Manual QA

Техніки тест-дизайну. Частина 2

Тема уроку

Техніки тест-дизайну. Частина 2

План уроку

- Позитивні та Негативні тести
- Black box
  - Попарне тестування (Pairwise testing)
  - Тестування переходів станів (State transition testing)
  - Користувацькі сценарії (Use case testing)
- Experience based
  - Передбачення помилки (Error guessing)
  - Дослідницьке тестування (Exploratory testing)
  - Інтуїтивне тестування (Ad-hoc testing)
- Як писати гарні тест-кейси

Що було минулого разу

План уроку

- Black/Gray/White box
- Класи еквівалентності (Equivalence Class)
- Граничні значення (Boundary Value)
- Таблиці рішень (Decision Tables)
- Тестування операторів (Statement testing)
- Тестування умов (Condition testing)
- Тестування рішень/гілок (Decision/branch testing)
- Як писати гарні ТК
- Практика написання простих документів

Аспекти проектування тест-кейсів

Позитивні та Негативні тести

Позитивні та Негативні тести

Позитивне тестування – дозволяє перевірити, чи працює система в нормальних умовах так, як задумувалося.

Іншими словами:

Позитивні тести – перевіряють, що те, що має працювати - працює.

Негативне тестування – гарантує, що програма продовжить роботу у разі помилки або непередбаченої поведінки.

Іншими словами:

Негативні тести – перевіряють реакцію програми на невалідні дані/поведінку.

Позитивні та Негативні тести

Рекомендація:

Починайте з простих очевидних тест-кейсів, які перевіряють працездатність основних функцій програми. А потім розширюйте покриття більш складними сценаріями.

Рекомендована послідовність розробки та виконання тест-кейсів:

- прості позитивні;
- прості негативні;
- складні позитивні;
- складні негативні.

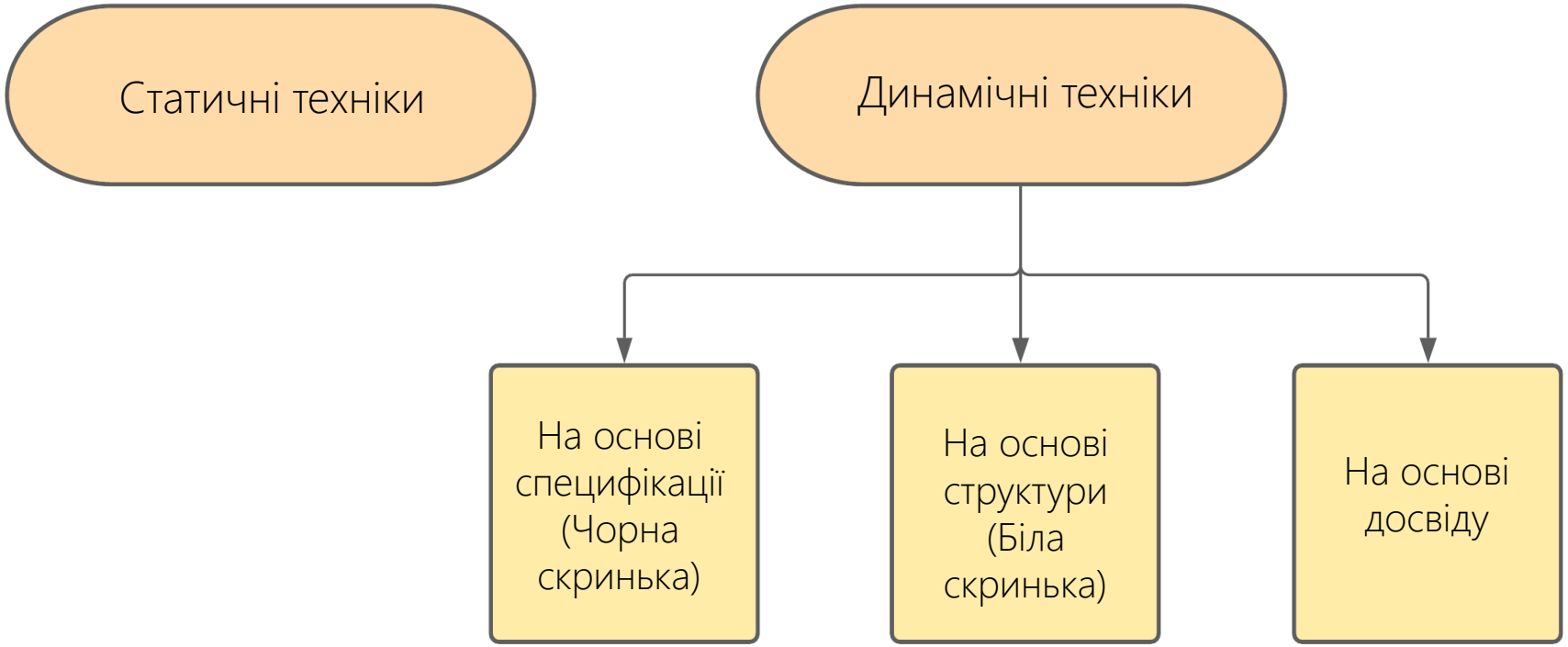
Техніки тест-дизайну

Test Design

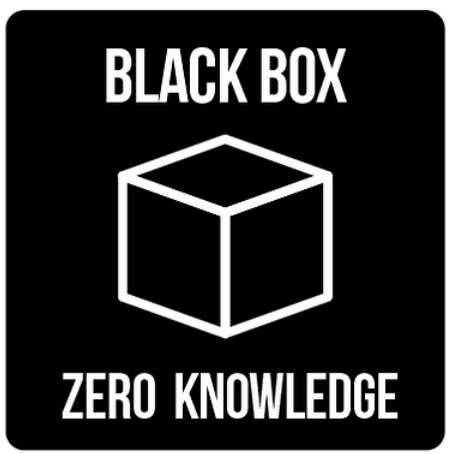
Тест-дизайн – це один з етапів процесу розробки програмного забезпечення, етап придумування, розробки та впорядкування тестів у набори на основі аналізу вимог та функціональності продукту.

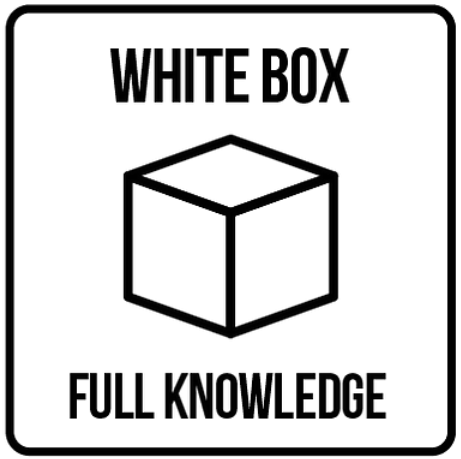
Техніка тест-дизайну – це метод створення тестів.

Test Design. Техніки



Test Design. Динамічні техніки





Test Design. На основі специфікації

- Класи еквівалентності (Equivalence partitioning)
- Граничні значення (Boundary value analysis)
- Таблиця прийняття рішень (Decision tables)
- Попарне тестування (Pairwise Testing)
- Діаграма переходів станів (State transition testing)
- Користувацькі сценарії (Use case testing)

Попарне тестування (Pairwise Testing)

Попарне тестування – техніка тестування, в якій замість перевірки всіх можливих комбінацій значень усіх параметрів перевіряються тільки комбінації значень кожної пари параметрів.

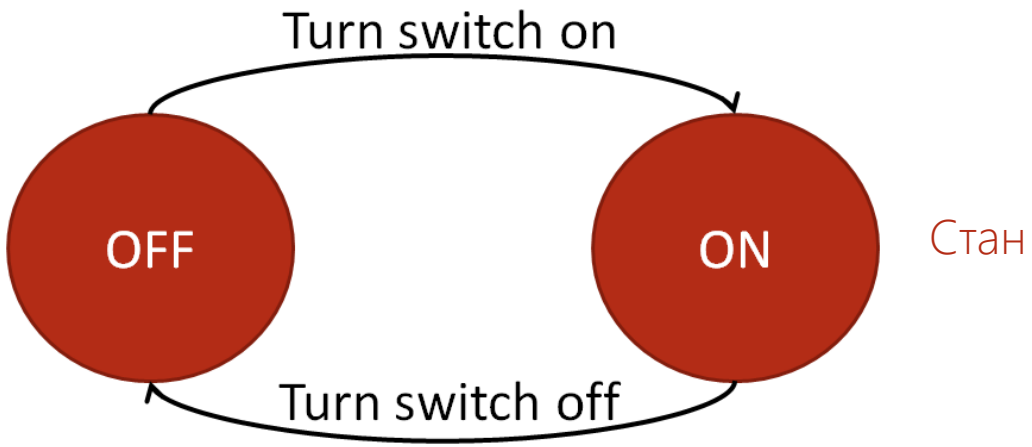
Google Chrome	Windows	RU
Google Chrome	Windows	EN
Google Chrome	Linux	RU
Google Chrome	Linux	EN
Opera	Windows	RU
Opera	Windows	EN
Opera	Linux	RU
Opera	Linux	EN

Google Chrome	Windows	RU
Google Chrome	Linux	EN
Opera	Windows	EN
Opera	Linux	RU

Онлайн інструмент 1
Онлайн інструмент 2

Тестування переходів станів (State transition testing)

Схема станів і переходів – техніка для візуалізації ТЗ. Вона наочно показує, як об'єкт переходить з одного стану до іншого.

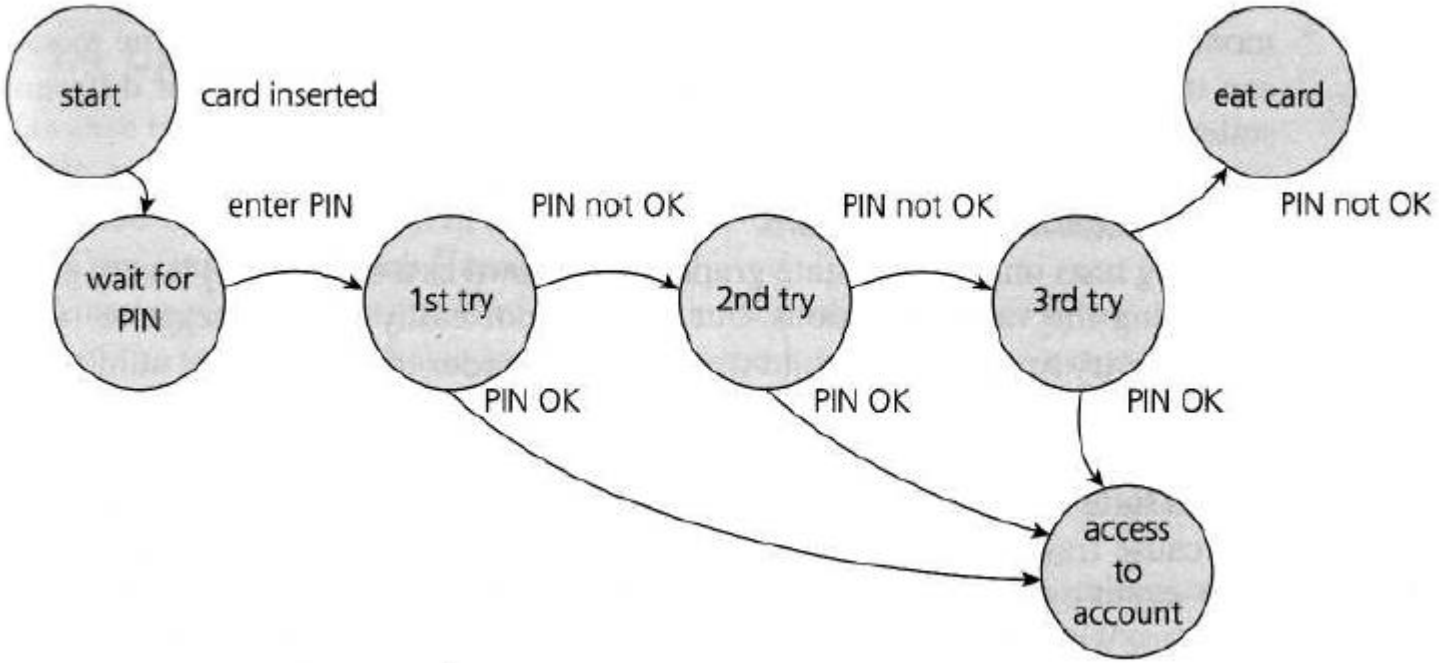


Стан

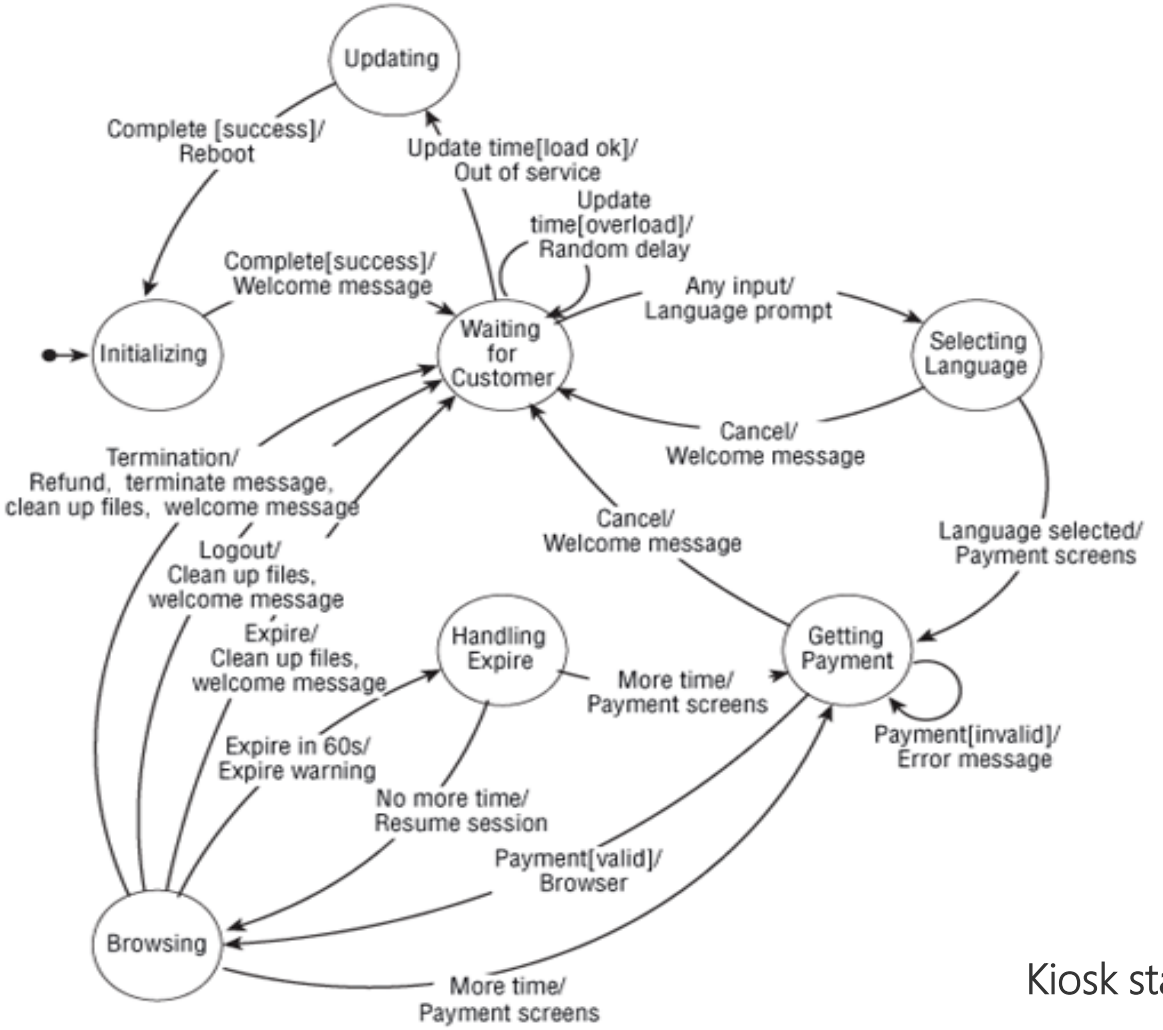
Перехід

Діаграма переходів станів

Схема станів і переходів для процесу логіну в банкомат:



Діаграма переходів станів



Kiosk state transition diagram

Користувацькі сценарії (Use case testing)

Use case – це сценарії, що описують те, як actor (зазвичай людина, але може бути й інша система) користується системою для досягнення певної мети. Варіанти використання описуються з точки зору користувача, а не системи.

Користувацькі сценарії (Use case testing)

Use case описує, як актор намагається досягти будь-якої мети за допомогою системи або застосунку.

Структура:

- ціль;
- актори;
- попередні умови (необов'язково);
- основний сценарій;
- розширення (альтернативний сценарій).

Користувацькі сценарії (Use case testing)

Приклад: Реєстрація

Сайт має надавати можливість реєстрації. Користувач має заповнити форму для реєстрації. Сайт має надіслати електронного листа з підтвердженням після успішної реєстрації користувача.

Те саме у форматі Use Case:

UC: реєстрація на сайті
Актор: Користувач
Умова: користувач знаходиться на сторінці входу
Основний сценарій:

1. Користувач натискає кнопку "Зареєструватися".
2. На сайті відображається форма реєстрації.
3. Користувач заповнює форму та підтверджує реєстрацію.
4. Сайт підтверджує правильність заповнення форми.
5. Сайт реєструє користувача та надсилає йому на пошту листа з підтвердженням реєстрації.

Test Design. На основі досвіду

- Передбачення помилки (Error guessing)
- Дослідницьке тестування (Exploratory testing)
- Інтуїтивне тестування (Ad-hoc testing)

Передбачення помилки (Error guessing)

Передбачення помилки (Error guessing) – підхід, при якому тестувальник думає над тим, яких помилок можна допустити в процесі розробки, а також визначає шляхи їх появи, використовуючи інтуїцію, знання та досвід.

Дослідницьке тестування (Exploratory testing)

Дослідницьке тестування (Exploratory testing) – це одночасне вивчення програмного продукту, проектування тестів та їх виконання.

Це неформальний метод проектування тестів, при якому тестувальник активно контролює проектування тестів у той час, як ці тести виконуються, та використовує отриману під час тестування інформацію для проектування нових тестів

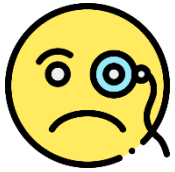
Інтуїтивне тестування (Ad-hoc testing)

Інтуїтивне тестування (Ad-hoc testing) – це вид тестування, який виконується без підготовки до тестування продукту, без визначення очікуваних результатів та проектування тестових сценаріїв.

Це неформальне, імпровізаційне тестування. Воно не вимагає жодної документації, планування, процесів, яких слід дотримуватися при виконання тестування.

Властивості якісних тестів

Декілька перевірок в одному тесті



4	As a non-existing user, I can't login into application	Medium	Login	1. Open application 2. Go to Login page	1. Enter non- existing email 2. Enter invalid password 3. Click 'Login' button	Not clear what we will get in result - validation on non-exsting email or on invalid password???
5	Verify invalid data entering	Medium	Login	1. Open application 2. Go to Login page	1. Enter email with length 51 symbols (50 is max) 2. Enter email without @ 3. Enter email without dot 4. Enter email with length 4 symbols (5 is min) 5. Enter email with special symbols)(*&^%$# 6. Enter password with length 51 symbol (50 is max) 7. Enter password with length 4 symbols (5 is min) 8. Enter password with special symbols)(*&^%$# 9. Click 'Login' button	Message 'Entered email is invalid. Please, try it again' or 'Entered password is invalid. Please, try it again'????

Тест-кейс чи тест-сценарій?

Тест, який тестує нічого

Id	Test Summary	Priority	Module	Precondition	Test Steps	Expected Result
1	Login page is available when user open application	High	Login		1. Open application ...	Login page is available
2	Check 'Get Started' link exists when user login in application	High	Login	Open application	1. Open login tab 2. Verify that 'Get Started' link exists	'Get Started' link exists and avaible
3	Check 'Get Started' link is clicked, page for registration is opened	High	Login	Open application	1. Open login tab 2. Verify that 'Get Started' link exists 3. Click on 'Get Started' link	Registration page is opened
4	Check 'Email address' edit-field exists and editable	High	Login	Open application	1. Open login tab 2. Verify that 'Email address' edit-field exists 3. Enter valid symbols	2. 'Email address' edit-field exists 3. Valid symbols can be entered into this field
5	Check 'Password' edit-field exists and editable	High	Login	Open application	1. Open login tab 2. Verify that 'Password' edit-field exists 3. Enter valid symbols	2. 'Password' edit-field exists 3. Valid symbols can be entered into this field
6	Check 'Login' button exist and clickable	High	Login	Open application	1. Open login tab 2. Verify that 'Login button' exists 3. Click on 'Login' button	2. 'Login' button exists 3. 'Login' button is clickable
7	Check 'Forgot your password' link exist	High	Login	Open application	1. Open login tab 2. Verify that 'Forgot your password' link exist	2. 'Forgot your password' link exists
8	Check if 'Forgot your password' link is clicked, page for password changing is opened	High	Login	Open application	1. Open login tab 2. Click on 'Forgot your password' link	2. Page for password changing is opened

Всі ці тести можна пройти, але користувач не зможе увійти в систему або створити новий обліковий запис.

Ми витрачаємо час на їх створення та виконання, але не можемо гарантувати роботу цього функціоналу

Незалежність чи обґрунтована пов'язаність

Пов'язані тест-кейси явно чи неявно (у межах набору) посилаються на інші (як правило, на попередній).

Незалежні тест-кейси не посилаються на жодні інші.

Переваги незалежних тест-кейсів:

- легко виконувати;
- можуть працювати навіть після збою програми на інших тест-кейсах;
- можна групувати будь-яким чином і виконувати у будь-якому порядку.

Переваги пов'язаних тест-кейсів:

- можуть імітувати роботу реальних користувачів;
- наступний у наборі тест-кейс використовує дані та стан застосунку, що підготовлені попереднім.

Тести, які будуть автоматизовані

Що важливо для автоматизації:

- тестові дані повинні бути вказані в тест-кейсі або додати запит, як їх створити або знайти;
- 1 тест – 1 основна перевірка;
- забезпечити простежуваність між ручними та автоматичними тестами:
  - відзначити, які тести автоматизовані в інструменті тест-менеджменту;
  - додати коментар з ідентифікатором тесту в скрипт автотесту;
- дізнатися, який підхід до автоматизації використовується, і надати зручні для них тести;
- подумати про варіанти тестування з мінімальними зусиллями для автоматизації, але з найкращим охопленням:
  - з боку ручного тестування позитивні, негативні та граничні тести навколо однієї і тієї ж функціональності займають однаковий час, з точки зору автоматизації позитивні тести можуть бути розширені негативними та граничними перевірками набагато швидше;
  - навпаки, автоматизувати всі можливі комбінації даних немає сенсу, необхідно використовувати попарне тестування, щоб знайти оптимальний набір комбінацій.

Особливості тест-кейсів для автоматизації

- їх немає;

- Гарний тест-кейс = якісний алгоритм;
- Автоматизація = це програмування;
- Програма виконує алгоритм.

Пишіть якісні тест-кейси!

Отже підіб'ємо підсумок

Гарний тест-кейс:

- має зрозумілу ідею - хоча може бути складний у виконанні
- послідовний у досягненні мети.
- не містить зайвих (безглуздих) дій.
- не дублює інші тест-кейси.
- робить виявлену помилку очевидною.
- нічого не передбачає, а забезпечує те, що необхідно (пре-і пост-умови)

Необхідно ставитися до тестів як до інструкції або рецепту.

Головне – це зрозумілість та завершеність.

Що ми сьогодні вивчили

План уроку

- Позитивні та Негативні тести
- Black box
  - Попарне тестування (Pairwise testing)
  - Тестування переходів станів (State transition testing)
  - Користувацькі сценарії (Use case testing)
- Experience based
  - Передбачення помилки (Error guessing)
  - Дослідницьке тестування (Exploratory testing)
  - Інтуїтивне тестування (Ad-hoc testing)
- Як писати гарні тест-кейси

Підсумки

Що одне, найголовніше, ви дізнались сьогодні?

Дякую за увагу! До нових зустрічей!

Інформаційний відеосервіс для розробників програмного забезпечення

TestProvider Pearson CyberBionic systematics Microsoft Partner Windows Azure EBA