You need a local emulator?

Yves Goeleven

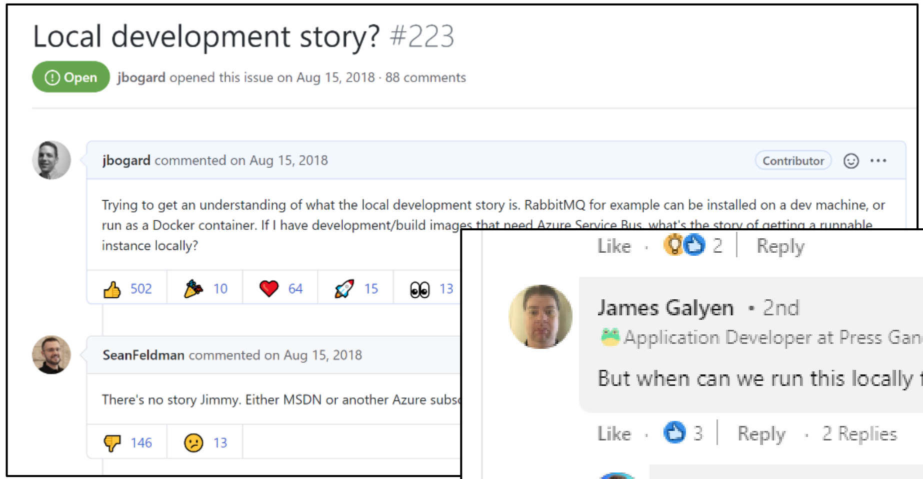
Independent Solution Architect

- My mission is to simplify distributed software development on Azure
- 22 years of experience, in Azure since 2008
- Co-founder AZUG, crew member Cloudbrew, first belgian Azure MVP
- Still building software (for) myself
  - www.clubmanagement.io
  - www.messagehandler.net

I need a local emulator!

I need a local emulator

Most common feedback for all Azure Services



Local development story? #223

Open jbogard opened this issue on Aug 15, 2018 · 88 comments

jbogard commented on Aug 15, 2018 Contributor

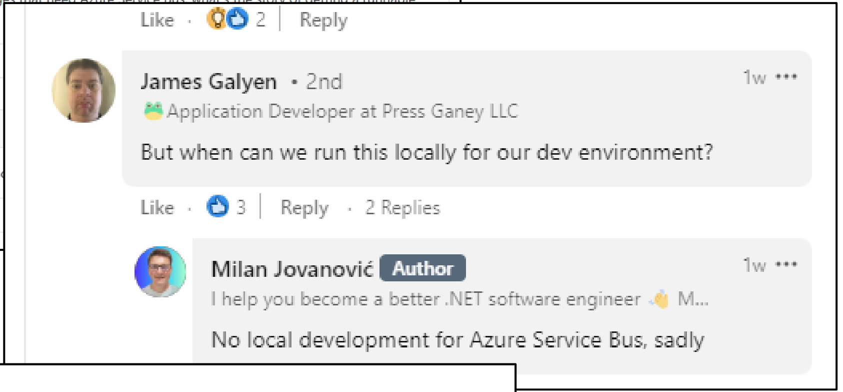
Trying to get an understanding of what the local development story is. RabbitMQ for example can be installed on a dev machine, or run as a Docker container. If I have development/build images that need Azure Service Bus, what's the story of getting a runnable instance locally?

👍 502 🎉 10 ❤️ 64 🚀 15 👀 13

SeanFeldman commented on Aug 15, 2018

There's no story Jimmy. Either MSDN or another Azure subs

👎 146 😕 13



Like · 2 | Reply

James Galyen • 2nd 1w

Application Developer at Press Ganey LLC

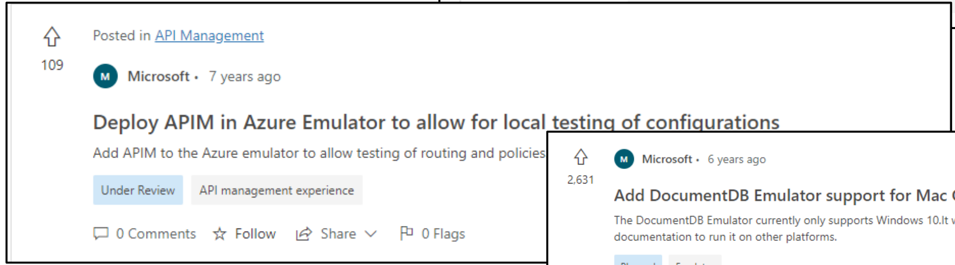
But when can we run this locally for our dev environment?

Like · 3 | Reply · 2 Replies

Milan Jovanović Author 1w

I help you become a better .NET software engineer 👋 M...

No local development for Azure Service Bus, sadly



109

Posted in API Management

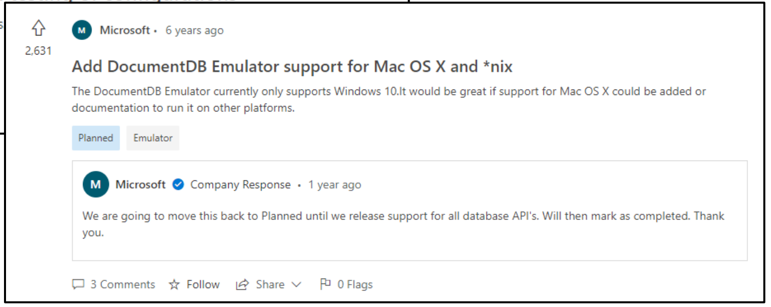
Microsoft • 7 years ago

Deploy APIM in Azure Emulator to allow for local testing of configurations

Add APIM to the Azure emulator to allow testing of routing and policies

Under Review | API management experience

0 Comments · Follow · Share · 0 Flags



2,631

Microsoft • 6 years ago

Add DocumentDB Emulator support for Mac OS X and *nix

The DocumentDB Emulator currently only supports Windows 10.It would be great if support for Mac OS X could be added or documentation to run it on other platforms.

Planned | Emulator

Microsoft Company Response • 1 year ago

We are going to move this back to Planned until we release support for all database API's. Will then mark as completed. Thank you.

3 Comments · Follow · Share · 0 Flags

Origin of the question

Good reasons people ask for this

- The ability to run an app offline
- To eliminate latency
- Improve test performance
- Environment per developer
- Test isolation
- Easier to debug state locally
- Avoiding development costs

You don’t need no stinking emulator!

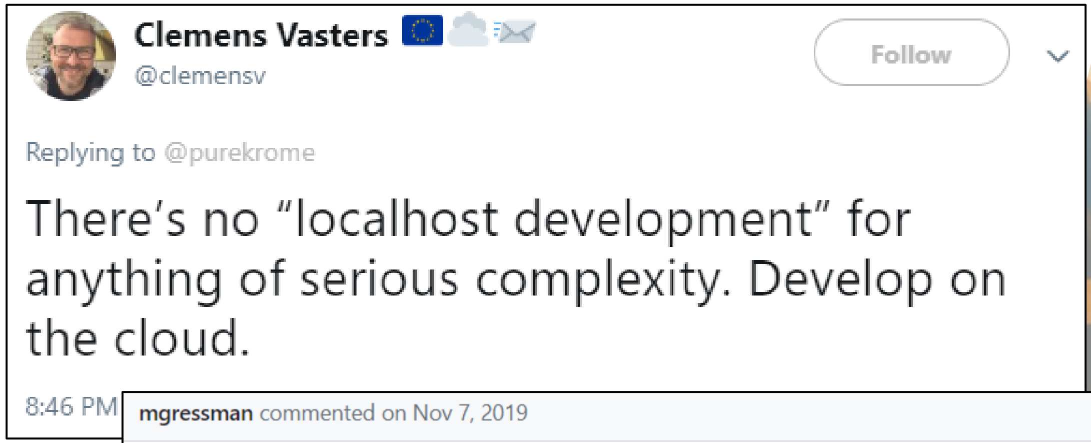
Why not?

Good reasons not to use emulators

- When your system fits on your laptop, do you really need a cloud?
- Emulators behave different from the real service
  - E.g. different response codes
  - Service limits are vastly different. (rps)
- Emulators are not hostile towards your code
  - E.g. throttling
- Latency matters: more likely to result in chatty code
- You need to know your operational costs asap
  - How you code makes a massive difference

The tension is rising

Old men arguing on the internet

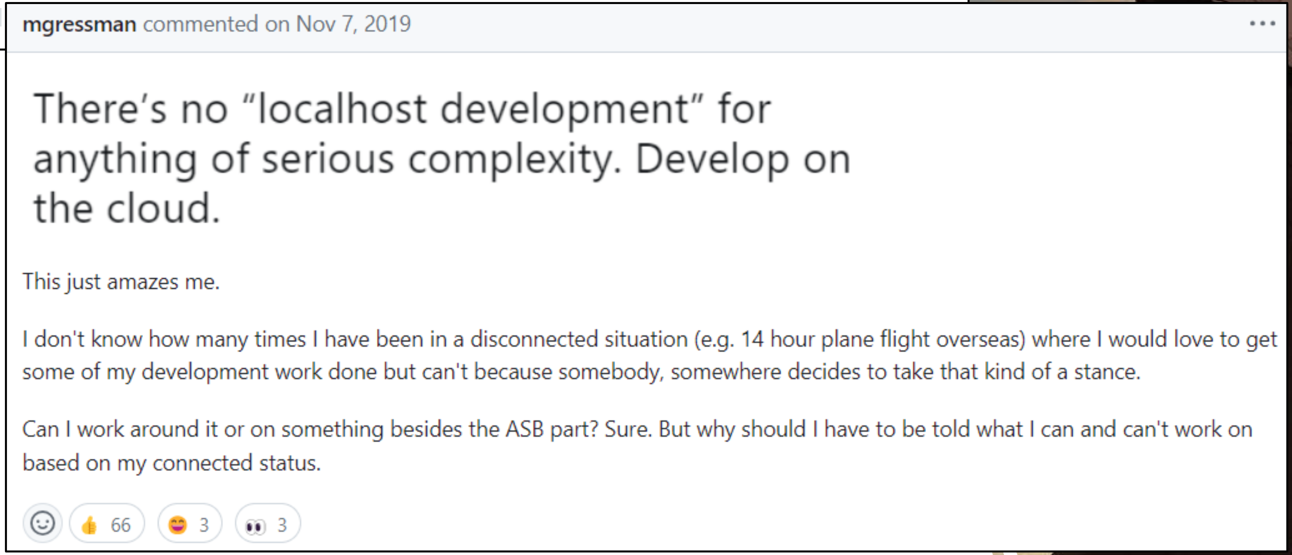


Clemens Vasters
@clemensv

Replying to @purekrome

There’s no “localhost development” for anything of serious complexity. Develop on the cloud.

8:46 PM



mgressman commented on Nov 7, 2019

> There’s no “localhost development” for anything of serious complexity. Develop on the cloud.

This just amazes me.

I don't know how many times I have been in a disconnected situation (e.g. 14 hour plane flight overseas) where I would love to get some of my development work done but can't because somebody, somewhere decides to take that kind of a stance.

Can I work around it or on something besides the ASB part? Sure. But why should I have to be told what I can and can't work on based on my connected status.

👍 66 😄 3 👀 3

How to solve this dilemma?

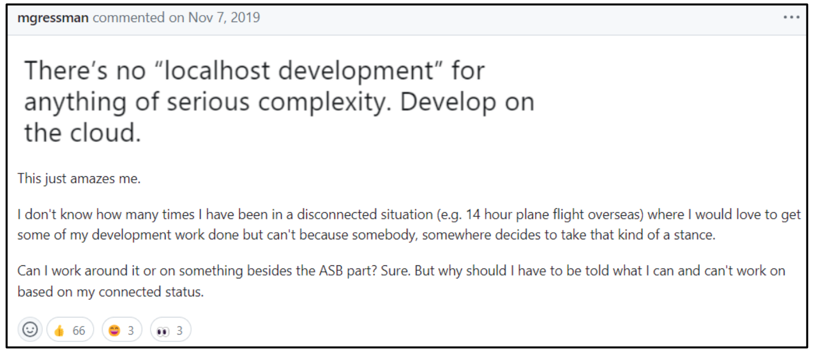
Root cause: Bad testing practices

The testing pyramid

How it is supposed to work

- 1 Manual test
- 10 Integration tests
- 100 Component tests
- 1000 Unit tests

- This person is testing manually



mgressman commented on Nov 7, 2019

> There's no "localhost development" for anything of serious complexity. Develop on the cloud.

This just amazes me.

I don't know how many times I have been in a disconnected situation (e.g. 14 hour plane flight overseas) where I would love to get some of my development work done but can't because somebody, somewhere decides to take that kind of a stance.

Can I work around it or on something besides the ASB part? Sure. But why should I have to be told what I can and can't work on based on my connected status.

👍 66 😄 3 👀 3

What most of us do

Our pyramids tend to be a bit top heavy

- 100 Manual tests
- 1000 Integration tests
- 100 Component tests
- 1000 Unit tests

Why ?

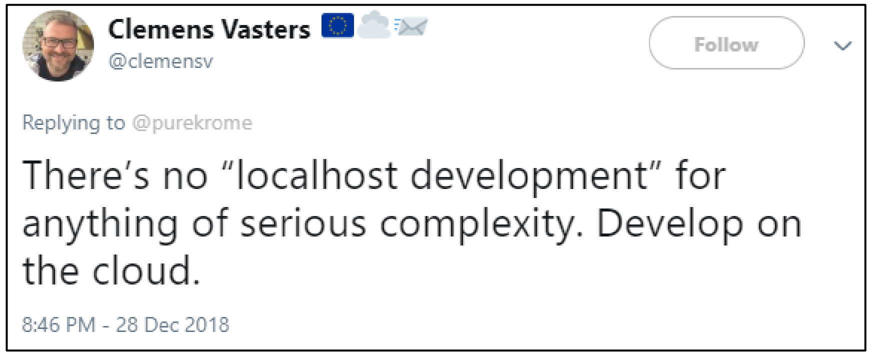
Multiple reasons

- Don’t trust unit tests
- Desire to visually confirm
- Fail to decompose our business domain
- Disconnect QA & Dev

Distrust in unit tests

Fake data not matching real data

- Unit tests use fake data
- Not in correspondence with reality
- Therefore need for more
  - Manual & Integration tests
- Using real data
- Thus, need for a live system
  - Works only for small systems



Clemens Vasters
@clemensv

Follow

Replying to @purekrome

There's no "localhost development" for anything of serious complexity. Develop on the cloud.

8:46 PM - 28 Dec 2018

Testing strategy

Add contract testing

Tests for your test data

- Tests for your test data
- Perform a few narrow integration tests
  - Against the real service
  - Serialize and store output in a verification file
- In a contract test
  - Assert the test data against verification
  - Use equality or equivalence assertions
- You can now trust your test data suite
  - Reuse in 1000s of unit and component tests
  - Without hitting the network

Manual assertions

Using string comparison

- Serialize actual test data to file
- Expectation in verification file
- Assert.Equal(expected, actual)

- Benefits
  - Absolute equality
  - Diff tools allow you to inspect the file content visually

- Downsides
  - Manual file management
  - Some properties may vary between runs, e.g. timestamps

Verification frameworks

Using Verify (By Simon Cropp)

- No file management needed
- Available for multiple dependency types

- Supports ‘Scrubbers’
  - Replaces values of certain types
  - Timestamps, guids, machine name, …

- Alternatives:
  - Use BeEquivalentTo comparison of Fluent Assertions on deserialized verification files
  - Pact.Net, biased towards API output only

DEMO: Integration testing

DEMO: Contract testing

New testing pyramid

How it can actually work

- 1 Manual test
- 10 Integration tests
- **100 Contract tests**
- 1000 Component tests
- 10000 Unit tests

10 Seconds

I challenged my team

Keep individual test runs below 10 seconds

- Additional practices
  - IO To the boundary
  - Proper functional decomposition

IO to the boundary

Only IO at specific points in call stack

- At an entry point
    - e.g. API controller
    - Load all data needed for the transition
- No IO in the middle
- At the exit point
    - 1 outbound IO operation
    - e.g. Save
    - Maximum 1!!!!!!!!
- Makes component testing a lot easier

Demo: IO to the boundary & Component tests

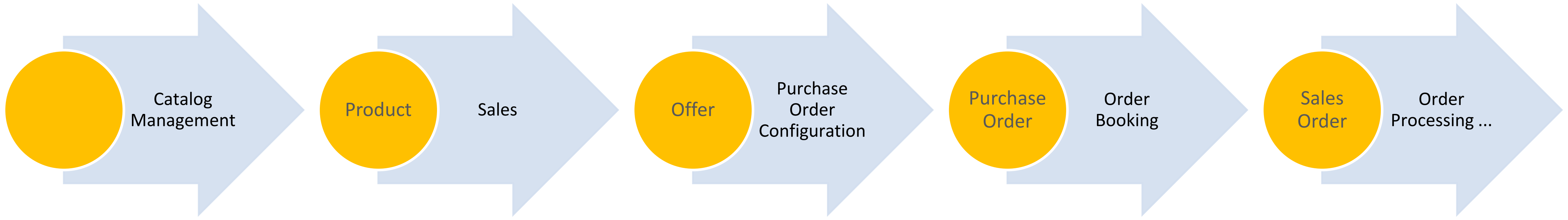
DEMO: Full Test Run & Unit Tests

Functional Decomposition

Failure to decompose business processes

Need for proper functional decomposition

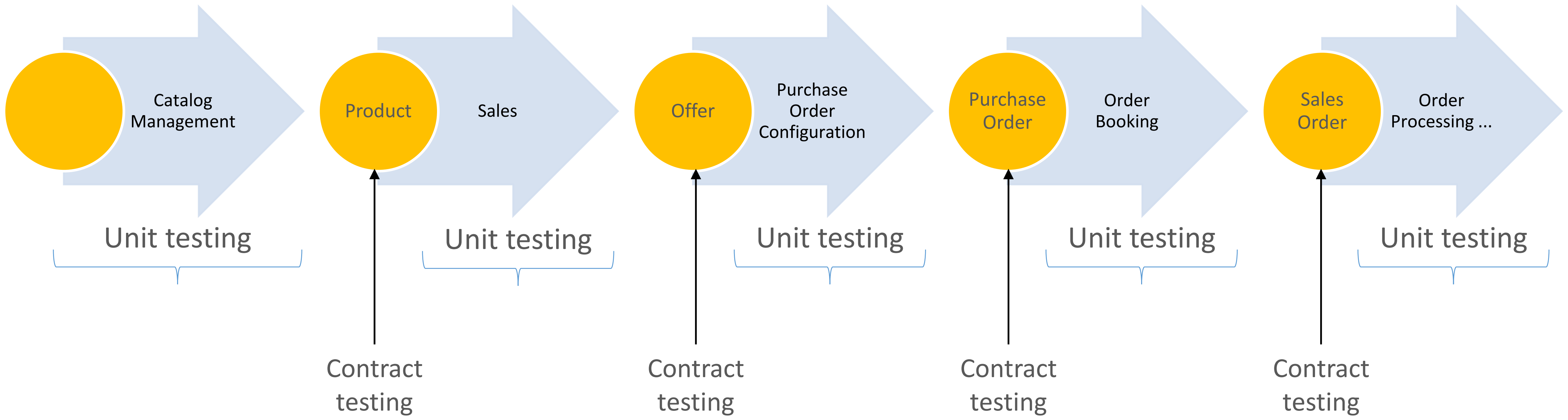
- All data is the result of process transitions (business capabilities)
- Tendency to see, and test, this process as a whole
- Resulting in manual tests or broad integration tests (E2E tests)



Replace slow end to end tests

With sequences of unit testing and contract testing

- The transitions can be tested using unit testing
- The exchanged data through contract testing



Share contract tests

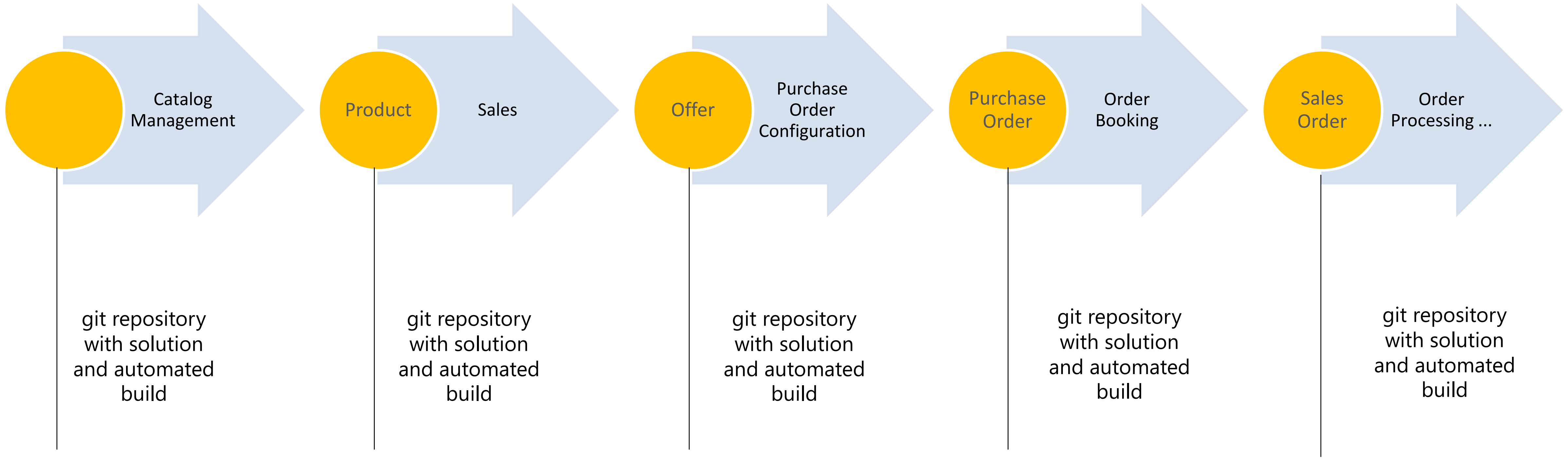
With the dependents of your API

- Write contract tests for your own API
- Embed tests & verification files in source package
- Share with dependents
- Run tests on both ends
- Both teams can now trust the test suite

Split your code base

To keep test runs short

- Separate git repository or solution per business capability
- With automated build and test run


Catalog Management
Product
Sales
Offer
Purchase Order Configuration
Purchase Order
Order Booking
Sales Order
Order Processing ...
git repository with solution and automated build
git repository with solution and automated build
git repository with solution and automated build
git repository with solution and automated build
git repository with solution and automated build

UI snapshot test

Just another contract test

- Instead of serializing in JSON
- Serialize in HTML (or XAML, or bitmap)

- Perform equality or equivalence comparisons

- Verify: Blazor, Images, Xamarin, Xaml, …
- Jest: HTML, CSS, JS, Images, …

QA Feedback loop

Organizational Trust

It does not matter you trust your tests

- When QA doesn’t know about, or does not trust, the unit test set
- They will still test manually
- Or use slow end to end integration tests

Set up feedback loop

Part 1

- During sprint planning talk about the end to end scenarios
- Map the end to end scenarios to Unit tests
  - Visual Studio / Azure Devops
- Let QA team review the unit tests for readability

Set up feedback loop

Part 2

- Replicate any bug as a failing unit test
  - Before fixing it
- Let QA report exploratory tests that succeeded
  - Add these as a unit test
- Eventually all scenarios will get covered
- Visualize on a dashboard
  - Report unit tests results
  - Aggregate per end to end scenario

Before you run

A Summary

- Adjust your testing practices
- Use real services, but sparingly
- Ensure an in-memory dataset that you and your organization can trust.
- Set up a feedback loop to improve your test suite over time

Thank you for your attention

Additional resources

- Simon Cropp's Verify Framework
  - https://github.com/VerifyTests/
- Dennis Doomen's Fluent Assertion Framework
  - https://github.com/fluentassertions/fluentassertions
- Pact Foundation, Pact.Net
  - https://github.com/pact-foundation/pact-net
- Jest Snapshot Testing
  - https://jestjs.io/docs/snapshot-testing
- Associate automated unit tests with test cases
  - https://learn.microsoft.com/en-us/azure/devops/test/associate-automated-test-with-test-case?view=azure-devops

LinkedIn/Goeleven