



David Tung  
DevOpsDays Taipei 2024

# 在 DevOps 中加入 AI 賦能

~ 從自動化 Code Review 開始 ~



# About Me...

董大偉 [David@isRock.com.tw](mailto:David@isRock.com.tw)

- 光岩資訊 資深技術顧問
- 微軟認證講師(Microsoft Certified Trainer)
- 微軟技術社群區域總監(Microsoft Regional Director)
- 微軟最有價值專家(Microsoft MVP)
- LINE API Expert (LAE, LINE官方認證API專家)
- Professional Scrum Master (PSM I)
- 恆逸教育訓練中心特約講師、集英信誠資深顧問講師
- Regional Scrum Gathering Taipei 2021 大會講師
- Agile Summit 2023, 2022, 2019大會講師
- DevOpsDays Taipei 2023, 2022, 2021, 2019, 2017 大會講師
- 歷屆台灣微軟TechED, TechDays, DevDay 大會講師, China TechED大會講師
- TSMC、ViewSonic、遠傳電信、中華電信學院、世界先進、信義房屋、國防部、國衛院、渣打銀行、家扶基金會...Agile/Azure DevOps教育訓練顧問講師
- Visual Studio上市發表會巡迴講師





2021



2022



2023





自動化

文化

平台工程

SRE



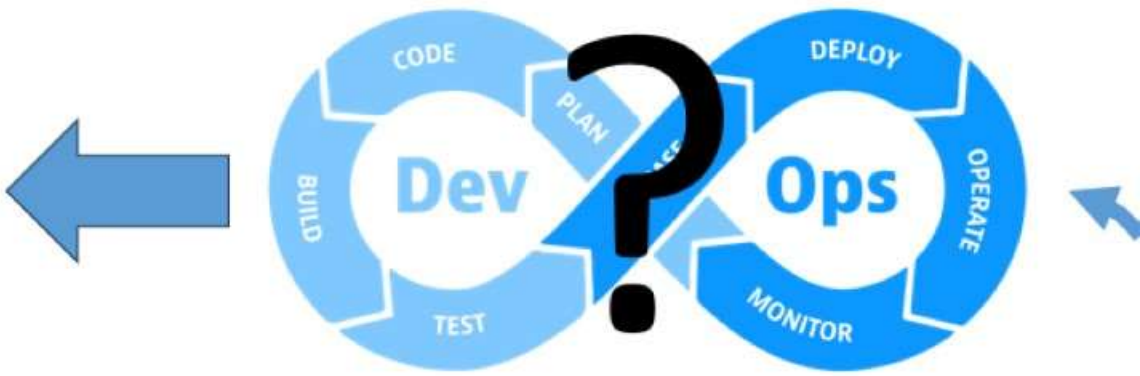
DevOpsDays Taipei



平台工

# 頻繁交付

(持續改善)



2021



2022



2023



4+12

個人與互動 基於 流程與工具 可用的軟體 基於 群體的文件 與客戶合作 基於 合於協同 風險變化 基於 敏捷計劃

1. 我們的責任是通過**頻繁交付**來滿足客戶的需求。
2. 我們**歡迎變更**，甚至在開發後期，敏捷利用變化來實踐客戶的競爭優勢。
3. **頻繁的交付成果**，從數周到數個月，持續驗證和改進。
4. 開發人員和業務人員必須在**專案中天天一起工作**。
5. 以**積極的個人(motivated individuals)**獨立運作，為他們提供所需的權限和資源，並授權他們能夠完成工作。
6. **面對面的溝通**是傳遞資訊最有效的方法。
7. **可以用的軟體**是敏捷交付的主要目標。
8. 敏捷程序提供**可持續的開發**，幫助者、開發者，及使用者確保不斷地獲得穩定的交付。
9. **持續追求卓越**的技術與優良的設計，以優化敏捷性。
10. 簡單性 - 最大化完成工作量的簡便 - 至關重要。**(減少浪費)**
11. 良好的架構、需求與設計，都來自於能**與組織**的溝通。
12. 團隊定期自省也可提高效率，並據之調整地調整與修正自己的行為。**(持續改善)**

~~軟體?~~  
價值!

敏捷是DevOps的**目標**  
DevOps是敏捷的**手段**

# 頻繁交付

(持續改善)

## 功能

Specification  
User Story  
Acceptance Criteria  
Test Case  
...

## 安全性

DevSecOps  
DAST/Pen. Test  
...

## 品質

Unit Test  
Pair Programming  
Code Review  
Code Scan/SAST  
...

個人與文藝	晏殊	送程周王員
可憐的軟體	晏殊	詞畫的文雅
醉臥不合作	晏殊	食肉滋味
思舊情懷	晏殊	讀書計劃

- 1. 將目的與責任明確化並透過成員間溝通與討論來達成共識，而達成共識是成功的第一步。
- 2. 他們都對計畫有參與感，甚至能開發新點，對整個計畫產生高度責任的參與感。
- 3. 規劃好交付物清單，以激勵計畫進行，評定領導者表現。
- 4. 與他人共同負責任與計畫清楚定義其共同一致的工作。
- 5. 以積極的人員(motivated individuals)達成目標，為他們提供有關的激勵和支援，並協助他們克服困難工作。
- 6. 對計劃的溝通是促進達成內部的工作。
- 7. 寫下共同的目標是衡量進展的重要標準。
- 8. 目標設定不難，但如何選擇目標、資訊來源、評定標準，並使團隊成員對目標不致感到挫折是困難的。
- 9. 持續檢查新目標的達成與舊目標的檢討，以達成目標性。
- 10. 目標性、溝通力與達成工作目標的關係，至關重要。(劉永福)
- 11. 溝通的障礙：語言和時間，都與時間和語言有關的障礙。
- 12. 溝通的障礙也包含語言障礙，並透過溝通型態與修正語言的行為。(劉永福)

# 頻繁交付

# 功能

- Specification
- User Story
- Acceptance Criteria
- Test Case

# 安全性

- DevSecOps
- DAST/Pen. Test
- ...

品質

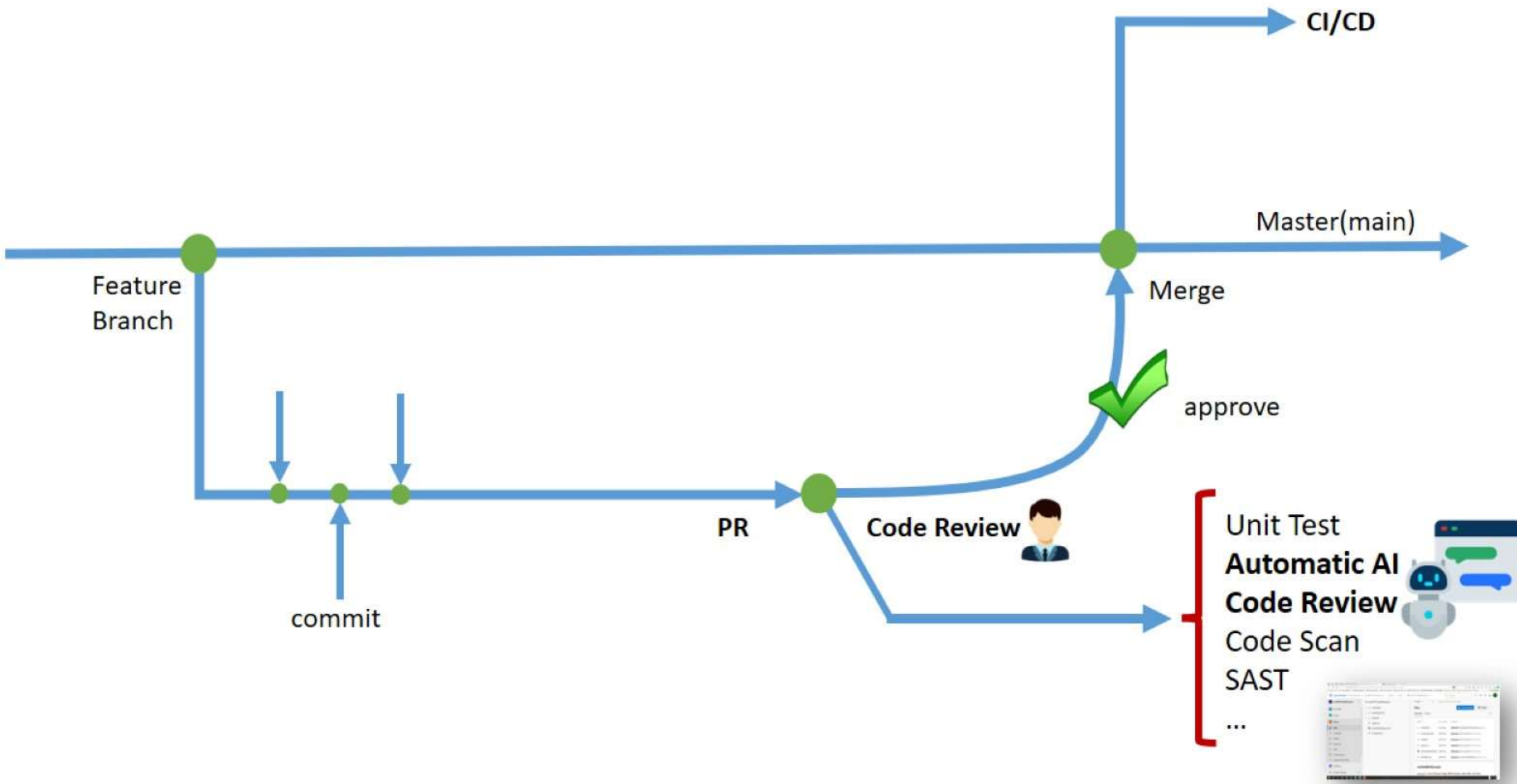
- Unit Test
- Pair Programming
- Code Review** ✓
- Code Scan/SAST
- ...

~~軟體？~~  
~~軟體！~~  
價值！

# 健康小教室

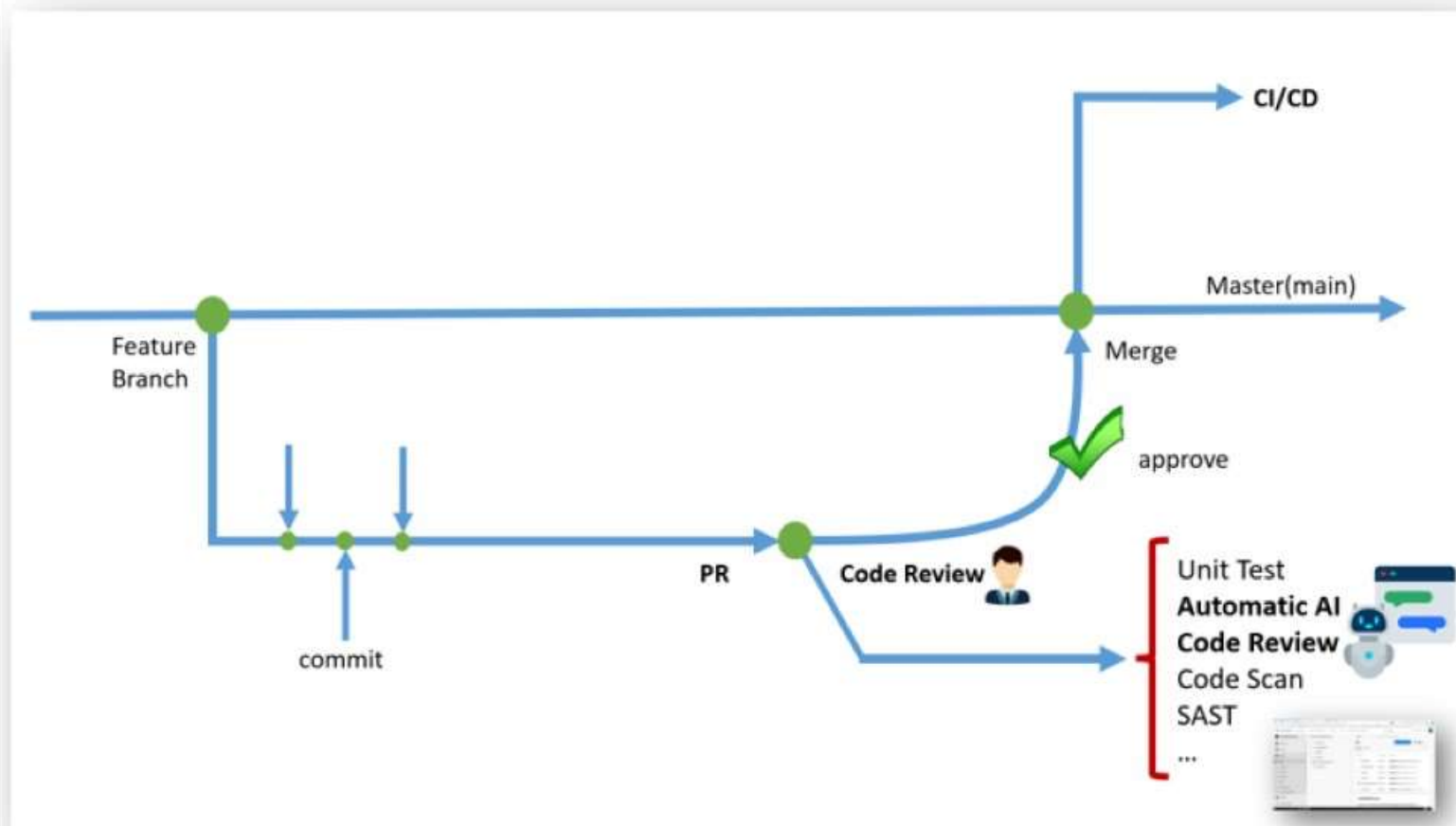
$$\text{BMI} = \frac{\text{體重}_{(\text{kg})}}{\text{身高}_{(\text{m})}^2} = \frac{70_{(\text{kg})}}{1.7_{(\text{m})}^2} = 24.22145\dots$$





# 使用 AI 提高 Code Review 效率

- ✓ 由 AI 自動進行初步的 Code Review
- ✓ 自動找出基本問題
- ✓ 減少 Reviewer 檢視時間
- ✓ 避免 Reviewer 檢視盲點
- ✓ 減少開發人員 等待/閒置 時間
- ✓ 溫馨提醒：**避免過度依賴**



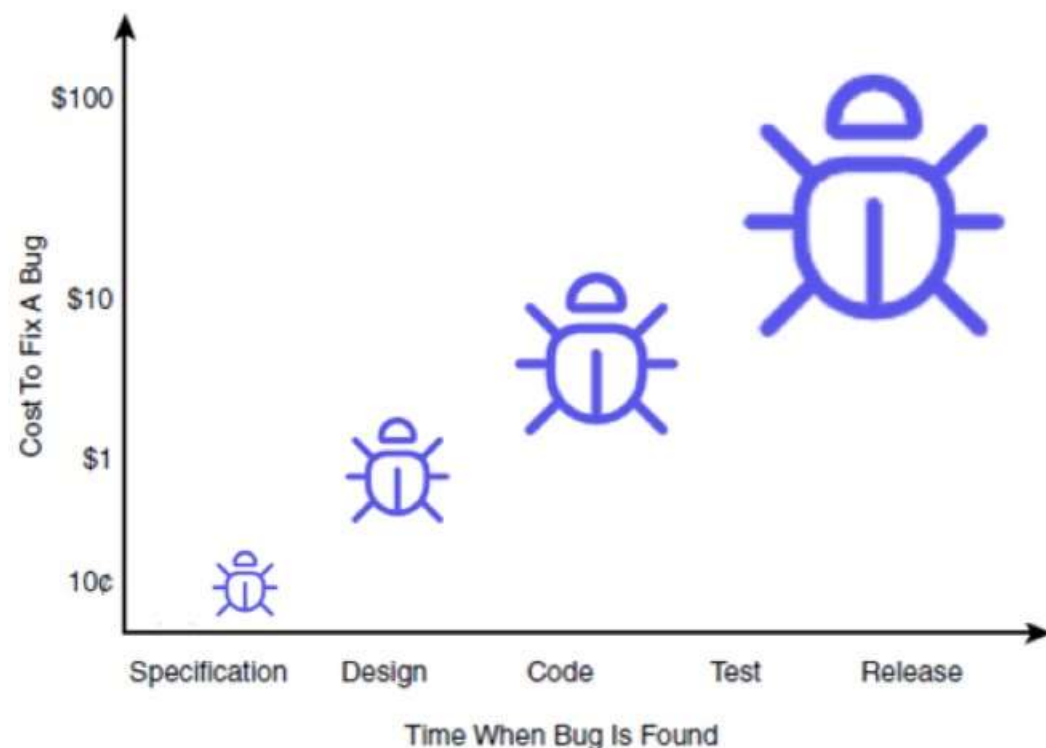
# DevOps小教室

對了，是不是有個詞叫作  
『測試左移』？



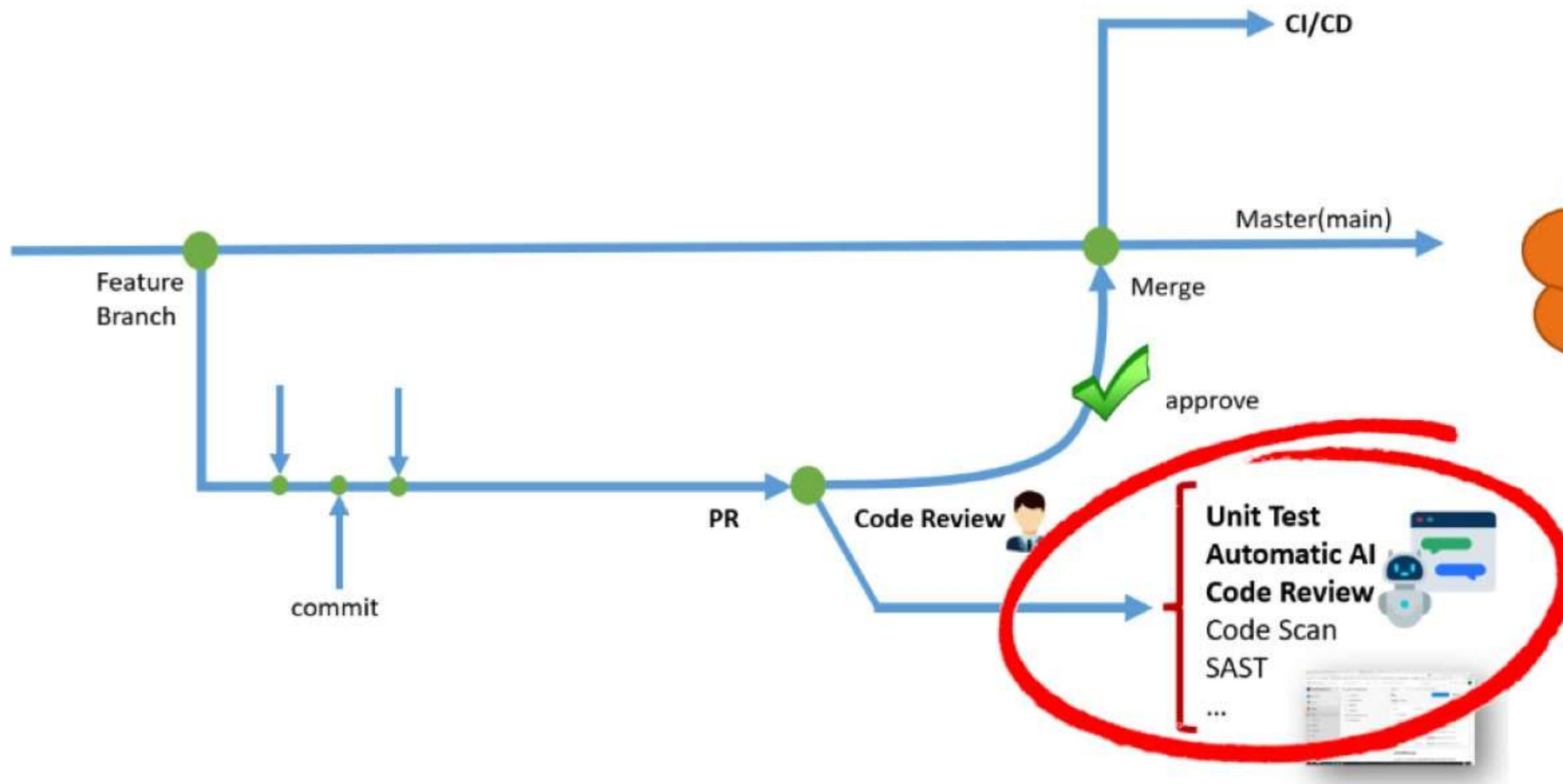
# 提早解決Bugs

- ✓ 測試左移
- ✓ 早期發現、早期治療



*Extracted from the book: Software Testing by Ron Patton*

一個Bug在運行階段被發現才處理所帶來的成本，遠高於開發階段。

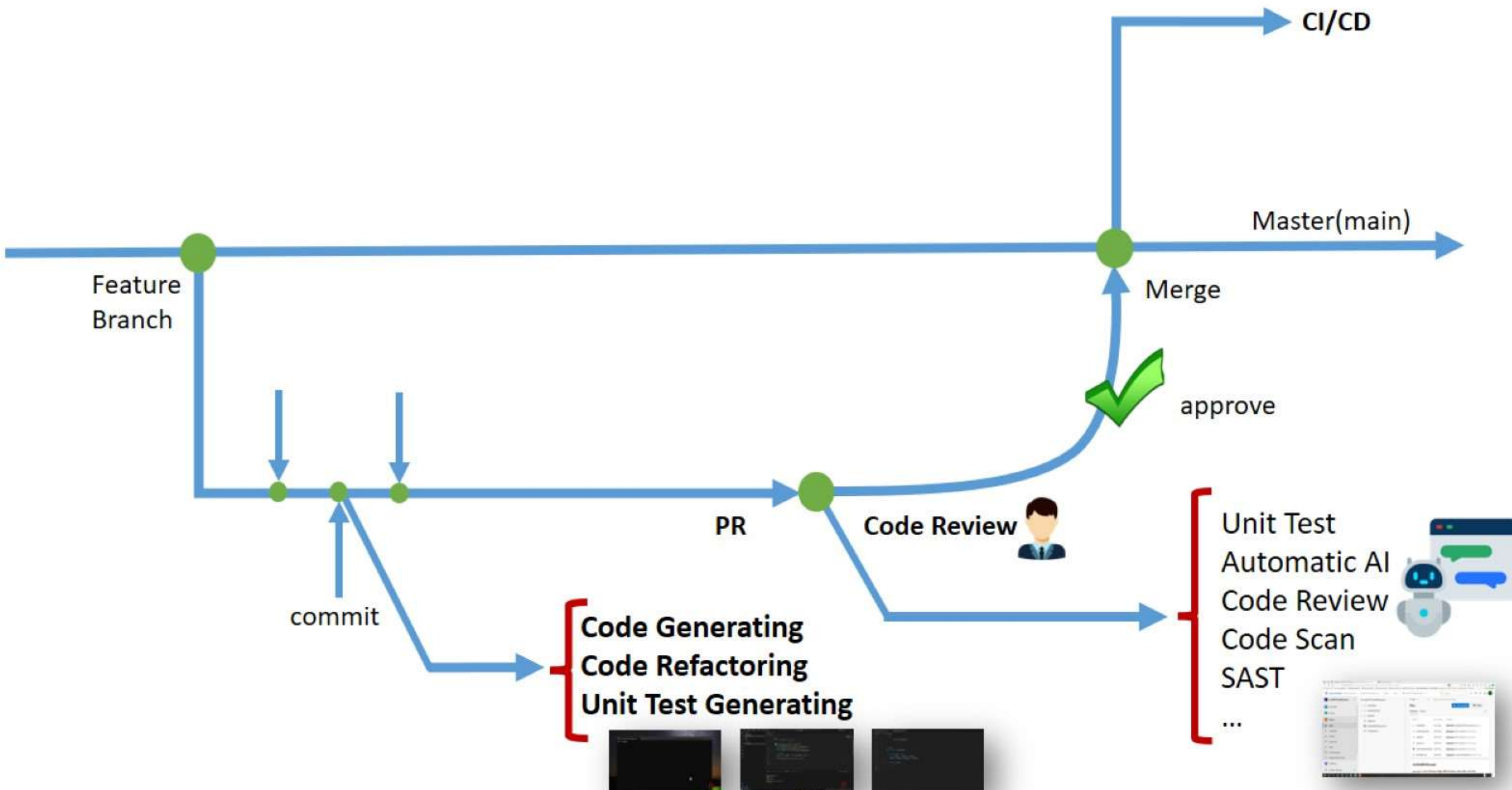


能不能再  
左移一點?



# Automatic AI Code Review

使用 AI 提高 Code Review 的效率

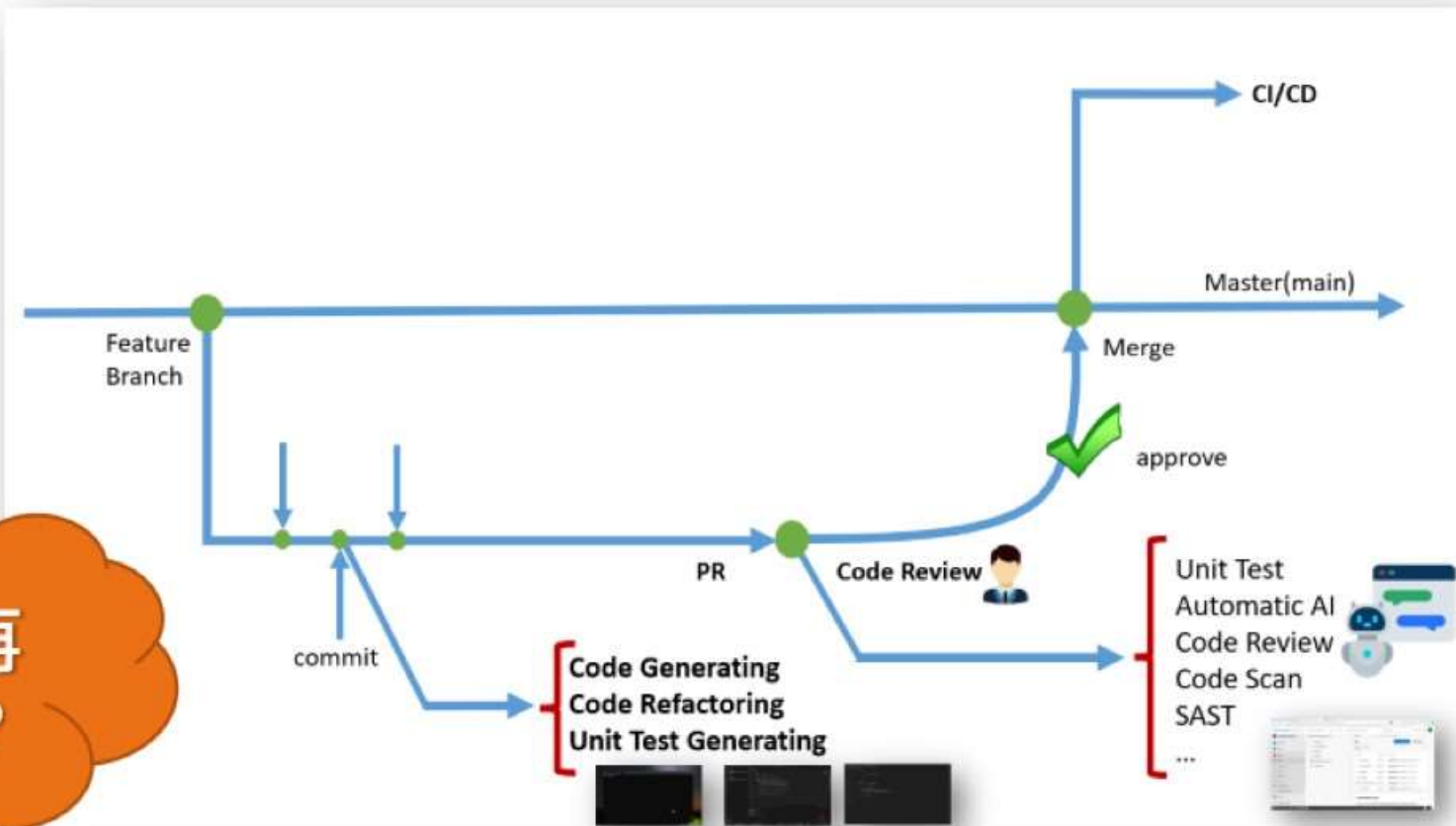




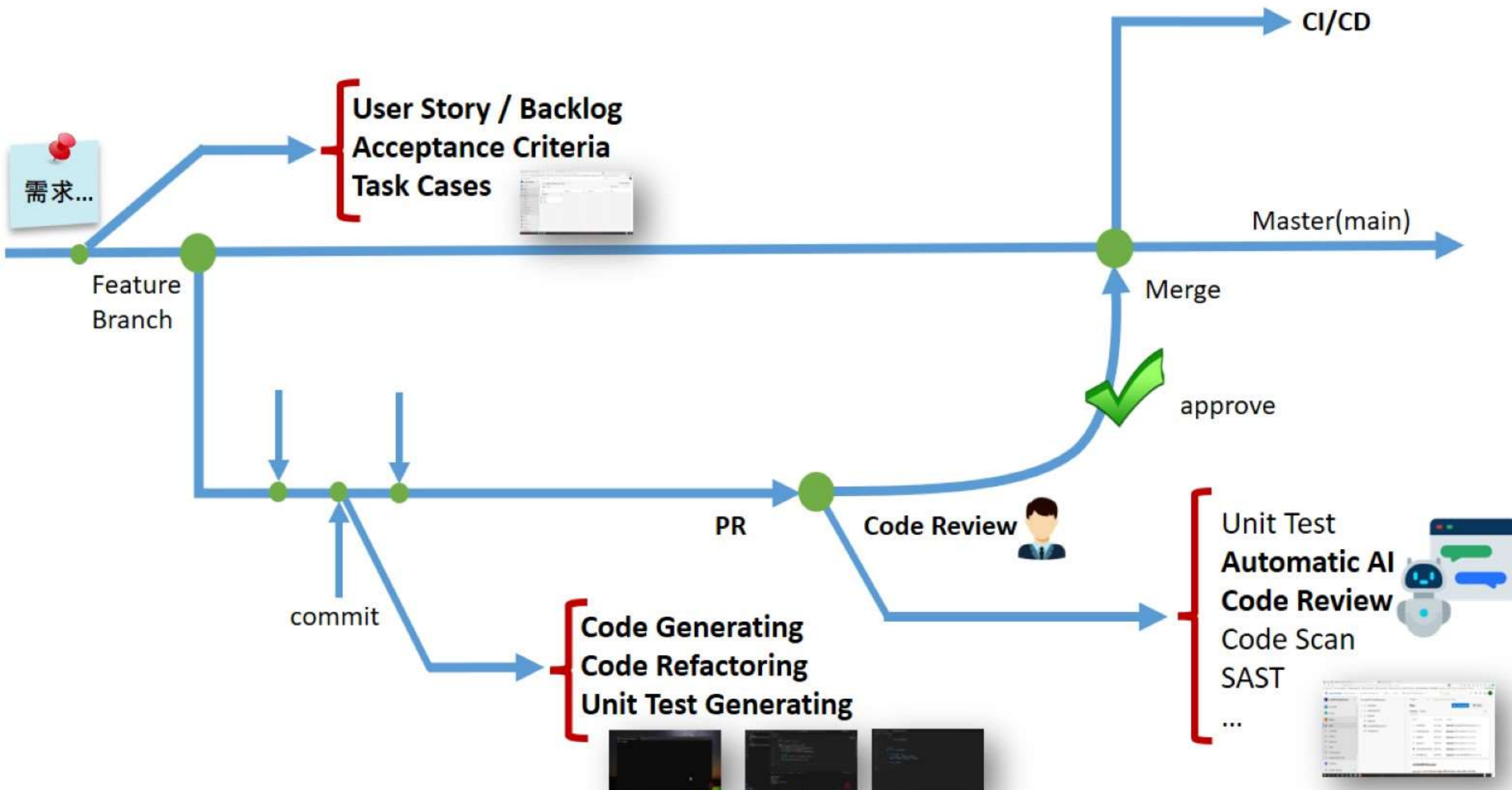
# 靠左一點 提早解決潛在Bugs

- ✓ AI生成程式碼
- ✓ AI協助重構程式碼
- ✓ AI生成單元測試

還能不能再  
左移一點?



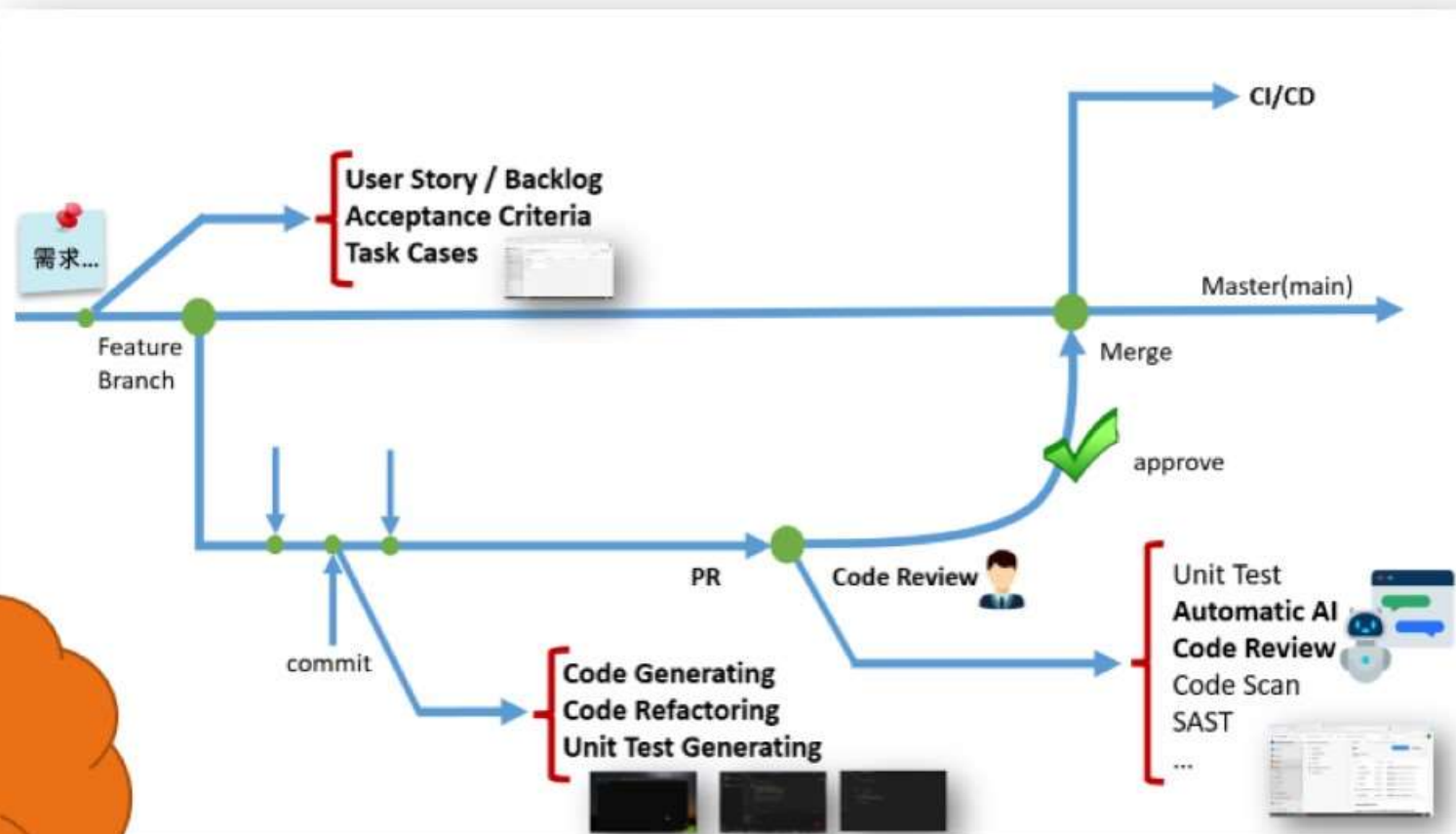
一個Bug在運行階段被發現才處理所帶來的成本，遠高於開發階段。透過AI生成高品質程式碼，以及單元測試，讓開發人員能夠在更早期提升程式碼品質，避免潛在Bugs產生...



# 再左一點 提早解決Bugs

- ✓ User Story / Backlog 生成
- ✓ Acceptance Criteria 生成
- ✓ Task Cases(測試案例) 生成

許多用戶總是講不清楚  
想要什麼？



透過 AI 生成高品質的需求，協助 PO/PM/SA 與用戶溝通釐清規格，在盡可能早期開始探索需求，避免重工與浪費



# DevOps小教室

都講到需求了，不談談文件怎麼行？

開發前(需求文件)

開發後(維護文件)

撰寫文件

User Story / Backlog  
Acceptance Criteria  
Task Cases

CI/CD  
Deploy

Master(main)

Feature  
Branch

Merge

approve

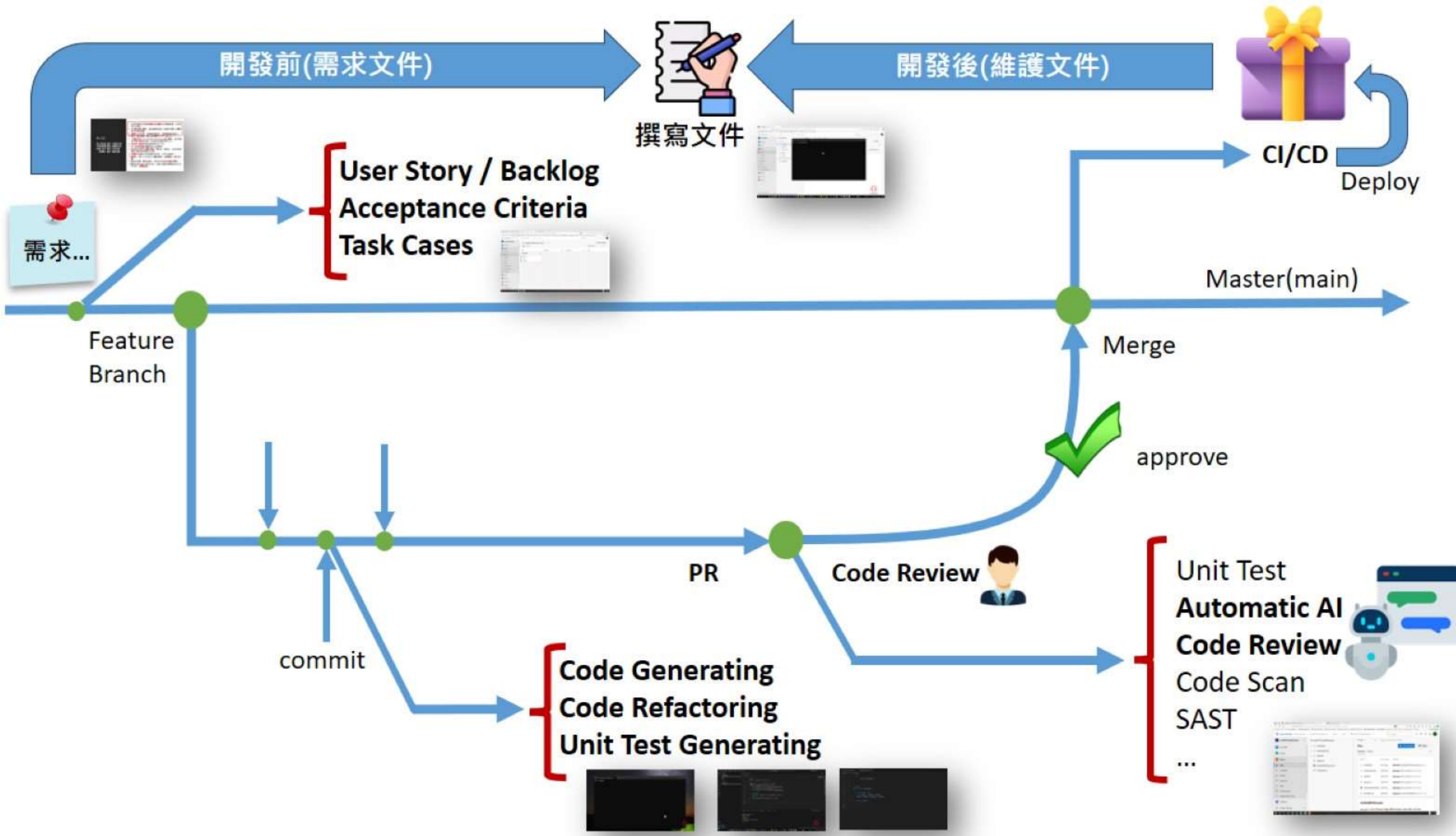
PR

Code Review

commit

Code Generating  
Code Refactoring  
Unit Test Generating

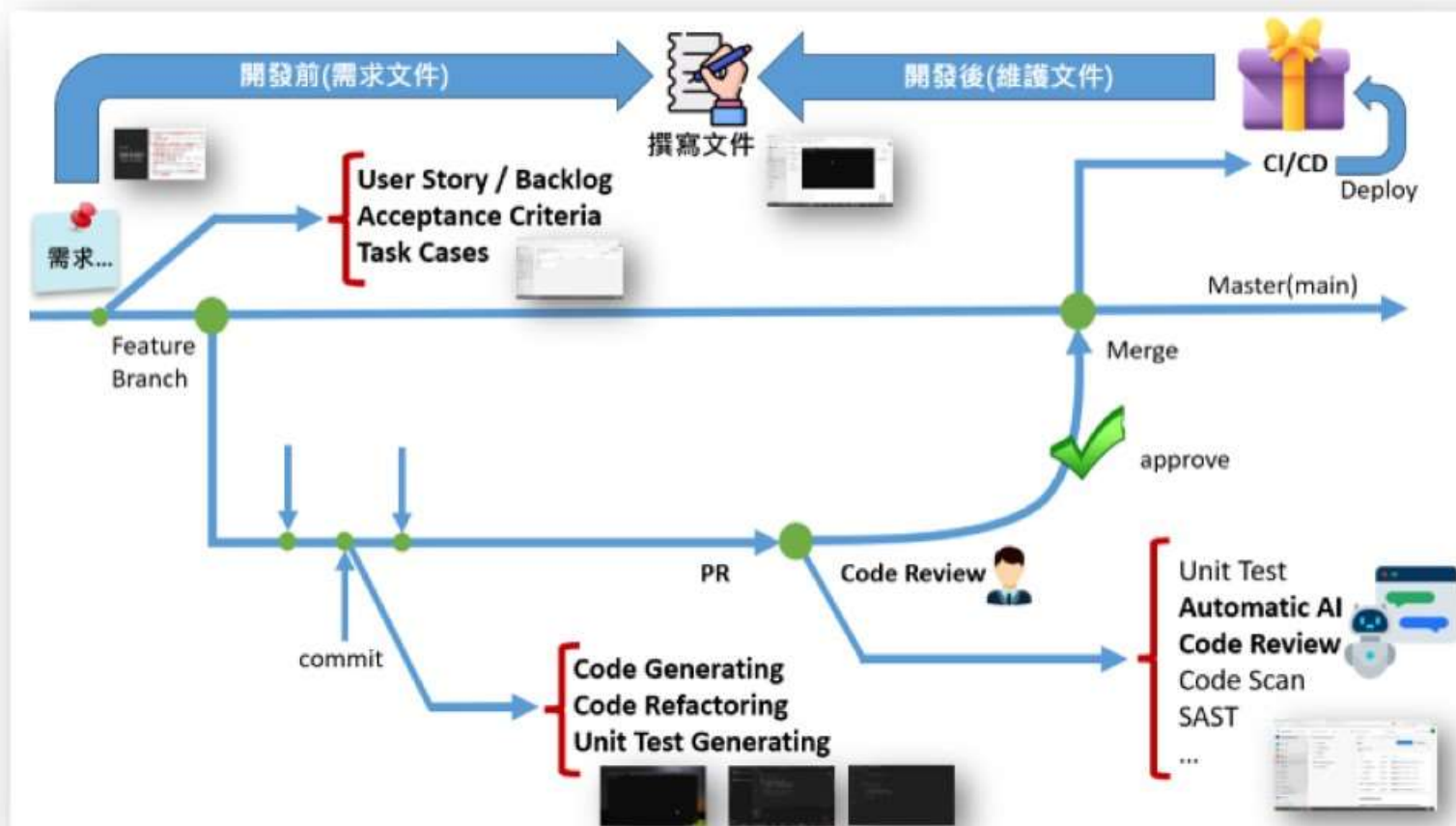
Unit Test  
Automatic AI  
Code Review  
Code Scan  
SAST  
...



# AI系統文件生成

- ✓ 程式碼/資料結構 說明
- ✓ 程式碼文件 生成
- ✓ 開發與維運 紀錄

幹嘛寫文件？



別裝傻，我認識的開發人員當中，程式碼寫得愈好的，就愈不喜歡寫文件。沒有例外。(至少我認識的人裡面，沒有例外)，透過 AI 生成高品質的維運文件，我猜沒有人會反對。



# DevOps小教室

都做到這個地步了，那有  
沒有可能乾脆...



# GitHub Workspace

從提出需求開始、到進行分析、到系統設計、到開發實作，全程由 AI 自動完成，會是一種怎樣的開發體驗？





短時間內

👉 軟體開發不會消失，但一定會被改變...  
所以，

- 拒絕使用所有AI工具，保障開發人員工作權 (X)
- 嘗試透過 AI 工具簡化你的開發流程&縮短時間 (O)
- 想想如何有效利用節省下來的時間(O)

## 結論感想

👉 花在實作上的時間會愈來愈少...所以，

- 與其學習框架不如學好架構 (O)
- 專研奇技淫巧不如扎實練好基本功(O)

👉 使用AI可能有風險，但沒人會回頭了...  
所以，

- AI給的建議照單全收、來者不拒 (X)
- 辨別能力很值錢，經驗是關鍵 (O)



那我們就到這裡...

LINE官方帳號...

@studyhost

<https://wwjd.tw/409k687>



用LINE掃QR Code