Hardware-in-the-Loop

Termeni înrudiți:

Software încorporat, Verificare software, Testare software, Banc de testare, Sisteme încorporate, Inginerie software

Vezi toate subiectele

Dezvoltarea și verificarea modelului

George Ellis, înGhid de proiectare a sistemului de control (ediția a patra), 2012

13.2.6 Hardware în buclă

Hardware in the loop (HIL) sau Hardware-in-the-loop-simulation-testing (HILST) este, într-un fel, opusul RCP: legile de control sunt implementate hardware-ului final în timp ce convertorul de putere, instalația și feedback-ul senzorii sunt simulați. Ca și în cazul RCP, simularea trebuie să fie executată în timp real, deoarece o parte a sistemului (aici, controlerul) este hardware fizic. De asemenea, trebuie să existe hardware pentru interfața cu ieșirea legii de control și semnalele senzorului de feedback. Această configurație este prezentată în Figura 13.12.

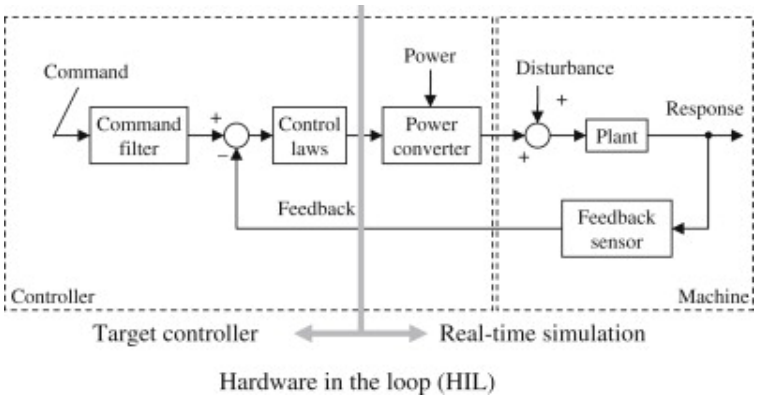

Figura 13.12. Hardware în buclă (HIL).

HIL a fost folosit de ani de zile în industrii precum aerospațial sau auto, unde instalațiile sunt deosebit de complexe. HIL poate spori testarea plantelor fizice pentru a oferi multe avantaje, cum ar fi:

• Costuri reduse de testare Costul ridicat de executare a testelor pe mașini complexe, cum ar fi subsistemele de aeronave, poate justifica investiții substanțiale în HIL. Testarea pe teren poate fi redusă prin utilizarea HIL ca o completare. Pe site-ul lor, National Instruments prezintă un exemplu în care Process Automation a redus timpul de testare pe teren la jumătate utilizând HIL, economisind 150.000 USD în costuri asociate testării.

un sistem de control al dispozitivului de oprire a aeronavei (www.ni.com/HIL, faceți clic pe „Ce este testarea HIL?").

• Reduceți riscurile asociate cu defecțiunile Cu instalațiile complexe, funcționarea defectuoasă a sistemului de control poate duce la defecțiuni catastrofale, distrugerea echipamentelor sau prezentând pericole pentru siguranță. HIL poate fi folosit pentru a valida controlerele înainte de a rula echipamentele fizice. Acesta poate fi folosit la începutul proiectului pentru a valida un nou controler; poate fi, de asemenea, utilizat pe parcursul dezvoltării pentru a reduce șansele ca modificările software ale controlerului să introducă noi moduri de defecțiune (deseori denumite *testarea regresiei*). Ambele utilizări reduc probabilitatea de a întâlni moduri de eșec neașteptate pe hardware-ul fizic.

• Dezvoltarea concomitentă a componentelor sistemului de control În multe proiecte de dezvoltare a sistemului de control, controlerul poate fi disponibil cu mult înainte ca convertorul de putere, instalația și senzorii de feedback să fie disponibili. Folosind HIL, testarea controlerului de producție poate începe înainte ca celelalte componente să fie gata.
Acest lucru poate reduce timpul general de dezvoltare, sporind în același timp încrederea în controler.

• Testarea multor variații de plantePlantele variază adesea, iar testarea mostrelor din fiecare variație poate fi costisitoare. De exemplu, turbinele morilor de vânt sunt disponibile în mai multe mărimi de putere. Dacă fiecare ciclu de testare trebuie efectuat pentru fiecare variație, este posibil să fie necesare atât de multe versiuni ale instalației pentru testare, încât costurile sunt prohibitive. Folosind HIL, testarea dintr-un subset al variațiilor disponibile poate fi valorificată. De exemplu, execuția pe teren a unor teste ar putea fi efectuată pe câteva variații (de exemplu, cel mai mare, cel mai mic și alte câteva modele). Apoi testarea HIL ar putea fi utilizată pentru unele funcții ale modelelor intermediare.

• Testarea modurilor de eroareHIL permite testarea mai robustă a modurilor de eroare. Adesea, testarea amănunțită a modurilor de eroare este nepractică atunci când se bazează în întregime pe sisteme fizice. Cu HIL, defecțiunile pot fi induse prin software și pot fi sincronizate cu o gamă largă de condiții. De exemplu, o defecțiune a frânei ar putea fi simulată atunci când modelul HIL al vehiculului funcționează într-un număr mare de combinații de viteză, greutate a vehiculului și diferite condiții ambientale. Pentru defecțiuni care apar din defecțiuni mecanice, inducerea defecțiunilor într-o gamă atât de largă de condiții la o frână fizică ar putea fi nepractică.

Un exemplu de HIL folosit pentru accelerarea dezvoltării și îmbunătățirea siguranței poate fi luat din EcoCAR Challenge (http://www.ecocarchallenge.org/). În provocarea EcoCAR, echipele din universități concurează pentru a produce cele mai eficiente prototipuri de automobile. Un sondaj recent a arătat că numeroase echipe se bazează pe HIL. De exemplu,

Virginia Tech și Ohio State University, două dintre cele mai performante echipe în provocarea EcoCAR, s-au bazat foarte mult pe HIL pentru succesul în competiție. Ambele echipe au condus controlere de vehicule și de motor conectate la echipamente de simulare HIL de la diferiți furnizori - Virginia Tech folosind produse National Instruments și Ohio State University folosind produse dSPACE. Ambele au raportat rezultate similare: programele sunt comprimate deoarece testarea poate începe înainte ca un tren motoare să fie instalat în vehicul.

> Citiți capitolul complet

Mediu ușor de utilizat pentru cercetarea și dezvoltarea controlerelor în mașini grele

T. Virvalo, ... M. Kivikoski, înMecatronică prietenoasă cu oamenii, 2001

5 CONCLUZII ȘI CERCETĂRI VIITOARE

Rezultatele arată că HIL este o abordare bună pentru testarea sistemelor de control distribuit. Este ușor să generați diferite condiții de încărcare în modelul de simulare și, prin urmare, hardware-ul controlerului poate fi testat în toate situațiile necesare, dintre care unele pot fi chiar periculoase în sistemul real. Se poate aștepta ca abordarea HIL să accelereze remarcabil testarea sistemelor de control.

Cercetările vor continua. Modelul de simulare ar trebui îmbunătățit și fiecare controler necesită, de asemenea, multă cercetare și dezvoltare. Fiabilitatea și performanța sistemului de control distribuit, inclusiv, de exemplu, analiza întârzierilor, sincronizarea și deviațiile ceasului controlerelor în sistemul CAN bus necesită, de asemenea, multă cercetare și dezvoltare.

> Citiți capitolul complet

Testarea vehiculelor inteligente folosind medii virtuale și scenarii în etape

Arda Kurt, ... Ümit Özgüner, înProgrese în vehiculele inteligente, 2014

2.1 Introducere

Testarea suficientă este o parte esențială a cercetării și dezvoltării în inginerie. Pentru obiectivul nostru principal, sistemele inteligente de transport (ITS), faza de testare include adesea teste preliminare în numeroase medii de simulare, precum și teste fizice care implică unul sau mai multe vehicule.

Încorporarea echipamentelor virtuale în testarea și evaluarea sistemelor de inginerie controlate este acum obișnuită, iar disponibilitatea unor echipamente puternice de calcul și achiziție de date și control a făcut ca hardware-ul în buclă să fie o practică standard. Aceste abordări sunt, de asemenea, utilizate în proiectarea și testarea tehnologiilor inteligente și automate ale vehiculelor. Există o serie de factori care justifică această abordare, printre care:

• Costul sau disponibilitatea echipamentului

• Timpul și efortul necesar pentru gestionarea și personalizarea activităților de testare pe teren

• Considerații de risc și siguranță, în special în cazul tehnologiilor nedovedite.

Din câte cunoștințele noastre, cele mai timpurii utilizări ale hardware-ului virtual în domeniul sistemelor de transport inteligente apar în lucrarea lui Robert Fenton, începând cu sfârșitul anilor 1960. Fenton a fost unul dintre cei mai timpurii cercetători activi în domeniul vehiculelor automate și al interacțiunilor șofer-vehicul.

În Ref. [1], el descrie o strategie de control longitudinal pentru urmărirea mașinii, ceea ce am numi plutoning într-un sistem automat de autostradă, implementarea acestui controler pe un vehicul experimental drive-by-wire și experimente efectuate la viteze de autostradă. Deoarece nu exista o metodă practică sau un senzor disponibil pentru a măsura distanța de separare (progres) față de vehiculul de conducere și viteza sau viteza relativă a vehiculului de conducere, el a creat o „mașină fantomă” simulată, al cărei comportament putea fi variat în funcție de dorința dorită. test, pentru a produce citirile necesare ale senzorului pentru experimentele sale de control.

În Ref. [2], el descrie o tehnologie timpurie de asistență a șoferului pentru urmărirea vehiculelor. El a proiectat o interfață haptică pentru a oferi informații despre avansul mașinii conducătoare și viteza relativă șoferului uman și demonstrează experimental îmbunătățirea rezultată a performanței de urmărire a mașinii față de un șofer neasistat, în special în situații cu drum scurt. Pe lângă problemele practice ale senzorilor cu care se confruntă Ref. [1], aceste experimente au implicat subiecți umani neafiliați cu programul de cercetare, astfel încât utilizarea unui vehicul de plumb simulat în timpul încercărilor inițiale și în timp ce antrenarea subiecților de testare a fost un factor important de siguranță și un ajutor de instruire.

O limitare majoră a testelor care implică scenarii cu mai multe vehicule, așa cum am observat de-a lungul experienței noastre mai recente în acest domeniu particular [3–5], este că testarea în aer liber poate avea un cost ridicat în ceea ce privește logistica și programarea, așa cum s-a menționat mai sus. Deoarece zonele de testare echipate corespunzător nu sunt întotdeauna ușor accesibile,

programarea pentru transportul mai multor părți implicate și pentru condiții meteorologice favorabile poate fi o problemă în sine.

Pentru a atenua unele dintre problemele de logistică, cost și siguranță enumerate mai sus, acest capitol descrie un supliment flexibil și cu costuri reduse pentru testarea în aer liber pentru cercetare și aplicații inteligente în domeniul transportului. Începând cu simulări pe computer complet virtuale și crescând treptat componentele din viața reală ale scenariilor testate, sistemele ciber-fizice complexe pot fi studiate fără multe dintre costurile asociate testării reale în aer liber.

Pentru traficul urban realist, în care vehiculele autonome interacționează cu vehiculele conduse de oameni, complexitățile menționate mai sus ale testelor în aer liber pot fi și mai descurajante; prin urmare, un banc de testare interior care emulează aspectele concentrate ale unui mediu exterior este adesea eficient pentru testele care implică mai multe vehicule, luarea de decizii la nivel superior și evaluarea situației. Vitezele în general mai scăzute ale scenariilor de trafic urban, spre deosebire de sistemele automate de autostrăzi, face ca testarea în interior să fie o opțiune deosebit de atractivă pentru teste consistente și repetabile.

Patul de testare interior de la Laboratorul de Cercetare pentru Controlul și Transportul Inteligent al Universității de Stat din Ohio [6], numit SimVille și văzut în Figura 2.1, a servit ca mediu de testare semi-virtual intermediar între simulările computerizate și testele în aer liber complet dezvoltate pentru o serie de proiecte de-a lungul anilor.

Figura 2.1. SimVille Indoor Testbed la OSU Control and Intelligent Transportation Research Lab, prin diverse încarnări și cu diferiți roboți mobili.

În următoarele subsecțiuni, vor fi descrise conceptele din spatele iterațiilor semi-virtualizate ale procedurii de testare, inclusiv simulări, bancuri de testare la scară mică și testare în aer liber la scară completă, iar exemplele vor fi folosite ca ilustrații ale acestora.

> Citiți capitolul complet

Verificarea proiectării funcționale și nefuncționale pentru sisteme software încorporate

Arnab Ray, ... Chris Martin, înProgrese în calculatoare, 2011

2.2 Verificarea funcțională

A*cerință funcțională*este o declarație a ceea ce trebuie sau nu trebuie să facă sistemul, de obicei exprimată sub forma: dacă o anumită condiție este valabilă, atunci sistemul ar trebui să răspundă în mod corespunzător.*Verificarea functionala*consta in verificarea daca software-ul satisface cerintele functionale.

În proiectele care urmează MBD, codul este generat automat de la modele și apoi flashat pe hardware. Testele sunt apoi executate pe combinația hardware-software (cunoscută ca testare hardware-in-the-loop) și dacă testele trec, se consideră că sistemul a fost verificat cu succes.

Testarea hardware-in-the-loop este un proces costisitor și consumator de timp nu numai pentru că paturile și mediile fizice de testare sunt dificil de întreținut, ci și pentru că erorile descoperite la generarea codului poștal sunt mai greu de rectificat (în termeni de timp și cost) decât dacă ar fi fost prinși chiar în faza de modelare. Principala critică la adresa practicilor existente în verificarea funcțională a modelelor este că proprietatea executabilă a notațiilor de modelare nu este valorificată corespunzător. Deoarece modelele pot fi rulate ca cod, o parte semnificativă a verificării funcționale poate fi împinsă în amonte în ciclul de dezvoltare.

Au fost dezvoltate o varietate de tehnici pentru a oferi metode automate și semiautomatizate pentru verificarea modelului care profită de semantica de execuție a notațiilor de modelare. Acestea includ (1) simulare ghidată [31] în care scenariile deduse din cazurile de utilizare sunt „rulate” pe model și rezultatele produse sunt studiate pentru a vedea dacă sunt ceea ce era de așteptat (2) verificarea modelului, o tehnică automată care a dat o modelul și o proprietate exprimată într-un limbaj logic determină dacă modelul satisface proprietatea și dacă nu furnizează utilizatorului un contraexemplu (3) raționament deductiv în care teoremele sunt căutate să fie demonstrate pe modele software.

> Citiți capitolul complet

Introducere

Daniel Aarno, Jakob Engblom, înDezvoltare software și sisteme folosind platforme virtuale, 2015

Testare de integrare

Fluxul clasic MDD nu atinge cu adevărat problema integrării sistemului. Accentul este pe generarea de cod corect pentru funcționalitatea de bază a sistemului. Codul respectiv va avea nevoie de un sistem de operare și o platformă hardware pentru a rula în lumea reală, iar integrarea cu platforma respectivă are loc adesea destul de târziu în ciclul de proiectare a sistemului. Într-adevăr, chiar și testarea HIL este adesea efectuată folosind hardware de dezvoltare, mai degrabă decât sistemul final.

Simics poate fi folosit pentru a muta integrarea mai devreme și pentru a permite testarea mai devreme a integrării. După cum se arată în Figura 1.6, cu un model Simics al sistemului țintă, portul OS la hardware-ul țintă și integrarea sistemului de operare și a aplicațiilor pot fi testate fără hardware.

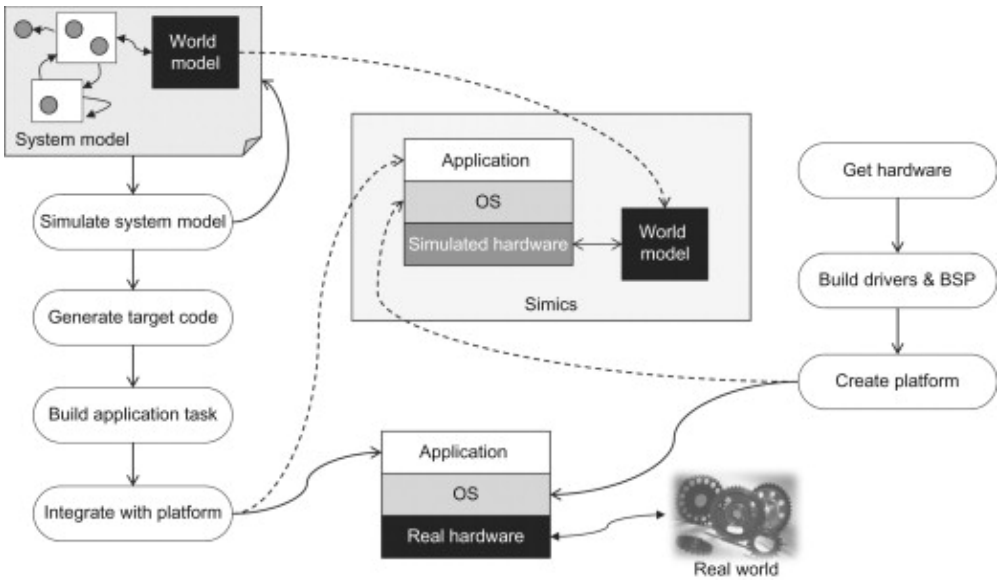

Figura 1.6. Testarea integrării modelelor cu Simics.

Ceva pe care Simics îl face posibil este să testeze dacă aplicația funcționează cu intrarea și ieșirea reală, așa cum sunt furnizate de sistemul de operare și platforma hardware țintă, folosind în continuare un model al lumii. Astfel, este posibil să se creeze un mediu de integrare complet virtual, în care hardware-ul, sistemul de operare, aplicațiile care conțin algoritmi de control și lumea pot fi rulate împreună în simulare.

> Citiți capitolul complet

Efectul factorilor umani asupra comportamentului șoferului

Jianqiang Wang, ... Xiao-Yun Lu, înProgrese în vehiculele inteligente, 2014

Abstract

Comportamentul șoferului este unul dintre cele mai importante aspecte în proiectarea, dezvoltarea și aplicarea sistemelor avansate de asistență la conducere (ADAS) și a sistemelor inteligente de transport (ITS), care pot fi afectate de mulți factori. Pentru a studia efectul factorilor umani în conducere, au fost dezvoltate un simulator cu Hardware-in-the-Loop (HIL) și Driver-in-the-Loop (DIL) și un vehicul instrumentat pentru a măsura cantitativ acțiunile șoferului, starea vehiculului. , și relația cu vehiculul. O analiză comparativă a diferențelor a fost efectuată într-un simulator de conducere și, de asemenea, în mediul real. După ce au fost efectuate 33 de experimente diferite de șofer, a fost dezvoltat și un set de programe speciale de procesare a datelor pentru a revizui, extrage și analiza datele capturate. Unii parametri reprezentativi, cum ar fi timpul până la coliziune (TTC) și timpul de avans (THW) sunt adoptați pentru a analiza și compara efectul factorilor umani, care includ vârsta, sexul, experiența de conducere, volumul de muncă, nivelul de educație și naționalitatea. În cele din urmă, pentru a asigura consistența clasificărilor bazate pe evaluarea subiectivă și pe datele experimentale obiective, teste care includ completarea unui chestionar de comportament al șoferului (DBQ) și conducerea în lumea reală au fost efectuate cu 52 de participanți și comparate. Teoria analizei factoriale a fost aplicată la prelucrarea datelor și extragerile caracteristice. Rezultatele studiului au indicat că au existat diferențe semnificative între evaluarea subiectivă și datele obiective, adică evaluarea subiectivă nu a fost suficient de fiabilă pentru proiectarea algoritmului, deși poate fi folosită ca referință.

> Citiți capitolul complet

Utilizarea simularii pentru cercetare

Thomas W. Edgar, David O. Manz, înMetode de cercetare pentru securitatea cibernetică, 2017

Instanciarea unui model

După cum am menționat anterior, un model este o reprezentare abstractă sau conceptuală a unui sistem. Modelele în sine nu sunt executabile. În schimb, este nevoie de un instrument pentru a transforma un model într-un proces executabil care generează date. Simularea este procesul prin care un model este transformat de la descriptiv la informativ.

mativ. Simulările necesită un cadru de programare prin care poate fi specificat un model și apoi o platformă executabilă care poate rula modelul în diferite condiții specificate. O simulare rulează prin scenarii de intrări cu un model pentru a genera date despre starea modelului și rezultatul care ar fi produs. De exemplu, unul dintre tipurile comune de simulatoare pe care le vom discuta este un simulator de rețea. Simulatoarele de rețea preiau un model al unei rețele și permit simularea comportamentului pe măsură ce comunicarea trece prin aceasta. Simulatoarele de rețea le permit cercetătorilor să studieze noi protocoale, cum ar fi modul în care un nou protocol fără fir va funcționa în diferite configurații ale unei rețele.

Tipuri de simulare

Există mai multe moduri de a defini un model și diferite platforme de simulare. Simulare bazată pe agențifolosește modele de agenți care funcționează independent. Aceste tipuri de simulări sunt bune pentru a descoperi comportamentul emergent atunci când există un control distribuit al unui sistem.Simulări bazate pe procesutilizați o definiție matematică a unui sistem pentru a genera comportament dintr-un proces. Simulările bazate pe procese sunt bune pentru simularea proceselor fizice care pot fi definite cu matematică continuă.Simulări bazate pe urme executați aplicații la scară mică pentru a genera jurnalele de comportament real, iar apoi cadrul de simulare interpolează aceste jurnale la o scară mult mai mare. Simulările bazate pe urme sunt bune pentru înțelegerea performanței scalabilității sistemelor. În cele din urmă, simulările Monte Carlo execută o serie de teste pentru a determina probabilitatea rezultatelor. Simulările Monte Carlo sunt bune pentru sprijinirea deciziilor sau atunci când există o cantitate mare de incertitudine asupra parametrilor de intrare.

Fiecare dintre aceste abordări are compromisuri. Unele oferă modelare mai ușoară. De exemplu, modelarea agenților independenți este mai ușoară decât modelarea unui sistem complet cu interdependențe. Deciziile pe care le iei cu privire la modele afectează performanța și scalabilitatea. Un model bazat pe matrice, cum ar fi rețeaua electrică, nu este ușor de descompus în subprobleme, ceea ce face dificilă scalarea.

Deși acestea sunt modalități generale de simulare a sistemelor, există tipuri specifice de simulare care sunt utile pentru cercetarea securității cibernetice. Următoarea listă oferă o imagine de ansamblu bună a diferitelor tipuri de simulare care ar putea fi utile pentru a vă ajuta să răspundeți la întrebările dvs. de cercetare.

Simulatoare de rețea

- *Simulatoare bazate pe containere*: Simulatoarele bazate pe containere folosesc virtualizarea ușoară pentru a permite generarea unei rețele de noduri de comunicare pe unul sau un număr mic de computere fizice. Simulatoarele bazate pe containere sunt bune în privința noilor comportamente ale protocolului de rețea. Cu toate acestea, ca toate

containerele vor folosi exact aceeași stivă de comunicare, nu sunt ▪ un bun simulator pentru teste de eterogenitate a comportamentului unic de răspuns. Aceste simulatoare au fost create pentru rețele definite de software,[11]aplicații fără fir,[12]și rețele generale.[13]

Simulatoare de evenimente discrete: Simulatoarele de evenimente discrete generează secvențe de evenimente discrete. Deoarece spațiul cibernetic este un spațiu discret, simulatoarele de evenimente discrete sunt potrivite pentru simularea unor aspecte ale acestuia. În special, simulatoarele de evenimente discrete au fost metoda principală de simulare a rețelelor și protocoalelor. Există mai multe simulatoare de rețea de evenimente discrete; comercial[14,15]și open source.[16,17]Fiecare are caracteristici unice, cum ar fi interfața grafică cu utilizatorul (GUI) de configurare și analiză, caracteristici de emulare a rețelei pentru hardware-in-the-loop și diverse modele de protocol și fizice acceptate.

Sapă mai adânc:Hardware-in-the-loop

Hardware-in-the-loop este un tip special de simulare care fuzează liniile cu experimentarea. Simularea hardware-in-the-loop oferă capabilități de emulare care permit integrarea echipamentelor reale în simulare. În general, un sistem mai mare este simulat cu câteva dispozitive reale pentru a permite răspunsul de înaltă fidelitate al echipamentelor reale, permițând în același timp amploarea simulării. Unul dintre motivele principale din spatele reproiectării lui ns-2 în ns-3 a fost acela de a permite testarea hardware-in-the-loop pentru a permite o progresie mai rapidă a evaluării implementărilor reale.

Generatoare de trafic

Acolo unde simulatoarele de rețea încearcă să modeleze și să captureze comportamentul realist și ieșirea rețelelor de comunicații, generatorii de trafic sunt, în general, preocupați doar de modelarea și simularea pachetelor de comunicații și a sarcinilor utile care ar fi produse de dispozitivele dintr-o rețea. Sau altfel spus, generatorii de trafic simulează multe dispozitive dintr-o rețea și comunicarea pe care le-ar produce, dar nu sunt preocupați să măsoare acel trafic în rețea. Generatoarele de trafic sunt adesea folosite pentru experimentarea aplicată pentru a investiga performanța infrastructurii de rețea, a senzorilor și a controalelor de securitate. Există simulatoare generale de trafic care generează diverse protocoale și comportamente.[18,19]În special pentru aplicațiile de securitate există diverse instrumente de generare a traficului care generează comportamente de atac.[20,21,22]Există, de asemenea, generatoare de trafic cu fidelitate mai mare care, în loc să simuleze traficul, emulează comportamentul utilizatorului pentru a genera trafic real din aplicații reale.[23,24]Utilizatorii emulați sunt utili atât pentru experimente aplicate, cât și pentru experimente fundamentale, pentru a oferi zgomot de fond pentru atacatori și apărători.

Simulare țintă

Deoarece înțelegerea atacatorilor este o parte importantă a cercetării pentru securitatea cibernetică, instrumentele de testare a acestora sunt importante. Deși este posibil să se efectueze studii asupra incidentelor cibernetice reale, numărul și frecvența acestora sunt încă relativ scăzute și este posibil să nu ofere răspunsuri definitive la întrebările deschise. Capacitățile de simulare a țintei sunt concepute pentru a oferi o modalitate controlată de stimulare a atacatorilor pentru studiu. Instrumentele de simulare a țintei (aka honeypots) oferă o modalitate de a crea ținte vulnerabile pentru a atrage atacatorii să le exploateze. Aceste instrumente sunt adesea numite pe o anumită variație de miere, bazată pe conceptul de captare a mierii, care este o tehnică de spionaj de utilizare a unei ținte ademenitoare din punct de vedere fizic pentru a atrage activele în divulgarea secretelor. Instrumentele de simulare a țintei oferă adesea senzori suplimentare pentru a monitoriza acțiunile atacatorilor.[25]rețele de honeypots,[26]browsere,[27]și, chiar și date.[28]

Știați?

Termenul*borcan cu miere*are o bază în lumea spionajului. Honeypotting sau capcana cu miere se referă la utilizarea seducției sexuale pentru a recruta bunuri. Tehnicile de honeypotting cibernetice au fost numite la fel prin utilizarea sistemelor „atrăgătoare”, bazate pe vulnerabilități ușoare, pentru a atrage atacatorii.

Simularea amenințărilor

Simularea amenințărilor încearcă să recreeze amenințările și pericolele la adresa unui sistem pentru a observa și analiza modul în care sistemele sau utilizatorii sistemului reacționează. Suita de instrumente care se încadrează în simularea amenințărilor are utilizări variate. Unele sunt utile pentru studii aplicate pentru înțelegerea sistemelor, altele oferă instrumente utile pentru experimentare și, în cele din urmă, altele oferă capabilități pentru a sprijini validarea soluțiilor. Există mai multe categorii de instrumente de simulare a amenințărilor care sunt utile în modelarea diferitelor sarcini și comportamente ale unui atacator. Simulatoarele de eșecuri, scanerele de vulnerabilități și platformele de testare a exploatării sunt discutate în această secțiune, dar consultați Capitolul 14, Tratarea adversarului, pentru o discuție mai aprofundată despre modelarea atacatorilor.

- *Simulatoare de eșec*:Simulatoare de eșecsimulează pericole pentru un sistem sau defecțiuni ale unui sistem. Simulatoarele de defecțiuni degradează sau împiedică performanța unei părți a sistemului de testare. Acest lucru oferă capacitatea de a testa și evalua acțiunile de remediere și măsurile de robustețe. Netflix a creat o suită de instrumente pentru a testa rezistența la eșec a infrastructurii cloud.[29]Alte instrumente oferă mecanisme de testare a condițiilor degradate ale rețelei.[30]În cele din urmă, există un tip special de simulator de defecțiuni, numit fuzzers, care utilizează protocoale sau API-uri în mod necorespunzător pentru a testa robustețea soluțiilor în primirea intrărilor proaste. Fuzzers sunt un instrument obișnuit pentru studierea securității implementărilor software în căutarea posibilelor vulnerabilități. Fuzzers vin în toate tipurile de arome; cadre fuzzer,[31,32]fuzzers în format de fișier,[33,34]fuzzeri de protocol,[35,36]și fuzzere de aplicare.[37,38,39]

- *Scanere de vulnerabilitate*:Scanere de vulnerabilitatesunt instrumente care caută semnături software și servicii pentru a determina dacă au vulnerabilități. Scanerele de vulnerabilitate simulează procesul atacatorilor în găsirea vulnerabilităților sistemului pentru exploatare. Cu toate acestea, este important de menționat că scanerele de vulnerabilitate sunt adesea concepute ca instrumente de audit și nu sunt concepute pentru a modela cu acuratețe tacticile ascunse ale atacatorilor. Rezultatele acestor instrumente pot oferi o colecție rezonabilă de cunoștințe care ar fi obținută de un atacator. Există open source[40]si comerciale[41]instrumente de scanare a vulnerabilităților.
- *Exploatați platformele de testare*:Exploatați platformele de testarefurnizați pachete și un cadru pentru a executa exploit-uri împotriva vulnerabilităților cunoscute. Platformele de testare a exploatării oferă un mijloc rezonabil de simulare a capacităților atacatorilor de a efectua studii sau experimente. Cu toate acestea, până când exploit-urile sunt adăugate pentru a exploata platformele de testare, acestea sunt în general departe de a fi zero zile. Prin urmare, kiturile de exploatare nu sunt adesea potrivite pentru modelarea tacticilor de ultimă oră pentru atacatori. Metasploit[42]este un cadru de exploatare cu sursă deschisă bine-cunoscut, care a generat câteva suplimente.[43,44]Pânză[45]și Core Impact[46]
  sunt principalele cadre de exploatare comercială.

Știați?

Zero days sunt vulnerabilități cunoscute de atacatori, dar necunoscute publicului sau dezvoltatorilor. Termenul lor*zi zero*își are originea în comunitatea warez, care era un grup de oameni care împărtășeau software piratat. Software-ul Zero Day Warez se referea la software care nu era încă lansat publicului. Comunitățile warez și hackeri au fost polenizate încrucișat, ceea ce a condus la aplicarea termenului pentru vulnerabilități.

Simulare generală

Restul instrumentelor de simulare pe care le-am discutat s-au concentrat în mod special pe spațiul cibernetic și securitate. Cu toate acestea, există o mulțime de simulatoare de uz general și specifice domeniului care ar putea fi utile în funcție de subiectul dvs. de cercetare. Inginerii folosesc MATLAB și Simulink pentru a modela procesele fizice. Modelica este o platformă integrată de modelare pentru modelarea multor procese diferite. Există instrumente precum Portico,[47]Ptolemeu,[48]și FNCS[49]care sunt concepute pentru a integra și a pune în legătură diferite simulatoare de diferite tipuri. Alegeți simulatorul potrivit pentru cercetarea pe care o faceți. Doar pentru că unele simulatoare nu sunt concepute pentru securitatea cibernetică nu înseamnă că nu sunt cele mai potrivite pentru a răspunde la întrebarea dvs. de cercetare.

> Citiți capitolul complet

Rapid Control Prototyping (RCP) pentru un sistem de mișcare1

George Ellis, înGhid de proiectare a sistemului de control (ediția a patra), 2012

19.1 De ce să folosiți RCP?

Există cel puțin două moduri de a utiliza RCP și beneficiile depind de care este selectată:

1. Îmbunătățirea și validarea modelului În majoritatea proiectelor industriale este suficient să existe un sistem fizic funcțional; de obicei este inutil să aveți un model foarte detaliat. Cu toate acestea, există multe excepții, inclusiv: • Când există dorința de a îmbunătăți o plantă complexă, dar nu este clar ce caracteristici ar trebui modificate. Având un model detaliat, precis, poate indica modificările care pot beneficia de performanța sistemului; fără un model detaliat, efectuarea de modificări la instalație poate fi greșită sau greșită. • Proiectul poate avea nevoie de Hardware în buclă (HIL) (Secțiunea 13.2.3) pentru a fi utilizat pentru testarea produsului. Prin definiție, HIL necesită un model detaliat și precis al centralei, al convertorului de putere și al dispozitivelor de feedback.•Pentru aplicații de înaltă fiabilitate, cum ar fi aerospațiale și apărare,
   sistemele modelate și fizice ar trebui să arate rezultate aproape identice.•Pentru cercetarea sistemelor de control, este adesea important să se valideze modelul pentru a demonstra că soluția va funcționa pe un set larg de hardware. Arătarea unei metode lucrate pe o anumită plantă este insuficientă atunci când se propune o soluție generală.

2. Oferirea accesului la componentele fizice pentru a înlocui modelul Atunci când se elaborează legi de control, de obicei este nevoie de multă experimentare și validare asupra sistemului fizic. Aici, modelul poate fi punctul de plecare pentru dezvoltare, deoarece dezvoltarea inițială este mult mai eficientă pe sistemele simulate. După implementarea inițială, performanța hardware-ului fizic este adesea preocuparea dominantă, iar sistemul RCP poate înlocui sistemul simulat ca mediu de dezvoltare principal.

19.1.1 Utilizarea RCP pentru îmbunătățirea și validarea modelului

Înțelegerea exactă a modului în care se comportă sistemul fizic (adică având un model detaliat și precis) este de obicei un proces iterativ:

A. Începeți cu modelul așa cum este cunoscut,

B. Execută legile de control asupra modelului respectiv și asupra sistemului fizic,

C. Comparați rezultatele și,

D. Dacă există diferențe semnificative, îmbunătățiți modelul și reveniți la Pasul B.

Acest proces se repetă până când rezultatele modelului și ale sistemului fizic se potrivesc suficient de bine. Executarea acestui proces fără RCP implică compararea unui sistem complet simulat cu un sistem fizic. Acest lucru necesită schimbarea simultană atât a legilor de control, cât și a modelelor de convertor de putere/centrală/senzor de feedback. Când rezultatele sistemelor simulate și fizice diferă, este adesea neclar dacă aceste diferențe au apărut din inexactitatea modelului componentelor fizice sau dintr-o eroare în conversia legilor de control. RCP permite utilizarea acelorași legi de control în ambele cazuri, izolând astfel majoritatea problemelor de modelele componentelor fizice.

Procesul de dezvoltare a unui model precis este de obicei iterativ, deoarece nu este practic să se includă fiecare efect cunoscut al fiecărei componente din sistem. Convertizoarele de putere, motoarele, sarcinile și dispozitivele de feedback au toate efecte liniare și neliniare complexe. De exemplu, mulți senzori de feedback ale motorului au un cuplaj mecanic conform cu arborele motorului, care este de obicei dificil de caracterizat; de cele mai multe ori, efectul său este nesemnificativ, dar în rare ocazii poate fi limita de performanță dominantă. De unde știi de la început dacă ar trebui să investești timp pentru a modela acest efect? De asemenea, frecarea are multe componente (Secțiunea 12.4.5) și dacă nu se știe că o anumită componentă este semnificativă, efortul de a o modela cu acuratețe poate fi excesiv (în special efectele Dahl și Stribeck). Asa de,

19.1.2 Utilizați RCP pentru a oferi acces la componentele fizice și pentru a înlocui modelul

Majoritatea sistemelor de control se bazează pe produse standard care utilizează legi de control predeterminate, cum ar fi buclele de viteză PI. Cu toate acestea, unele sisteme necesită legi de control specializate pentru a îmbunătăți performanța. De exemplu, poate fi necesar să compensați un anumit comportament neliniar care nu a fost anticipat de către furnizorul unui produs standard. Sau poate fi necesar să executați un filtru neobișnuit sau să utilizați o lege nouă de control, care nu este disponibilă pentru produsele standard.

Majoritatea designerilor încep dezvoltarea legilor de control personalizate în simulare completă și apoi port acele legi într-un sistem fizic. Când legile de control sunt executate pe sistemul fizic, este obișnuit ca rezultatele să nu se potrivească cu modelul de la început. De obicei, urmează un proces plictisitor pe măsură ce designerul încearcă să rezolve diferențele. Ca și în secțiunea anterioară, fără RCP, atât legile de control, cât și instalația/convertorul de putere/senzorul de feedback trec de la simulare la hardware fizic la

o dată, așa că nu este clar dacă cauza principală a unei probleme ar putea proveni dintr-o eroare în conversia legii de control sau o eroare în modelul uneia dintre componente. Folosind RCP, legile de control pot fi validate complet pe instalația fizică înainte de a fi transferate la un controler încorporat.

RCP oferă un set larg de instrumente de proiectare: acces ușor la diagramele din domeniul timpului și măsurarea tuturor semnalelor interne ale legilor de control, instrumente versatile de generare a diagramelor Bode și limbaje grafice pentru proiectarea legilor de control. Limbajele grafice oferă legi de control standard (cum ar fi PI și PID), o gamă largă de filtre și o gamă variată de efecte liniare și neliniare pentru a permite proiectanților să pună laolaltă o mare gamă de legi de control. Algoritmii sunt de obicei executați în virgulă mobilă și, prin urmare, sunt ușor de introdus într-un sistem fără problemele de saturație, rollover și cuantizare atât de des întâlnite cu implementările de matematică întregi comune în sistemele încorporate. Luate împreună, accesul la aceste instrumente de proiectare poate accelera timpul pentru a încerca idei noi și a valida operarea.

Desigur, legile de control ar putea trebui în cele din urmă să fie convertite într-un controler încorporat, deoarece controlerele încorporate sunt de obicei mai mici, mai puțin costisitoare și mai fiabile decât sistemele RCP. În acel moment, proiectantul va trebui să se ocupe de probleme complexe rezultate din conversia legii de control, mai ales dacă controlerul încorporat se bazează pe matematica întregului. Cu toate acestea, în acel moment, proiectantul va fi încrezător că legile de control funcționează pe sistemul fizic, astfel încât marea majoritate a problemelor pot fi izolate de conversia legii de control pentru a rula pe controlerul încorporat.

Restul acestui capitol va demonstra pașii inițiali pentru proiectarea unui sistem de control al mișcării RCP. În Secțiunea 19.2, un servosistem cu o sarcină cuplată rigid va fi convertit de la simulare la RCP; în secțiunea 19.3, un cuplaj conform va fi înlocuit și procesul va fi repetat.

> Citiți capitolul complet

Învățare profundă pentru navigarea bazată pe viziune în cursele autonome cu drone

Huy Xuan Pham, ... Erdal Kayacan, înÎnvățare profundă pentru percepția și cunoașterea roboților, 2022

15.4.1 Medii de simulare pentru cursele autonome cu drone

Simulările au fost mult timp un instrument valoros pentru dezvoltarea vehiculelor robotizate. Permite inginerilor să identifice defecțiunile la începutul procesului de dezvoltare și permite științei

Știștii prototipează rapid și își demonstrează ideile fără riscul de a distruge hardware potențial costisitor. În timp ce sistemele de simulare au diverse beneficii, mulți cercetători văd rezultatele generate în simulare cu scepticism, deoarece orice sistem de simulare este o abstractizare a realității și va varia de la realitate la o anumită scară. În ciuda scepticismului cu privire la rezultatele simulării, au apărut câteva tendințe care au determinat comunitatea de cercetare să dezvolte sisteme de simulare mai bune de necesitate în ultimii ani. O tendință majoră de conducere către simulatoare realiste provine din apariția unor metode algoritmice bazate pe date în robotică, de exemplu, bazate pe învățarea automată care necesită date extinse sau RL care necesită un mediu de învățare sigur pentru a genera experiențe. Sistemele de simulare oferă nu numai cantități mari de date, ci și etichetele corespunzătoare pentru antrenament, ceea ce îl face ideal pentru aceste metode bazate pe date. Această tendință de conducere a creat o nevoie critică de a dezvolta sisteme de simulare mai bune și mai realiste. Simulatorul ideal are trei caracteristici principale:

1. Colectare rapidă de cantități mari de date cu timp limitat și calcule.

2. Acurate din punct de vedere fizic pentru a reprezenta dinamica lumii reale cu fidelitate ridicată.

3. Fotorealistic pentru a minimiza discrepanța dintre simulări și observațiile reale ale senzorului.

Aceste obiective sunt, în general, de natură conflictuală: cu cât acuratețea este mai mare într-o simulare, cu atât este nevoie de mai multă putere de calcul pentru a oferi această precizie, deci un simulator mai lent. Prin urmare, atingerea tuturor acestor obiective într-un singur simulator uniform este o provocare. Cele trei instrumente de simulare care au încercat să echilibreze aceste obiective într-un simulator UAV sunt FlightGoggles [87], AirSim [86] și Flightmare [88]. Aceste instrumente de simulare folosesc motoare moderne de jocuri 3D, cum ar fi unitatea sau motorul ireal, pentru a produce imagini fotorealiste de înaltă calitate în timp real.

15.4.1.1 AirSim

În 2017, Microsoft a lansat prima versiune de simulare a informatică și robotică aeriană (AirSim). AirSim este un simulator fotorealist open-source pentru vehicule autonome construite pe un motor ireal. Un set de interfețe de programare a aplicațiilor (API), scrise fie în C++, fie în Python, permite comunicarea cu vehiculul, indiferent dacă acesta observă fluxurile senzorilor, coliziunea, starea vehiculului sau trimite comenzi către vehicul. În plus, AirSim oferă o interfață pentru a configura mai multe modele de vehicule pentru quadrotoare și acceptă hardware-in-the-loop, precum și software-in-the-loop cu controlere de zbor precum PX4.

AirSim diferă de celelalte două simulatoare, deoarece modelul dinamic al vehiculului este simulat folosind motorul de fizică NVIDIA PhysX, un motor de fizică popular folosit de marea majoritate a jocurilor video de astăzi. În ciuda popularității în gaming

industrie, PhysX nu este specializat pentru quadrotori și este strâns cuplat cu motorul de randare pentru a simula dinamica mediului. AirSim realizează unele dintre cele mai realiste imagini, dar datorită acestei conexiuni stricte între randare și simularea fizică, AirSim poate atinge doar viteze limitate de simulare. În Fig. 15.16, sunt prezentate exemple de informații fotorealiste colectate prin AirSim, oferind informații RGB, segmentate și de profunzime.

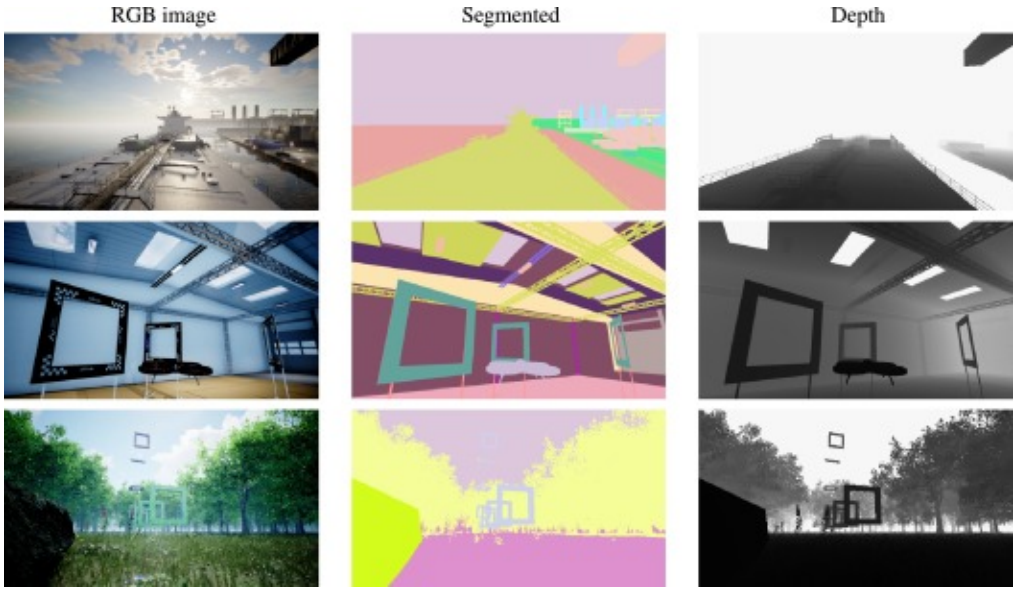

Figura 15.16. Informații RGB, segmentate și de profunzime extrase din trei medii diferite folosind AirSim. Cele trei medii sunt create folosind un motor ireal și sunt prezentate; port (rândul de sus), laboratorul de drone (rândul din mijloc) și în aer liber (rândul de jos).

15.4.1.2 FlightGoggles

În 2019, FlightGoggles este dezvoltat la Massachusetts Institute of Technology ca un simulator de senzor fotorealistic open source pentru vehicule robotizate bazate pe percepție. Contribuția de la FlightGoggles constă din două componente separate. În primul rând, decuplarea motorului de randare fotorealistă de modelarea dinamică oferă utilizatorului flexibilitatea de a alege complexitatea modelelor dinamice. Scăderea complexității permite motorului de randare mai mult timp pentru a genera date de percepție de calitate superioară cu prețul preciziei modelului, în timp ce prin creșterea complexității, sunt alocate mai multe resurse pentru modelarea cu acuratețe. În al doilea rând, furnizarea unei interfețe cu vehicule și actori din lumea reală într-un sistem de captare a mișcării pentru testarea vehiculului în buclă și a omului în buclă. Utilizarea sistemului de captare a mișcării este foarte utilă pentru redarea imaginilor camerei date traiectorii și măsurători inerțiale de la vehicule zburătoare în lumea reală, în care setul de date colectat este utilizat pentru testarea algoritmilor bazați pe viziune. Capacitatea de a efectua experimente vehicul-in-the-loop cu simularea senzorului fotorealistic facilitează noi direcții de cercetare care implică, de exemplu, zbor autonom rapid și agil în medii bogate în obstacole, interacțiune umană sigură.

15.4.1.3 Flightmare

În 2020, grupul de robotică și percepție de la Universitatea din Zurich a lansat prima versiune a Flightmare: un simulator de quadrotor flexibil. Flightmare împărtășește aceeași motivație ca și FlightGoggles prin decuplarea modelării dinamice de motorul de randare fotorealistă pentru a obține flexibilitatea de a alege cât timp vrem să simulăm un model precis în comparație cu colectarea rapidă a datelor. Deși nu încorporează sistemul de captare a mișcării, Flightmare oferă interfețe pentru faimosul simulator robotic Gazebo împreună cu diferite motoare fizice de înaltă performanță. Flightmare poate simula câteva sute de agenți în paralel prin decuplarea modulului de randare de motorul fizic. Acest lucru este util pentru aplicațiile multidrone și permite colectarea și instruirea extrem de rapidă a datelor, ceea ce este crucial pentru dezvoltarea aplicațiilor DRL. Pentru a facilita și mai mult RL, Flightmare oferă un înveliș standard (OpenAI Gym), împreună cu linii de bază populare OpenAI pentru algoritmi RL de ultimă generație. În cele din urmă, Flightmare contribuie cu o suită de senzori multimodali considerabile, oferind: IMU, RGB, segmentare, adâncime și un API pentru a extrage informațiile 3D complete ale mediului sub forma unui nor de puncte.

Această secțiune prezintă unele dintre simulatoarele de percepție de înaltă calitate de pe piață. Tabelul 15.3 rezumă principalele diferențe ale simulatoarelor prezentate, dar alegându-l pe cel potrivit în funcție de problemă și de ce date sunt esențiale pentru proiect. De exemplu, dacă imaginile fotorealiste de înaltă calitate sunt prioritatea proiectului, AirSim ar putea fi cel mai bun, dar dacă este nevoie de o evaluare a performanței quadrotorului, poate că FlightGoggles cu vehiculul său în buclă este alegerea potrivită. Aceste instrumente de simulare sunt îmbunătățite și extinse în fiecare zi, așa că nimeni nu poate prezice simulatorul potrivit de utilizat în viitor, dar sperăm că această secțiune a scos în lumină modul în care simulatoarele pot fi un instrument util pentru antrenament, testare și prototip. și cât de valoroasă este pentru vehiculele autonome.

Tabelul 15.3. Compararea caracteristicilor simulatorului.

Simulator	Redare	Dinamica	Senzor suită	Mișcare captură	RL API	Vehicule
AirSim	Ireal En-motor 4	PhysX	IMU, RGB, Adâncime, Seg	Nu	Nu	Multiplu
FlightGog-gles	Unitate	Flexibil	IMU, RGB	da	Nu	Singur
Flightmare	Unitate	Flexibil	IMU, RGB, Adâncime, Seg	Nu	da	Multiplu

> Citiți capitolul complet

Modelare multi-paradigma și co-simulare în prototiparea unui sistem de producție ciber-fizic

Mihai Neghină, ... Ken Pierce, înAbordări de modelare multi-paradigma pentru sistemele ciberfizice, 2021

7.3 Tehnici

Această secțiune prezintă principalele tehnologii utilizate pentru construirea modelelor inițiale, inclusiv tehnologia de co-simulare INTO-CPS și modul de generare a modelelor Discrete-Event First (DE-first) în Overture [10]. Această secțiune prezintă, de asemenea, deficiențe în aplicarea directă a fluxului de lucru al INTO-CPS și motivația pentru o abordare alternativă.

7.3.1 Tehnologia INTO-CPS

Tehnologia de co-simulare INTO-CPS este o colecție de instrumente și metode care au fost legate pentru a forma un lanț de instrumente pentru proiectarea CPS-urilor bazată pe model. În loc să suprima diversitatea formalismelor software, de control și mecatronică necesare în proiectarea CPS-urilor prin solicitarea tuturor disciplinelor să adopte aceeași notație cu scop general, INTO-CPS îmbrățișează această diversitate integrându-le la nivel semantic [11-15], permițând inginerilor să colaboreze folosind tehnici și metode familiare de modelare. Fluxul general de lucru și serviciile din lanțul de instrumente utilizate în acest proiect sunt ilustrate în Fig. 7.4.

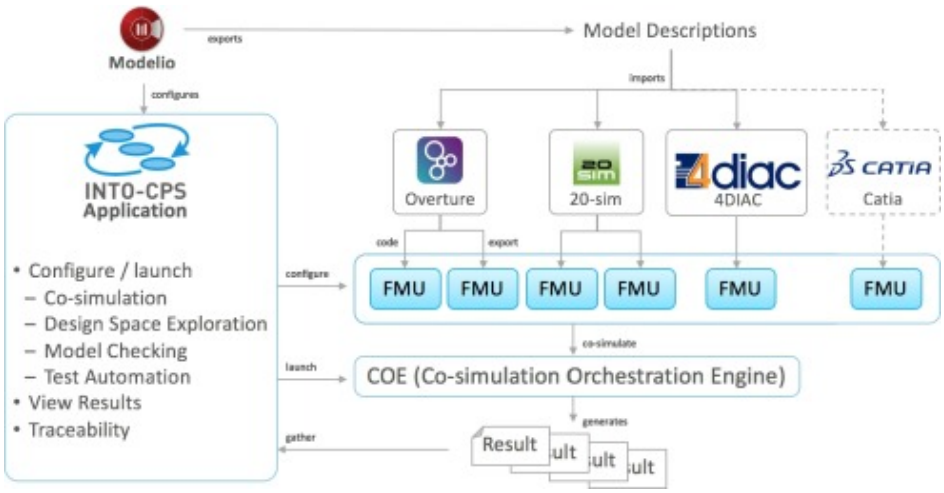

Figura 7.4. Lanțul de instrumente INTO-CPS utilizat în acest proiect.

Lanțul de instrumente INTO-CPS este centrat pe simularea simultană a modelelor eterogene folosind standardul Functional Mock-up Interface (FMI) [16], care permite ca modele din diferite instrumente și formalisme să fie împachetate ca unități de modele funcționale.

(FMU-uri). Astfel de modele pot fi combinate și analizate prin co-simulare, fiecare FMU acționând ca o unitate de simulare independentă și având o descriere a modelului care include interfața sa. FMU-urile pot fi furnizate și ca cutii negre pentru a proteja proprietatea intelectuală (IP) conținută în detaliile modelului.

INTO-CPS include un motor de co-simulare, numit Maestro [17], care implementează complet versiunea 2.0 a standardului FMI și a fost testat cu succes cu peste 30 de instrumente [16]. În jurul acestui nucleu de co-simulare, lanțul de instrumente INTO-CPS leagă instrumente suplimentare pentru a sprijini proiectarea bazată pe model pe parcursul dezvoltării. Un profil Systems Modeling Language (SysML) este furnizat și susținut de instrumentul Modelio [18], permițând descrierilor modelelor FMU să fie capturate și legate la cerințe. Descrierile modelului pot captura atât părți fizice, cât și cibernetice ale sistemului, permițând atât simularea Hardware-in-the-Loop (HiL) cât și Software-in-the-Loop (SiL) și pot fi exportate pentru utilizare în instrumente de modelare [6]]. De asemenea, profilul permite acestor descrieri ale FMU-urilor să configureze co-simularile și alte forme de analiză susținute de INTO-CPS.

Suport specific pentru importul descrierilor de model și producerea modelelor de schelet este oferit de instrumentele din lanțul de instrumente INTO-CPS: Overture [10], care sprijină modelarea DE; 20-sim [19] și OpenModelica [20], ambele suportând modelarea CT. Aceste instrumente garantează, de asemenea, exportul de FMU, care pot fi apoi utilizate într-o co-simulare cu Maestro [17]. Exportul FMU este inclus într-un număr tot mai mare de instrumente industriale. În timpul co-simulării eterogene din etapele ulterioare ale experimentului nostru, am folosit 4DIAC [21], un instrument open-source pentru dezvoltarea sistemelor de control industrial, și CATIA [22], o suită de software standard industrial pentru proiectare și proiectare asistată de computer. de fabricație. Ambele sunt prezentate în pozițiile lor respective în cadrul lanțului de scule INTO-CPS în Fig. 7.4.

7.3.2 Modele inițiale

Abordarea generării directe a descrierilor modelelor FMI din profilurile SysML și importării acestora în instrumente de modelare dedicate, cu așteptarea ca FMU-urile generate să se combine perfect în co-simulare, poate fi predispusă la eșec. Deși permite diferitelor echipe să lucreze separat la modelele constitutive, necesită ca UMF din toate echipele să fie disponibile înainte de testarea integrării prin co-simulare. Orice întârziere în generarea oricărei componente duce la întârzieri suplimentare în co-simulare și la descoperirea târzie a problemelor. În mod similar, dacă există o singură echipă care produce toate FMU-urile secvenţial, în întreaga lor complexitate, co-simularea poate începe doar la sfârșitul modelării.

O strategie potențială pentru a atenua aceste riscuri este ca fiecare echipă să producă versiuni inițiale rapide ale FMU-urilor cât mai curând posibil și să efectueze teste de integrare cu aceste modele. Modelele inițiale pot fi apoi actualizate într-o manieră iterativă către mai multe

modele detaliate, cu fiecare echipă capabilă să se miște în propriul ritm și cu versiunile anterioare care oferă soluții de rezervă în cazul problemelor și linii de bază pentru testarea regresiei. Totuși, această abordare ar putea fi mai dificilă în unele paradigme de modelare, unde modelele rapide și simple ar putea să nu funcționeze suficient de bine pentru testare.

Abordarea DE-first adoptată pentru proiectul IPP4CPPS urmează strategia de a produce UMF inițiale și de a le înlocui cu modele mai detaliate pe măsură ce devin disponibile. Mai degrabă decât utilizarea fiecărui formalism individual, un singur formalism DE precum Overture [10] poate fi utilizat pentru toate modelele inițiale, generând astfel rapid modelul abstract al întregului sistem. Schițarea comportamentului modelelor componente permite testarea timpurie a ipotezelor la începutul procesului. Un formalism DE ar trebui selectat deoarece acestea sunt concepute pentru a capta comportamente abstracte și logice, adesea descrise în termeni de interfețe și, prin urmare, sunt potrivite pentru această sarcină [23].

7.3.3 Evenimentul discret mai întâi cu VDM-RT/Overture

Aplicarea unei abordări DE-first la o setare FMI folosind Overture [10] urmează principiile Vienna Development Method (VDM) [24,25], un set de tehnici de modelare aplicate cu succes atât în cercetare, cât și în aplicații industriale. Proiectul inițial VDM Real-Time (VDM-RT) [7] conține o clasă pentru fiecare FMU, cu obiecte de port corespunzătoare interfeței prezentate în descrierea modelului, o clasă principală (de sistem) care instanțează obiecte de port adecvate și instanțe ale fiecăruia. Clasa FMU la care trece porturile și o clasă mondială care oferă o metodă ca punct de intrare pentru simulare, pornind firele de execuție ale obiectelor și blocurilor FMU până la finalizarea simularii.

Fig. 7.5 oferă diagrame de clasă și obiect și prezintă o astfel de configurație folosind două modele constitutive, FMU1 și FMU2. Un astfel de model poate fi simulat în Overture pentru a vedea cum se comportă și interacționează FMU-urile. Odată ce se obține suficientă încredere în aceste modele inițiale, acestea pot fi exportate individual ca FMU și integrate într-o co-simulare.

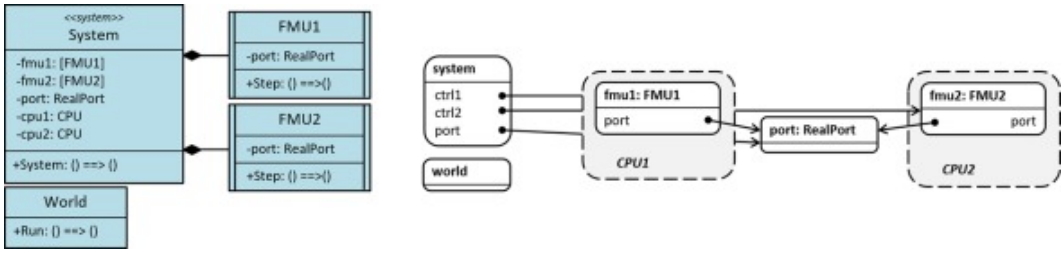

Figura 7.5. Diagrama de clasă care arată două clase de FMU simplificate create într-un singur proiect VDM-RT și o diagramă de obiecte care arată că acestea sunt instanțiate ca un test.

Plug-in-ul Overture FMI poate fi apoi utilizat pentru a exporta un FMU de la fiecare unitate de proiect individuală, acestea putând fi apoi combinate într-o co-simulare. Aceste FMU pot fi revizuite dacă sunt găsite probleme, apoi înlocuite cu modele de fidelitate mai mare. Modelele ar putea fi păstrate pentru utilizare ulterioară și ca rezervă în cazul unor probleme viitoare de integrare.

> Citiți capitolul complet