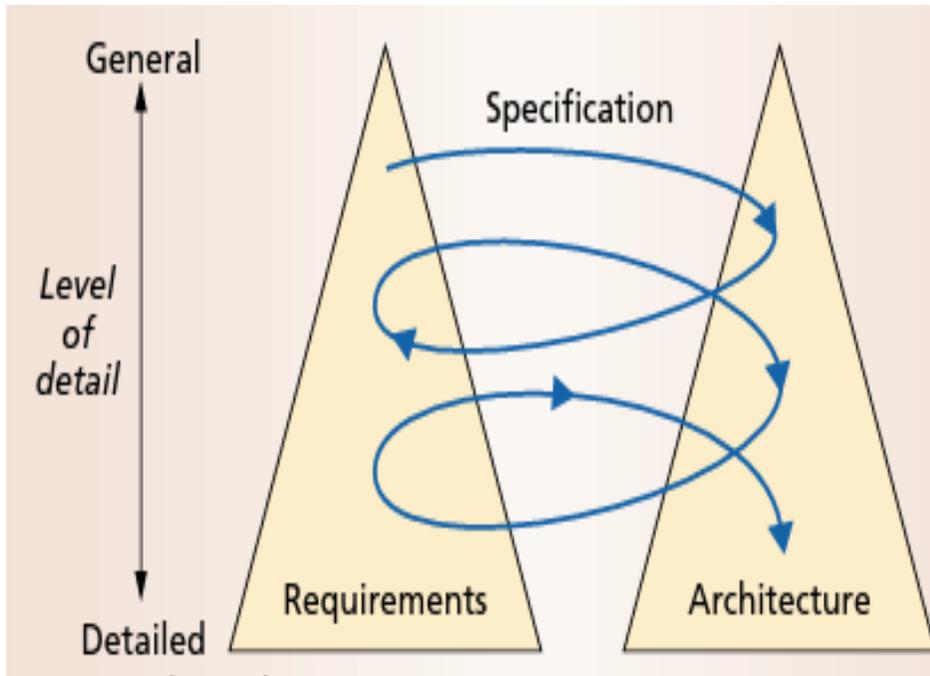
위의 프로그램을 마이크로서비스로 분해하면 모듈성, 확장성 및 유연성을 제공할 수 있습니다. 다음은 프로그램을 마이크로서비스로 분해하는 방법입니다.

- 1.인증 서비스: 사용자 인증 처리 및 로그인 자격 증명 관리를 담당합니다
- 2.차량등록 서비스: 주차관리시스템에 번호판 등록을 관리합니다.
- 3.차량조회 서비스: 주차관리시스템에서 차량정보 조회를 처리합니다.
- 4.알림 서비스: 클라이언트에게 알림 또는 완료 메시지를 보냅니다.
- 5.클라이언트 서비스: 사용자가 시스템과 상호 작용할 수 있는 클라이언트 쪽 인터페이스를 나타냅니다.

2. SW Delivery 의 가속 도구, 생성형AI

◆ AI의 SW엔지니어링 활용 원리 : 언어의 층위구조를 이해하는 트랜스포머 모델을 이용

< 명세화 >

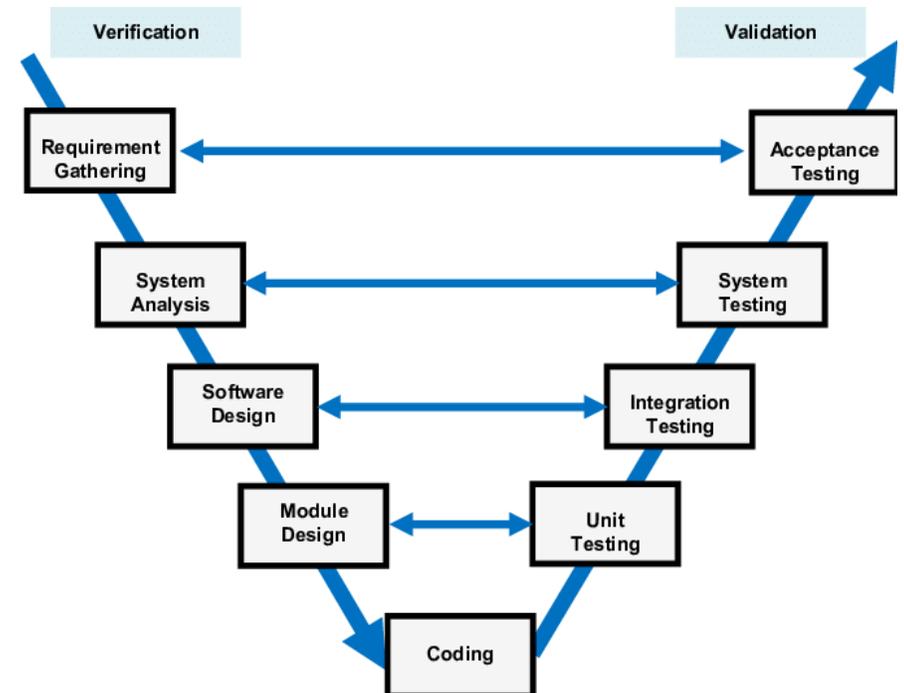


언어의 층위

개념어

구체어

< 구체화 >

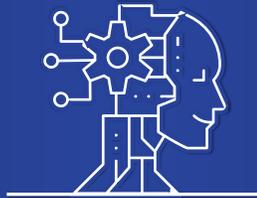


B. Nuseibeh, "Weaving together requirements and architectures," in *Computer*, vol. 34, no. 3, March 2001

SW공학의 V-모델

3. AI가 만드는 Software 2.0

Software 2.0 by Andrej Karpathy, 2017



AI는 새로운 형태의 SW

Neural networks are not just another classifier,
they represent the beginning of
a fundamental shift in how we write software.
They are Software 2.0.

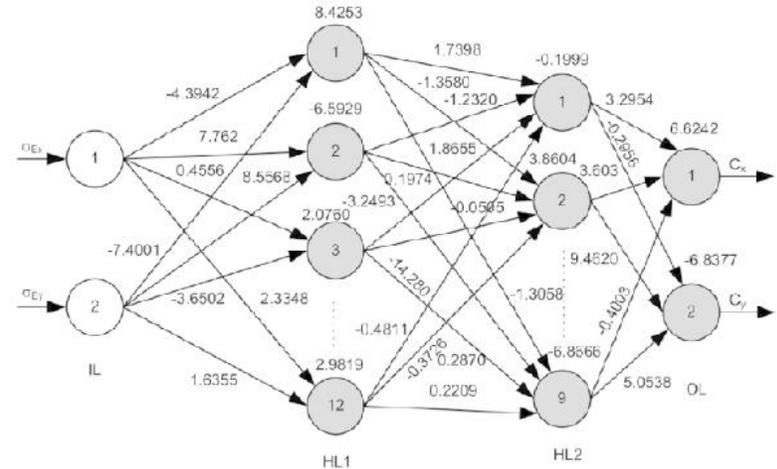


Andrej Karpathy

@karpathy

Gradient descent can write code better than you. I'm sorry.

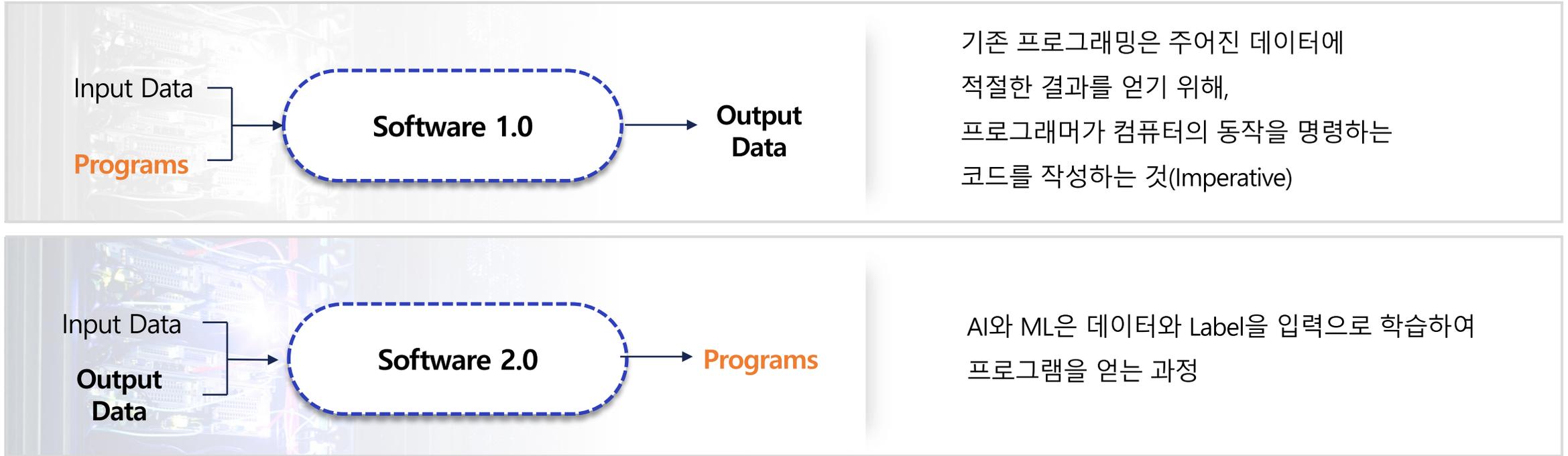
3:56 PM - 4 Aug 2017



Neural Networks

3. AI가 만드는 Software 2.0

◆ AI는 데이터로 만들어지는 SW



기존 프로그래밍은 주어진 데이터에 적절한 결과를 얻기 위해, 프로그래머가 컴퓨터의 동작을 명령하는 코드를 작성하는 것(Imperative)

AI와 ML은 데이터와 Label을 입력으로 학습하여 프로그램을 얻는 과정

Compiling == Optimization ?

The 2.0 programmers manually curate, maintain, massage, clean and label datasets; each labeled example literally programs the final systems because **the dataset gets compiled into Software 2.0 code via the optimization.**

- Angrej Karpathy, 2017

3. AI가 만드는 Software 2.0

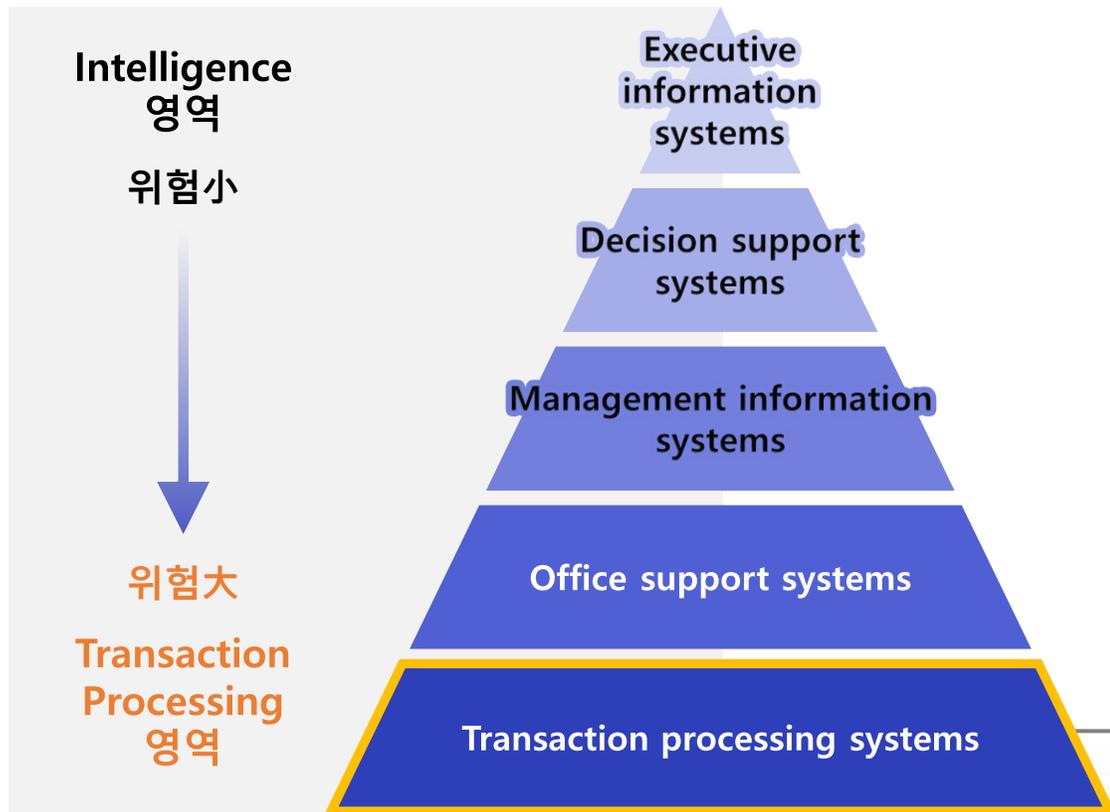
Software 1.0 의 기반 위에서 Software 2.0 이 혁신적 서비스를 창출

	Software 1.0	Software 2.0
개발 주기	<ul style="list-style-type: none"> • 디자인 > 개발 > 디버깅 > 제품 > 소프트웨어 유지보수 > 회귀 테스트 	<ul style="list-style-type: none"> ◆ 데이터 준비 > 모델 훈련 > 모델 추론 > 모델 드리프트/튜닝
개발단계	<ul style="list-style-type: none"> • 각 단계에는 결과물이 있음 	<ul style="list-style-type: none"> ◆ 전체 과정에서의 반복
인프라 요구사항	<ul style="list-style-type: none"> • 개발 인프라와 운영 인프라가 유사함 	<ul style="list-style-type: none"> ◆ 개발 인프라와 운영 인프라가 원칙적으로 다름
프로젝트 리드	<ul style="list-style-type: none"> • 프로젝트 리드는 기업 아키텍트가 이끔 	<ul style="list-style-type: none"> ◆ 프로젝트 리드는 데이터 과학자, 데이터 엔지니어 그리고 소프트웨어 엔지니어로 변경
기술과 도구	<ul style="list-style-type: none"> • 필요한 기술, 방법 및 인력이 제공됨 	<ul style="list-style-type: none"> ◆ 도구, 최상의 방법, 그리고 기술이 부족하며 진화 중
프로젝트 기간	<ul style="list-style-type: none"> • 예측 가능함 	<ul style="list-style-type: none"> ◆ 예측하기 어려움
조직의 성숙도	<ul style="list-style-type: none"> • 대규모 프로젝트에 대한 기존 조직적 경험이 상당함 	<ul style="list-style-type: none"> ◆ 대부분 프로젝트에서의 기계 학습 경험이 부족

3. AI가 만드는 Software 2.0

Software 2.0 의 전망 : Responsible AI 로

Enterprise SW의 구조



위험 : 신뢰성, 설명가능성, 기밀보호 등

Reliability Models can produce different answers to the same prompts, impeding the user's ability to assess the accuracy and reliability of outputs.

Fairness Models may generate algorithmic bias due to imperfect training data or decisions made by the engineers developing the models.

Explainability Generative AI relies on neural networks with billions of parameters, challenging our ability to explain how any given answer is produced.

Intellectual property Training data and model outputs can generate significant IP risks, including infringing on copyrighted, trademarked, patented, or otherwise legally protected materials.

Privacy Generative AI could also be used to create and disseminate malicious content such as disinformation, deepfakes, and hate speech.

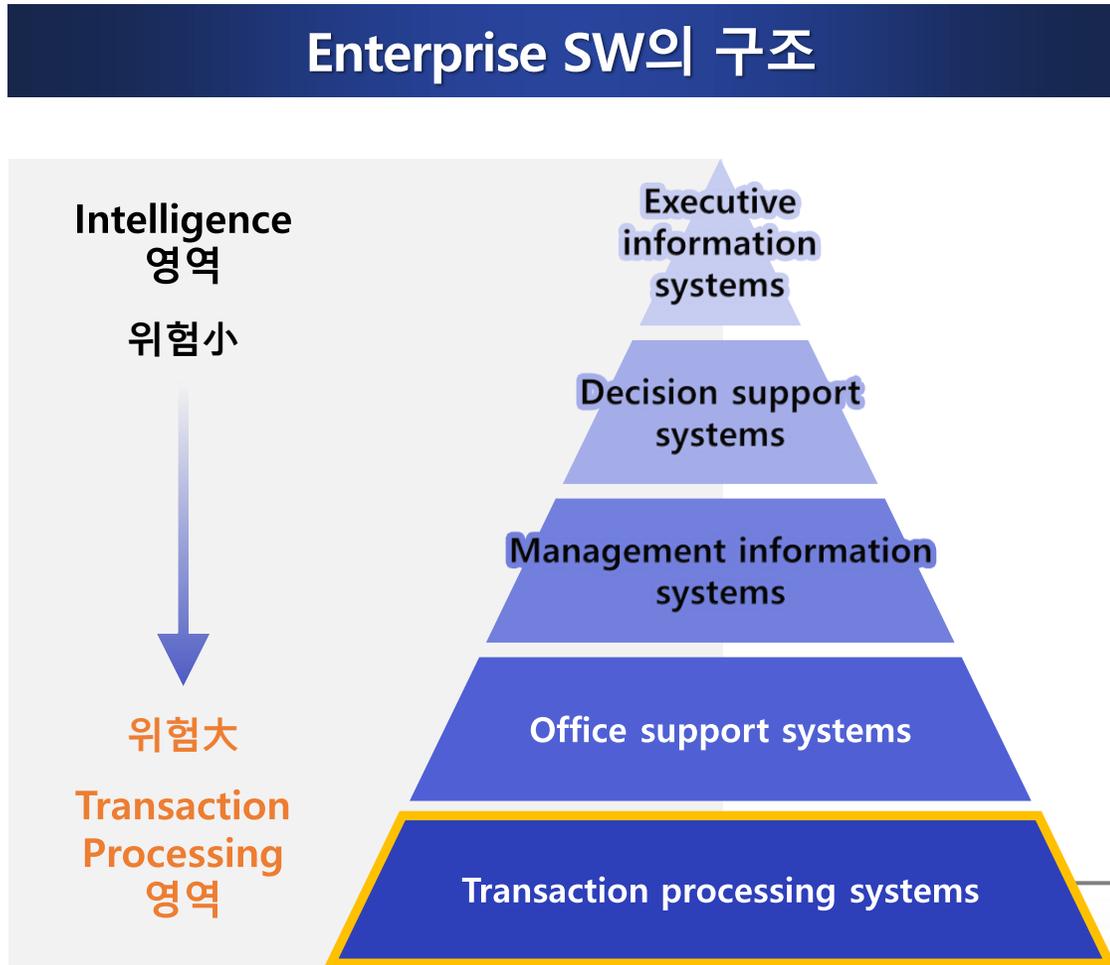
Security Generative AI may be used by bad actors to accelerate the sophistication and speed of cyberattacks. It also can be manipulated to provide malicious outputs.

Social and environmental impact The development and training of foundation models may lead to detrimental social and environmental consequences, including an increase in carbon emissions

* McKinsey(June 23), <The economic potential of generative AI>

3. AI가 만드는 Software 2.0

Software 2.0 의 전망 : Responsible AI 로



AI 의 보완재 : 플러그인

플러그인	기능
Expedia	여행 항공편, 숙박 예약 서비스
FiscalNote	글로벌 법률, 규제, 시장 데이터 서비스
Instacart	지역 소매점에서 주문 할 수 있는 전자상거래 서비스
Kayak	항공, 숙박, 렌터카 등 여행 관련 예약 서비스
Klarna	인터넷 쇼핑몰 검색, 최저가 검색
Milo Family AI	부모 안부 전달 서비스
OpenTable	레스토랑 정보 제공 및 예약 서비스
Shop	글로벌 브랜드의 수백만개의 상품 정보 검색
Speak	인공지능 기반의 외국어 학습 튜터
Wolfram	수리, 연산 기능 및 가공된 지식 및 실시간 정보 제공
Zapier	5천여개의 다른 앱들과 연동되어 업무 자동화 지원

* 플러그인 정리 : 유재흥&안성원 외, <생성AI의 부상과 산업의 변화>, SPRI이슈리포트

4. 생성형 AI 활용의 고려사항 : Vertical AI

◆ 데이터 보유 수준과 투자규모에 따라 AI활용 수준을 다르게 할 수 있음

Domain

화학/바이오

물성 예측, 분자구조 예측, 합성방식 제안 등

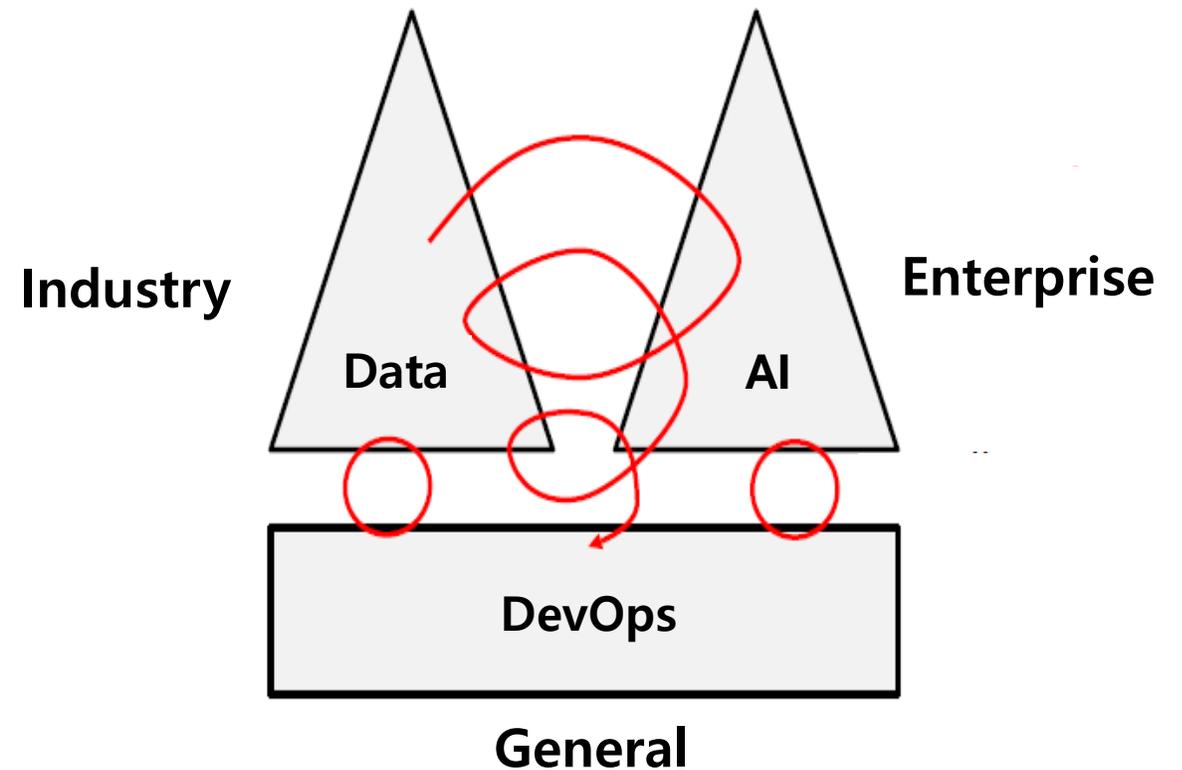
제조/금융

다양한 규정 문헌, 응대 매뉴얼, 전문가 Q&A

제약사 임상실험

병리조직 영상분석, 데이터 기반 피험자 선별

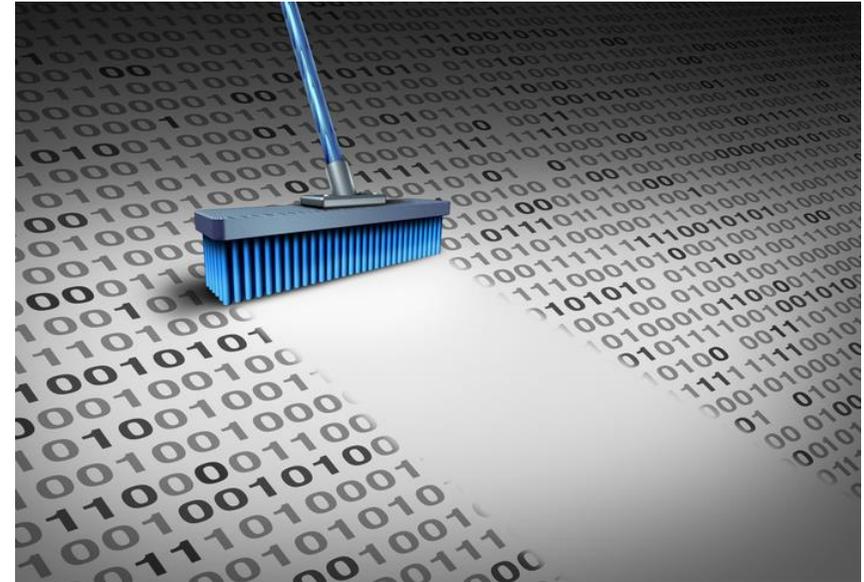
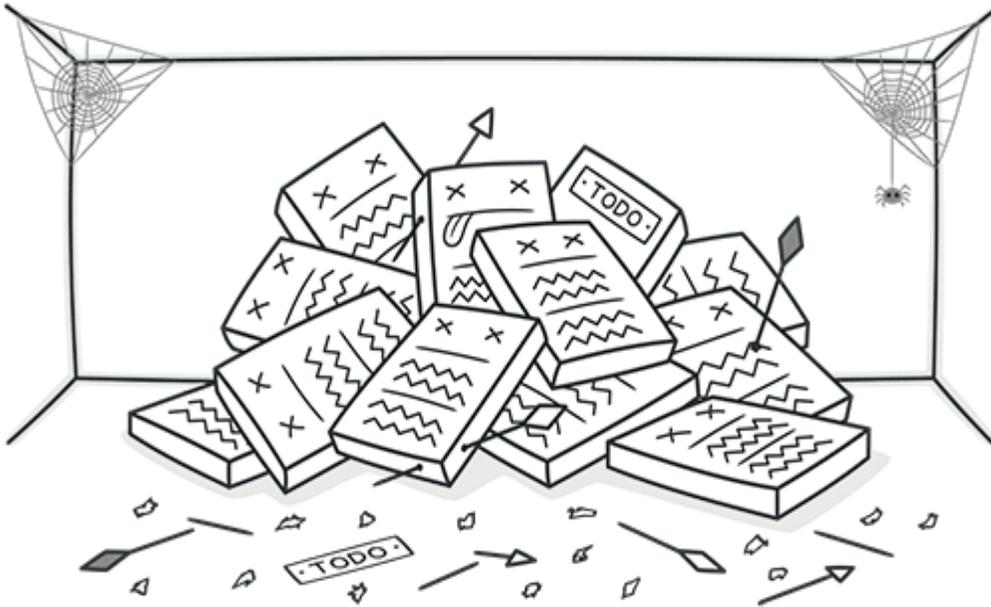
General / Industry / Enterprise AI



4. 생성형 AI 활용의 고려사항 : 자동생성 코드의 관리

◆ 기술 부채(technical debt)와 죽은 코드(dead code)

“생성AI 자동생성 코드의 양이 많아지면, 프로그램 개발자들의 업무는 쉬워지지만 기술책임자의 업무는 힘들어진다”
- EXL社 기술임원, WSJ인터뷰

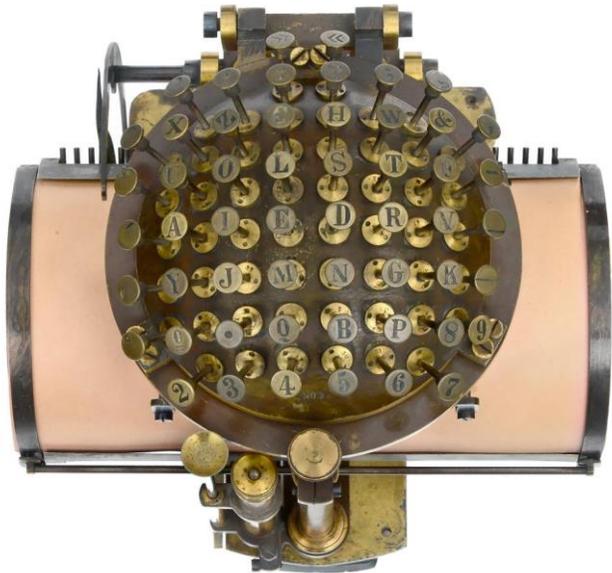


4. 생성형 AI 활용의 고려사항 : SW개발자의 장기 역량

◆ 도구로서의 AI가 SW개발자의 사고방식에 영향을 미칠 것

- 뇌의 가소성(neuroplasticity) : 뇌의 기능,역할,작동방식은 체계적인 반복으로 얼마든지 변화함

< 니체와 타자기, 1881 >



“우리의 글쓰기용 도구는 우리의 사고를 형성하는데 한 몫 하지”
- 철학자 니체

< 현대인과 인터넷, 2009 >



“종이 매체 또는 인터넷에서조차 장문의 기사를 읽는 능력을 잃어버렸다.”
- 미시간의대 병리학자 프리드먼

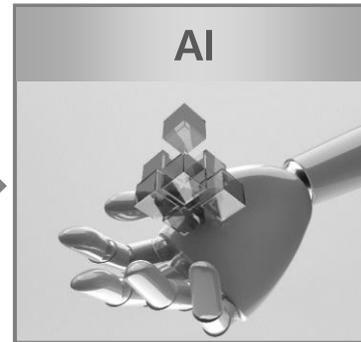
< SW개발자와 AI, 2023~ >



“컴퓨터에 의존하게 되면서 인공지능으로 변해버리는 것은 바로 우리의 지능이다.”
- IT미래학자 니콜라스 카

5. 나가며

AI-Augmented Software People



인간지능과 인공지능의 조화



AI로 인한 생산성 향상분을 고객 이해, 직원 관계, 창조적 업무에 투자

생성AI가 촉발하는 SW생산방식의 변화

감 사 합 니 다

유 호 석

hsy@spri.kr

031-739-7351

< 패널 토론 주제 >

산업별, 기업별 특화AI (Vertical AI)의 필요성과 가능성