End-to-End System Testing:

Introduction:

End-to-end (e2e) testing is a methodology used to test whether the flow of an application is performing as designed from start to finish. The major purpose of e2e testing is to identify system dependencies and data integrity between various system components, micro-services, and/or other systems. The entire application is tested in a real-world scenario such as communicating with the database, network, hardware and other applications.

Objective:

1. Ensure Industrial Internet Application (IIA) applications workflow function as design
2. Identify system dependencies between various micro-services and/or other external system
3. Identify design defects, which are different to functional defects, using a user journal approach.

Justification:

Normally, the majority of test cases are scripted per function of micro-services, but not from the user journal approach which is important for business focus. Although scripted testing provides a lot of value in traditional software development, it cannot cover "design defects" for the entire system. This section will evaluate the advantages and disadvantages of scripted testing and propose an additional testing methodology for consideration.

Characteristic of Scripted Testing:

- Specifies the test input, operations, & expected results
- Easy to automated for regression due to the comparison points are static for passed or failed the test
- Easy to control (change/update/retired) by versioning

Key Benefits of Scripts:

- Careful thinking about the design of each test cases
- Optimizing each test case for its most important attributes
- Reusability for regression
- Traceability for auditing
- Easy to calculate the metric of percentage of completed test cases
- Cost effective

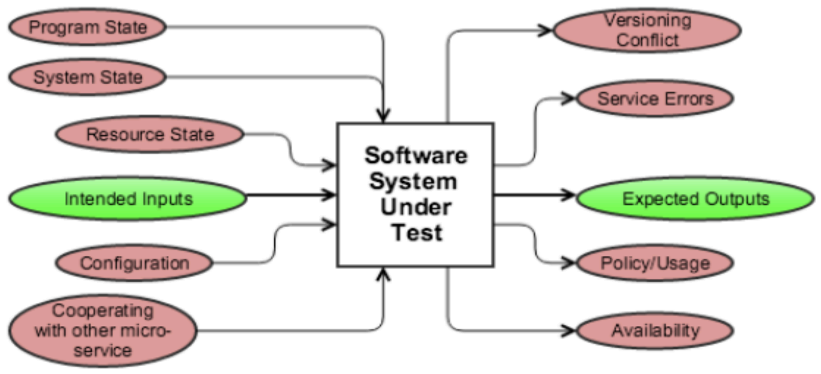
Challenge of Scripted Testing:

- May not cover all user scenario from end-to-end perspective
- Miss the same things every time (increase the cost)
- Risk profiles evolve over time
    - Data set is limited but not exhaustive (This is certainly for IIA. e.g. Aviation data vs. Health Care data)
    - System/Resource state change as microservices evolve
    - User expectations change over time
- Summary: Test stays the same, even though the risk profile is changing.

In summary:

Scripted testing is good for the following

- Fixed design
- Well understood risks
- Same set of errors appear on a statistically understood basis
- Test for the same things on each instance of the product



Design defect is the main reason for software failure in multi-machine system environments. Therefore, finding design defects is the most important task for a high quality software system. E2E System Test is the best fit testing methodology because it is an assessment of a design.

Recommendation:

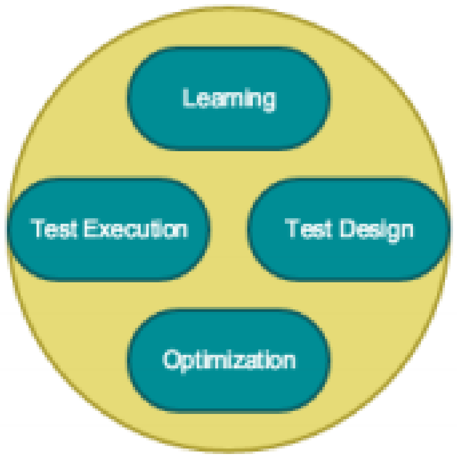
Implement design quality control by using exploratory testing techniques on IIA applications according to user scenarios from start to end.

Exploratory Testing

Rather than designing all tests in advance, we design and execute small, rapid experiments to uncover surprises, risks and potentially serious bugs that no one has done before.

Simultaneous...

- Learning
    - Requirements gathering: Identify users, objects, workflows, etc
    - Machine: "Does it do A?" vs. Human: "What else does it do?"
    - User and BA map requirements to unambiguous, active flowcharts.
    - Think inside the box!
- Test Design
    - Identifying interesting things to vary
    - Identifying interesting ways in which to vary things
- Test Execution
    - Integrated with CI/CD process (Automation vs. Manual)
    - Report process
- Optimization
    - If users make changes to the requirements, the tests can be automatically updated.
    - Think outside the box!



"In order to truly understand anything, you have to explore it." – Elisabeth Hendrickson

User Scenario Approach (Business Facing Testing)

- Intend to be used as an aid to communicating with non-programming members of a development team such as customers, business analysts, etc.
- Design to simulate a typical user's "scenario" through the system. Test will cover users' entire interaction with the system.
- Based on a user story about how the service is used - including information about the motivations of the person involved.
- The story is motivating. A stakeholder with influence would push to fix a program that failed this test.
- It can be automated, and can be integrated with CI/CD.

Risk of User Scenario Testing

- Scenario is complex. It involves a chain of actions with many features.
- If the first feature is broken, the rest of the test cannot be run. (Replication of failure is challenge)
- It is not designed for test coverage of the system. Only the selected workflow.

In Summary:

Since Industrial Internet Applications execute in a cloud environment which evolves rapidly, exploratory testing with a user scenario approach is the best fit testing methodology to discover the "design defects."

- Rapidly adopt to users' change requirement

- Rapidly adopt to evolved execution environment
- Reduce the learning curve on an evolving software system
- Focus on user expectation which define overall software quality

High-Level Requirements

- Working stable business systems with basic test data - users, assets, alarm, etc.
- Privilege that tester can start, stop, re-stage, deploy microservice
- Stakeholders agreement on design defect handling process

Implementation Timeline:

TBD

1. Setup operation foundation: Github, Jenkin, Rally, wiki 2.
2. Define initial test data (evolve): Alarm, Asset, TS data

Executive Milestones:

TBD

Budget Estimate: