GUIDE

Kom igång med stubbning

Zington

Inledning

Krånglar dina testmiljöer? Ägnar du mycket tid åt testförberedelser snarare än testning? Får du inte upp alla dina system i alla olika miljöer? Då är det stubbning du behöver få till för att kunna testa effektivt! Här får du ta del av våra bästa tips när du ska börja stubba.

Den här guiden är en praktisk inledning till hur du kan få bort komplexiteten med externa system för att snabbare komma igång med testning och även som en möjliggörare för att komma igång med automatisering. Vi kollar på när man ska/kan stubba, vilka verktyg som finns samt fördelar och nackdelar.

1 När passar stubbning?

Stubbning är inte ett okänt koncept bland utvecklare. De senaste åren har det dessutom blivit mycket lättare att stubba. Initiativet till att börja stubba kommer ofta från DevOps-intresserade tekniska testare eller seniora utvecklare. För dem är fördelarna ofta väldigt tydliga och man ser stubbning som en självklar strategi för effektiv utveckling.

Om man sitter med ett gammalt legacy-system med en massa integrationer och systemsamband upplevs stubbning lätt som betydligt krångligare. I de fallen kommer man ofta i gång eftersom man upplever att man har testdatautmaningar, och generellt börjar de då med samband/integrationer till organisationsexterna system, eftersom dessa ofta anses som de krångligaste och mest tidsödande.

Många organisationer har publika API:er i samband med till exempel Open Banking eller myndigheternas direktiv för öppet data. Har man publika API:er vill man också tillhandahålla sandbox-miljöer till de organisationsexterna partners och kunder som använder dessa så att de kan bygga lösningar och testa lösningar mot dessa. I det sammanhanget är stubbar ovärderliga.

2

Verktyg för stubbning

Numera är det många verktyg som kommer med egna stubbar. Det går till exempel att stubba direkt i SwaggerUI - ett gratis verktyg för att grafiskt utforska och testa API:er.

Verktyg som Postman och ReadyAPI kommer med egna stubbmekanismer och alla de större verktygsleverantörerna har inbyggda stubbningslösningar i sina testautomatiseringsverktyg.

Om man sträcker sig efter enterprise-lösningar kommer flera integrationsplattformar och API Management-lösningar inkluderade med lösningar för stubbning, som dock i deras fall ofta kallas för mockar.

Som tidigare nämnts tillhandahåller större mjukvaruleverantörer. Till exempel CA, MicroFocus och IBM, verktygslösningar för Service Virtualization på enterprise-nivå. Sådana lösningar kan användas för större mockningsinitiativ.

3 Fördelar med stubbning

Möjlighet till branschad utveckling	Utan att kunna testa vårt system fristående blir det svårt att hålla flera utvecklings-branschar igång samtidigt och samtidigt göra tester över enhetstestnivån.
Enklare med negativa tester med “omöjligt” data	Genom att man kan preppa testdata själv blir det lättare att åstadkomma de konstiga och knasiga data-variationerna.
Icke-skakiga testmiljöer	Med stubbar slipper man interagera med buggiga kringliggande system och mycket miljökonfiguration, knasiga testdata-lägen i test och sönderskrivna databaser. Man får därigenom ett enklare och stabilare sätt att testa sin egen systemfunktionalitet.
GDPR-säkert	Med en begränsad mängd testdata är det lättare att själv konstruera data som är GDPR-säkert - och lättare att återanvända detta eftersom man inte behöver så mycket testdata eftersom man inte konsumerar upp det i testerna.
Slipper behörighetsstök	Utan att kunna testa vårt system fristående blir det svårt att hålla flera utvecklings-branschar igång samtidigt och samtidigt göra tester över enhetstestnivån.
Risk-minimerat för organisationen	Tester mot fullt integrerade system innebär stor risk för buggar. Dessa buggar kan riskera förstöra testdata i andra systems databaser och krångla till testläget för hela organisationen. Med buggar kan man testa tidigt på sin egen kammare, utan risk att krånga till det för andra. Sönderkört testdata kan ta lång tid att återladda.

Incitament till att versionshantera API-specifikationer och applikationskod separat	Samband och gränssnitt mellan system ändras mindre ofta än applikationskoden i sig. Det är värt att versionshantera dessa separat, men kräver förberedda repositoryn för detta. Stubbning kan vara incitamentet som får till detta.
Felsökningsbart	Felsökning av ett avintegrerat system är mycket enklare än felsökning i ett stort och integrerat system. Ju mindre kodbas som kan härbergera felet desto snabbare går det att identifiera och åtgärda detta – vilket dessutom som bonus minskar risken att man bygger vidare på ett buggigt system och får ännu mer komplexa felelimineringsåtgärder senare.
Agil-anpassat	Att implementera en affärsförändring i t.ex. en SAFe-organisation kräver ofta insatser från flera olika team, och kanske flera olika agila tåg. Dessa olika konstellationer jobbar mot olika backloggar och det är troligt att inte alla inblandade hinner klart i samma sprint. Att då behöva vänta på andra för att kunna testa vår applikationskod gör kvalitetssäkringen ineffektiv och bygger både risk och kanske teknisk skuld.
Slipper testa med dåligt produktionsdata	Produktionsdata kan tyckas vara idealiskt, men det är egentligen oftast väldigt dåligt för tester. Produktionsdata innehåller bara sådant som hänt historiskt och inte sådant som kan komma att hända framöver. Ofta är datamängderna så stora att det är tungt att rulla tillbaka till tidigare testdatalägen, tungt att söka fram relevant testdata för just mitt testfall.
Moln-anpassat	Stubbar kan snabbt startas och stoppas – eller hållas igång kontinuerligt. Oavsett är de påfallande enkla för snabb miljö-setup inför tester i samband med CI/CD-pipelines.

4 Begränsningar med stubbning

När man stubbar är det svårt med tester som kräver stora datamängder; migreringstester, batch-jobb-driven funktionalitet, prestandatester samt tester av t.ex. machine learning (ML) och andra typer av artificiell intelligens (AI), business intelligence (BI), data-warehouse (DW) och liknande.

En del typer av Software-as-a-Service (SaaS) och COTS saknar väldokumenterade API:er vilket försvårar för stubbning.

Vi kan heller aldrig bli helt säkra på att affärsflöden fungerar i sammankopplade system om vi inte testar dessa under så produktionslika former vi kan. Vi kan bara bli tryggare med att det kommer att fungera när vi väl kopplar ihop systemen - och det är mycket värt i sig. För system med stor komplexitet i datasamband kan man behöva spela in svaren till stubbarna för att det ska hänga ihop. Exempel på sådant data är inom bank- eller försäkringsbranschen.

För att effektiv stubbning ska fungera finns det vissa förutsättningar som måste vara uppfyllda redan:

- Kommunikationsarkitektur som bygger på API:er
- Mekanism för versionshantering av API-specifikationer och access till dessa
- Disciplin kring att säkerställa att API-specifikationer alltid är uppdaterade

Stubbing underlättas också av att man har en lös länkning mellan system så att man inte behöver kompilera om för att byta integrationsinställningar.

Visst går det att stubba Enterprise Java Beans (EJB), Remote Method Invocation (RMI), CopyBook, Kafka, DCOM, Remote Procedure Call (RPC), Stored Procedures och liknande också. Det kräver dock vanligen också att man utvecklar mer själv eller använder någon form av produkt för Service Virtualizaiton. Det vanligaste är att stubba REST- och SOAP-tjänster.

5 Checklista vid val av stubbhjälpmedel

- [ ] Litet smidigt verktyg för att kunna checkas in/användas jämte applikationskoden.
- [ ] Full kontroll av testdata från API-konsumentens sida (mandat, behörigheter, förståelse).
- [ ] Kunna startas och stoppas i CI/CD-pipeline i samband med tester.
- [ ] Kunna förses med svarsdata (förberedda responses) via ett eget API för mångsidighet vid testautomatisering.
- [ ] Kunna hantera att simulera alla relevanta transportmekanismer för data över företaget (MQ, HTTP(S), FTP, WCF o.s.v.).
- [ ] Kunna hantera att simulera alla relevanta transportmekanismer för data över företaget (MQ, HTTP(S), FTP, WCF o.s.v.).
- [ ] Stöd för att hantera använda säkerhetslager som inte kan abstraheras bort (klientcertifikat, basic auth, OAuth2 o.s.v.).
- [ ] Inspelningsförmåga för att kunna skapa testdatastubbar, eller generering från API-specifikationer (=Swagger-filer/OpenAPI, XSD, XSLT o.dyl.).
- [ ] Inbyggd validering av JSON-data och XML-data mot schema, och av tjänster mot XSLT-beskrivningar.

Flertal metoder för att identifiera vilket svar den skall skicka. T.ex. genom att kombinera en eller flera av följande:

- [ ] Skicka tillbaka senaste mottagna meddelande.
  - → Välj svar att skicka beroende på URL-endpoint.
  - → Välj svar att skicka beroende på URL-parametrar.
  - → Välj svar att skicka beroende på vilken server som ställer frågan (HTTP Origin header).
  - → Välj svar att skicka beroende på innehåll i HTTP Request (HTTP body, HTTP verb, Header-värden o.s.v.).
  - → Välj svar att skicka beroende på tid på dygnet.
- [ ] Integreringsmöjlighet med eventuell testautomatiseringslösning.
- [ ] Dynamisk run-time-substituering av data i svaren från stubben, för t.ex. timestamps och korrelerade sessionsvärden.
- [ ] Möjlighet att sätta en delay innan svar skickas.
- [ ] Möjlighet att själv designa svarsmeddelande (t.ex. HTTP svarskod, headers och body) på returnerade meddelanden.
- [ ] Möjlighet till slip-through - att om man inte har ett registrerat svar att skicka går frågan vidare till bakomliggande system.

Avslutning

Vi på Zington har med framgång hjälpt många företag och organisationer att implementera effektiv testautomatisering, där stubbning är en del av kostnadseffektiv och smart automatisering. Det finns massor av tips och trix för att få till en effektiv automatisering, och många fallgropar som man bör undvika. Där har vi idag fler än 60 experter som har kunskapen och erfarenheten att hjälpa dig.

Är du intresserad av en genomlysning av hur ditt företag jobbar med kvalitetssäkring eller få praktisk hjälp att komma igång, både med det stora och det lilla. Då finns vi på Zington här för att hjälpa till med moderna metoder och tekniskt ”know how” inom mjukvaruutveckling med kvalitet.

1. **När passar stubbing?**
2. **Verktyg för stubbing**
3. **Fördelar med stubbning**
4. **Begränsningar med stubbning**
5. **Checklista vid val av stubbhjälpmedel**

→ **Läs mer om vårt erbjudande om Test & Kvalitet på zingtongroup.com**
Kontakt: anders.eng@zingtongroup.com