Felfri kod finns inte

EXPERTARTIKEL

IoT-utvecklare behöver få återkoppling från fältet

Av Johan Kraft, Percepio

Dr. **Johan Kraft** är VD för Percepio, som han grundade 2009. Han har doktorerat i datavetenskap och själv utvecklat den ursprungliga versionen av Tracealyzer, Percepios verktyg för visuell mjukvarudiagnostik. Hans bakgrund är inom tillämpad forskning på inbyggd programvara, utförd i nära samarbete med lokal industri. Innan doktorandstudierna arbetade han med utveckling av inbyggd programvara på ABB Robotics.

Ärligt talat kan vi aldrig påstå att någon programvara är helt felfri. Statistiken säger att utvecklare producerar ungefär 120 defekter per 1 000 rader programkod, och att omkring 5 procent –sex defekter per 1 000 kodrader – inte upptäcks under verifieringen och följaktligen finns kvar i den färdiga produkten som levereras till kunderna.

Det exakta antalet defekter må variera från projekt till projekt, men det underliggande problemet är välkänt sedan datoriseringens begynnelse. Och varför är det fortfarande ett problem, trots 50 år av förbättrade verktyg, programspråk och processer? Det finns flera anledningar men en av de viktigaste är att de system vi utvecklar blir allt mer komplexa, och att graden av komplexitet växer snabbare än vår förmåga att verifiera komplexa system. Därför blir det allt svårare att garantera kvaliteten på programvara.

I de flesta fall är det i praktiken omöjligt att testa alla möjliga användningsfall och vägar genom programkoden – det finns alldeles för många möjligheter. Det går alltid att spendera mer tid och pengar på verifiering men i slutänden kan du ändå inte veta att du har hittat alla defekter. Och de flesta utvecklingsprojekt har både en deadline och en begränsad budget.

Nåväl, din produkt är nu klar och du har testat koden efter bästa förmåga och i enlighet med de rutiner och processer som organisationen använder. Då börjar det verkliga testet: tusentals användare som börjar använda produkten, många gånger på sätt som utvecklingsteamet aldrig har tänkt på och ännu mindre testat. Ju fler användare och ju större programkod, desto mer sannolikt att kvarvarande defekter faktiskt orsakar problem.

I bästa fall kommer dessa användare kontakta dig, och förse dig med tillräcklig information för att du ska kunna reproducera och fixa felet. Men det troligaste är att de bara startar om enheten och fortsätter använda den, i synnerhet om det handlar om en konsumentprodukt. Om felet är allvarligt, eller om det upprepas gång på gång, är det risk att de skriver negativa recensioner och undviker dina produkter framöver.

Marknadsundersökningsföretaget VDC Research frågade häromåret ut ett antal företag som lanserat större inbyggda system, och enligt deras enkät behövde ett genomsnittligt system inte mindre än 79 uppdateringar under det första året i drift. Många av dem var sannolikt defekter som behövde rättas till. VDC konstaterar också att varje rättelse kostade i genomsnitt över 50 000 kronor (5 000 USD); totalt handlar det alltså om underhållskostnader på åtskilliga miljoner per projekt bara under det första året.

Övervakning för driftsatta IoT-enheter

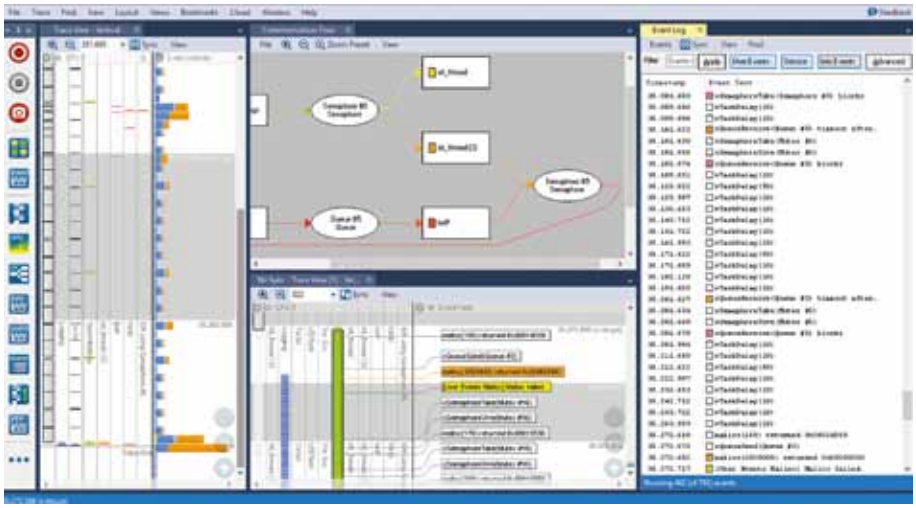
IoT-enheter, som ju per definition är uppkopplade, gör det möjligt för oss att bedriva utveckling av inbyggda system på ett nytt sätt. Den traditionella cykeln koda – testa – felsöka är fortfarande central, men eftersom vi vet att åtminstone några defekter kommer att finnas kvar i den färdiga produkten kan vi redan från början planera för att använda uppkopplingen för felrapportering.

Så snart ett fel upptäcks i enhetens programvara kan nätverksuppkopplingen användas till att automatiskt meddela utvecklarna, och tillsammans med meddelandet kan man skicka med diagnostisk information som underlättar felsökningen. Det kan förstås vara en felkod, men det går också att bifoga mycket mer, till exempel spårningsinformation och loggfiler som visar vad som hände i systemet just innan felet uppträdde. Du kan sedan analysera den informationen i ett lämpligt verktyg för visuell mjukvarudiagnostik som Percepio Tracealyzer.

Och när felet är diagnosticerat och rättat kan uppdaterad programvara skickas ut till de kunder som berörs som en over-the-air-uppdatering, vilket är både snabbare och säkrare än att uppmana kunderna att själva uppdatera sina enheter.

På det här sättet bygger man en återkoppling mellan utvecklarna och den kod som körs ute hos kunderna, och möjliggör ett mer agilt, DevOps-inspirerat arbetssätt med snabba uppdateringar. De som utvecklar mobil- och molnapplikationer har länge arbetat på det sättet och nått goda resultat, och tack vare säkra molntjänster som AWS IoT Core och Microsoft Azure IoT kan även IoT-utvecklare göra det nu.

Den affärsmässiga fördelen med omedelbara felrapporter och detaljerad diagnostik är att defekter i programvaran kan lösas betydligt tidigare och till lägre kostnad. Det



Det är värt att notera att insamling av diagnostiska data typiskt inte faller under dataskyddslagstiftning som GDPR. Det är först när informationen kan knytas till en enskild person, via e-postadress eller IP-adress eller liknande, som lagarna börjar gälla. DevAlert har utformats med tanke på detta, och den information som skickas till DevAlerts molntjänst är helt anonymiserad.

innebär i sin tur att färre användare kommer påverkas av varje fel, vilket direkt kan översättas till fler nöjda kunder. Varför blir det så? Jo, därför att fel som missats under testfasen typiskt bara uppträder under vissa speciella omständigheter – det var därför de inte hittades under testningen – och därför kommer de sannolikt bara drabba enstaka kunder till att börja med. En reparationstid som mäts i dagar eller till och med timmar, istället för som tidigare veckor och månader, blir då en konkurrensfördel.

Notera dock att återkopplingen kan resultera i mer jobb med felrättelser initialt efter lansering, eftersom fel som annars hade förblivit okända nu rapporteras in omgående. På längre sikt bör dock underhållsarbetet minska betydligt och inte minst kommer slutanvändarna vara betydligt mer nöjda med produkten.

Inte bara programfel

Det arbetssätt som presenterats ovan kan användas för att rapportera alla slags problem som kan upptäckas i programvara. Det kan vara programfel (felkontroller, krascher osv), problem med användbarheten (en konfigurationsguide som avbryts eller liknande), eller hårdvarufel som till exempel en givare som rapporterar in mätvärden utanför det tillåtna intervallet. Vidare är det också möjligt att implementera tidsbaserad rapportering, som till exempel att dagligen samla in statistik om hur enheten används och mäta parametrar som batteristatus, minnesförbrukning och liknande.

När det gäller hårdvaruproblem kan den omedelbara rapporteringen leda till en korrigering i produktionen så att antalet levererade felaktiga enheter minimeras. Diagnostiken kan avslöja om felet eventuellt ligger i en extern komponent, och i så fall utgöra ett bra underlag för diskussioner med den som levererat komponenten.

Vad som behövs

En förutsättning för att implementera det föreslagna arbetssättet är att kommunikationen mellan utvecklare och enhet(er) är säker. Det är erkänt svårt att få säkerhet rätt, men lyckligtvis erbjuder molnplattformarna för IoT – AWS IoT Core, Azure IoT, med flera – säker tvåvägskommunikation med hjälp av etablerade standardverktyg som TLS och X.509-certifikat. Genom att utnyttja dessa funktioner kan du implementera en återkoppling utan att samtidigt riskera att introducera nya säkerhetshål.

Fel kan detekteras exempelvis via explicita felkontroller eller undantagshanterare. Med största sannolikhet finns det redan feldetekterande kod på dessa platser, för att underlätta avlusningen, och den koden kan lätt utvidgas till att rapportera fel under drift, via molntjänsten, med hjälp av ett lämpligt kodbibliotek.

Den diagnostiska information som skickas tillsammans med felrapporten kan vara av endera av två slag:

- Direkta symptom, som felkoder och -meddelanden. Det kan också vara annan tillståndsinformation, som till exempel globala variabler eller olika räknare, allt efter vad ni som utvecklare anser att ni behöver.
- Spårningsinformation och loggar som visar händelser i systemet omedelbart innan felet uppträdde. Det ger ett sammanhang som underlättar förståelsen och även gör det lättare att reproducera felet vid behov. Förutsättningen för att den här informationen ska finnas är dock att relevanta händelser kontinuerligt spelas in medan systemet är i drift.

SPÅRNINGEN BÖR SES som en permanent del av systemet och vara aktiverad åtminstone från testfasen och framåt. Den innebär en viss overhead, typiskt några få procent CPU-tid och ett par Kbyte RAM och ROM, men i gengäld är felrapporteringen alltid tillgänglig.

Det som också behövs är en tjänst som tar emot, sparar och organiserar felrapporterna, samt hanterar att meddela utvecklarna. Tjänsten bör vara smart nog att kunna skilja på genuint nya felrapporter och sådana som huvudsakligen är dubbletter på tidigare fel; endast nya fel ska flaggas för utvecklarna, annars riskerar ni att översvämmas av meddelanden om många enheter skulle råka ut för samma fel.

Denna tjänst måste vara pålitlig, säker, och möjlig att skala upp till att hantera stora mängder driftsatta enheter, så även om den går att köra på en lokal server är det mycket som talar för en molnplattform i praktiken fungerar bättre.

Percepio DevAlert

Den tjänst som beskrivits ovan har nyligen lanserats kommersiellt som Percepio DevAlert. DevAlert är en molntjänst för insamling och hantering av felrapporter (”alerts”) från DevAlert Target Agent, ett kodbibliotek som du som utvecklare integrerar i enhetens programvara.

I DevAlert ingår också Percepio Tracealyzer, ett avancerat verktyg för visuell mjukvarudiagnostik för realtidssystem som har funnits på marknaden i snart tio år. Tracealyzer översätter spårningsinformationen till en grafisk tidslinje som visar alla programvaruhändelser i det spårade systemet – man kan likna det vid en övervakningsfilm över systemets inre tillstånd där du kan zooma in på ”misstänkta” händelser för att se vad som hände i detalj.

När Tracealyzer används inom DevAlert visas de inspelade händelser som ledde fram till det rapporterade felet, vilket gör det mycket enklare att hitta och rätta den underliggande defekten.

Alla inkommande felrapporter i DevAlert sparas i molntjänsten tillsammans med bifogade symptom och metadata som datum, version av systemet och så vidare. Informationen kan senare plockas fram för att generera olika statistikrapporter. Molntjänstens andra huvuduppgift är klassificering, att gruppera rapporter med samma symptom som ett och samma problem; endast den första förekomsten av ett problem ska normalt signaleras till utvecklarna. Utan den funktionen skulle det vara svårt att skala upp DevAlert till att hantera stora grupper av enheter.

En fungerande återkoppling

Mötet mellan inbyggd programvara och internet har skapat en marknad för Sakernas internet (IoT) som lockar med enorma möjligheter för hela industrin. IoT-plattformar som Microsoft Azure IoT och AWS IoT Core underlättar driftsättning och hantering av enheterna samtidigt som de erbjuder säkerhet och skalbarhet.

Förmågan att övervaka driftsatta applikationer är fundamental för den DevOps-filosofi som har blivit så framgångsrik för utveckling av mobila och molnbaserade applikationer. Tyvärr har det saknats lättanvända lösningar för att tillämpa samma filosofi vid utveckling av inbyggda applikationer. ■

Percepio DevAlert har tagits fram för fylla det behovet, erbjuda återkoppling och tillåta IoT-utvecklare att dra nytta av DevOps-tänkandet. DevAlert är nu tillgänglig för AWS IoT Core och enheter som kör FreeRTOS eller ThreadX – välkommen att kontakta Percepio för att få hjälp med att komma igång.