Raport tehnic şi ştiinţific final

privind implementarea proiectului PN-III-P2-2.1-PED-2016-0494
Acronim: ForVer

Răzvan Diaconescu

Decembrie 2018

1 Rezumat

Scopul principal al proiectului a fost dezvoltarea unui prototip, numit H, pentru specificarea şi verificarea formală a sistemelor reconfigurabile. Acestea sunt o clasă de sisteme software care admit moduri diferite de operare, numite configuraţii, şi pot comuta între aceste moduri pe parcursul execuţiei lor ca răspuns la evenimente interne sau externe. Logicile hibride apar ca o alegere naturală pentru formalizarea dinamicii acestor sisteme, pentru că sunt singurul tip de logici modale cu construcţii sintactice pentru aceasta. Modelul matematic care oferă fundamentul teoretic pentru H e bazat pe o metodă generică de a adăuga caracteristici ale logicii hibride peste o logică de bază arbitrară, printr-un proces numit *hibridizare*. Astfel, logica folosită pentru a exprima cerinţele la nivelul static al configuraţiilor poate fi aleasă în funcţie de problema pe care vrem sa o rezolvăm. Pentru a putea face demonstraţii în logica obţinută prin hibridizare, ne putem baza pe un proces generic de ridicare a unei translatări din logica de bază în logica de ordinul întâi la o translatare din logica hibridizată în logica de ordinul întâi. Cu translatarea astfel obţinută, putem verifica o conjectură in logica hibridizată prin translatarea problemei in logica de ordinul întâi şi rezolvarea ei acolo, cu un demonstrator automat de teoreme. Corectitudinea acestei metode este garantată de rezultate teoretice.

Am ales să implementăm prototipul H ca o extensie a Heterogeneous Tool Set (Hets), un pachet de tool-uri pentru parsarea, analiza statică şi managementul demonstraţiilor, folosit la specificarea si verificarea multi-formalism a sistemelor. Hets oferă suport pentru logica de ordinul întâi şi integrează demonstratoare automate de teoreme pentru logica de ordinul întâi. Mai mult, design-ul Hets permite adăugarea de noi logici prin instanţierea unei clase centrale. Implementarea prototipului nostru realizează procesul de hibridizare în întreaga lui generalitate: nu doar creăm noi instanţe ale acestei clase pentru anumite logici hibridizate, dar definim tipuri generice si implementăm în mod generic metodele care apar ca elemente ale unei logici, astfel încât noi logici hibridizate pot fi adăugate în sistem prin instanţierea acestor tipuri generice cu argumente care vin din logica de bază a hibridizării. Rezultatul acestui pas este că

obţinem un limbaj de specificare pentru logici hibridizate, ortogonal de logica folosită la nivelul static. Avantajul de a face asta este ca sintaxa folosită la nivelul dinamic este comună pentru toate logicile hibridizate, şi astfel utilizatorul nu trebuie să înveţe un nou limbaj de specificare pentru fiecare logică hibridizată. Mai mult, H oferă suport pentru două notaţii, una mai apropiată de limbajul natural si orientată spre dezvoltatorii de sisteme, şi cealaltă similară cu cea folosită în textele ştiinţifice şi orientată spre matematicieni, cu scopul de a mări baza de utilizatori. La un nivel similar de generalitate, am implementat ridicarea translatărilor în logica de ordinul întâi de la o logică de bază la una dintre hibridizările ei, astfel obţinând suport pentru demonstraţii bazat pe translatare pentru noul formalism de specificare. Teoria care rezultă prin translatare va avea în general dimensiuni mari, un grad ridicat de complexitate şi va folosi nume generate de simboluri introduse de-a lungul translatării, dar nu va fi vizibilă utilizatorului, ci va fi folosită doar pentru demonstraţii. Limbajul de specificare dezvoltat în proiect suportă de asemenea dezvoltarea modulară de specificaţii, astfel crescând lizibilitatea şi potenţialul de refolosire al specificaţiilor.

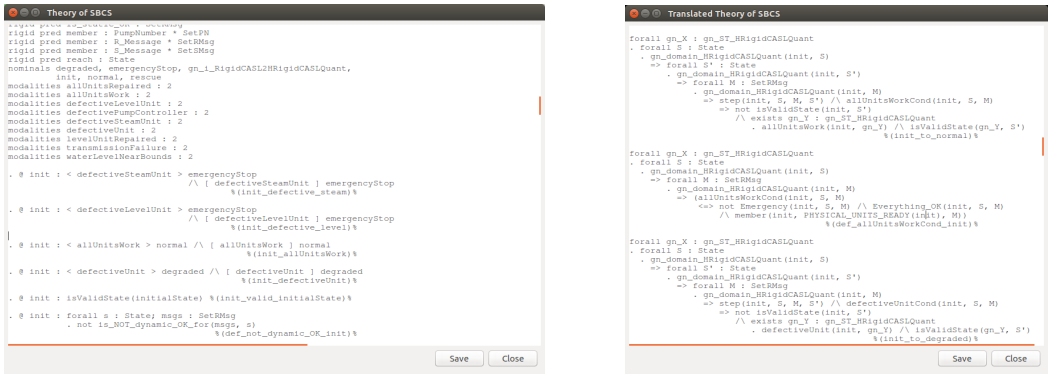

Figure 1: Sistemul de control al boilerului cu aburi în H.

H a fost testat pe un număr de exemple mici, disponibile ca o librărie de exemple de specificaţii, şi pe un studiu de caz de dimensiuni medii, specificarea şi verificarea formală a unui sistem de control pentru un boiler cu aburi. Această problemă a fost folosită ca un benchmark pentru a testa uşurinţa de a folosi in cazuri practice tehnicile majore de specificare şi verificare formală a programelor mari şi a sistemelor complexe. Figura 1 prezintă un fragment din teoria principală a sistemului de control pentru boilerul cu aburi, aşa cum e afişată in H, în stânga, şi un fragment al translătării acestei teorii în logica de ordinul întâi, în dreapta, unde sunt vizibile, în axiomele care apar, simbolurile generate de-a lungul translatării. Figura 2 arată rezultatul demonstrării unei leme în acest studiu de caz.

H a fost dezvoltat ca software open-source si este disponibil pentru descărcare pe pagina web a proiectului, împreună cu un ghid de utilizare şi cu documentul cu definiţia limbajului de specificare, care conţine detalii complete privind sintaxa si semantica sa.

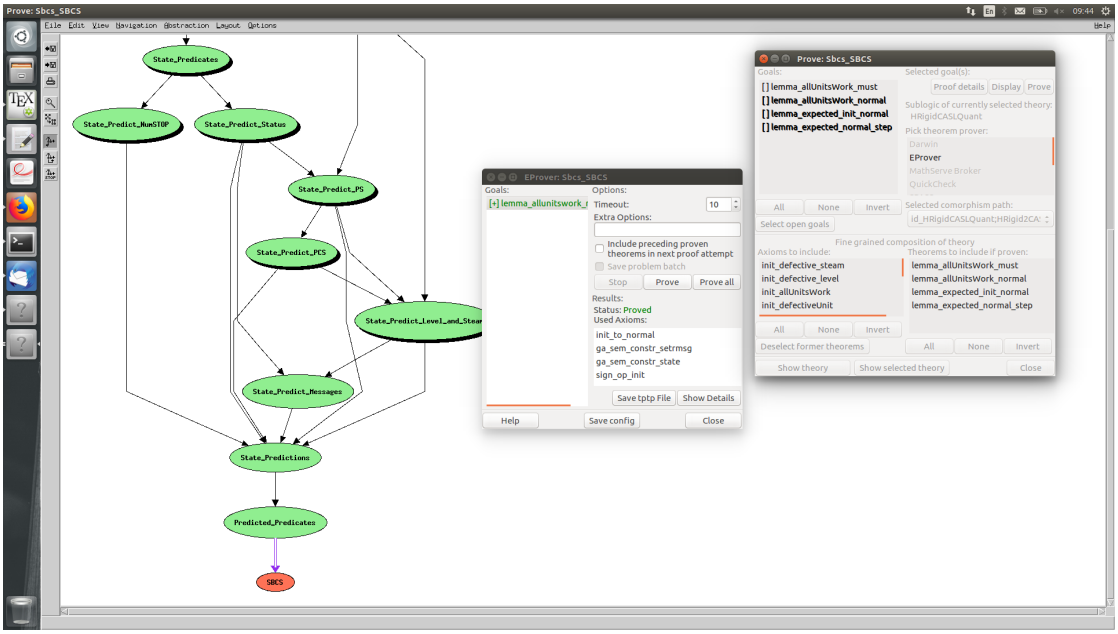

Figure 2: Sistemul H.

2 Implementarea sarcinilor din planul de lucru

Deoarece prima fază a proiectului a fost mult mai scurtă (17 august – 31 decembrie 2017), cea mai mare parte a rezultatelor au fost obţinute in faza a doua a proiectului. Din acest motiv, am inclus aici doar o scurtă prezentare a rezultatelor obţinute. Mai multe detalii pot fi obţinute consultând raportul ştiinţific al fazei a doua.

2.1 Work Area 1: Design şi metodologii

Work Package 1.1: Specificarea constrângerilor. Am introdus o sintaxa declarativă pentru specificarea parametrilor procesului de hibridizare: 1. logica de bază, 2. simbolurile permise în cuantificări şi 3. constrângerile asupra clasei de modele pentru logica hibridizată. Pentru constrângeri am introdus o gramatică fixată, care acoperă toate tipurile de constrângeri care apar în exemple.

Work Package 1.2: Limbaj generic de specificare pentru logici hibridizate. Am introdus construcţii sintactice pentru specificarea nominalilor şi a modalităţilor şi pentru a scrie formule într-o logică hibridizată. Limbajul este independent de cel folosit în logica de bază, care apare doar la nivelul formulelor de bază şi în partea de date a specificaţiei. Mai mult, am introdus suport pentru două sintaxe: una mai apropiată de limbajul natural si orientată spre dezvoltatorii de sisteme, şi cealaltă similară cu cea folosită în textele ştiinţifice şi orientată spre matematicieni.

Work Package 1.3: Metodologia de specificare. La nivelul specificaţiilor nestructurate, am introdus o separare clară între partea de date şi partea de configurare a sistemului, în care se specifică stările, tranziţiile şi proprietăţile sistemului folosind formule din logica hibridizată. Pentru modularizare folosim limbajul de specificare Distributed Ontology, Modeling and Specification Language (DOL) [3], un meta-limbaj pentru structurarea ontologiilor, specificaţiilor şi a modelelor MDE, independent de formalismul utilizat la nivelul specificaţiilor nestructurate, şi care poate fi folosit deci şi pentru logici hibridizate. DOL a fost adoptat recent ca un standard de Object Management Group şi este suportat de Hets.

Work Package 1.4: Metodologia de verificare. Pentru a obţine suport pentru demonstraţii pentru hibridizarea H a unei logici L, vom aplica procesul de ridicare a unei translatări din L în logica de ordinul întâi la o translatare din H în logica de ordinul întâi. Astfel, putem verifica dacă o conjectură în H este adevărată sau nu prin translatarea ei în logica de ordinul întâi şi verificarea ei cu ajutorul unuia din demonstratoarele automate de teoreme disponibile in Hets.

2.2 Work Area 2: Implementare

Sistemul H a fost implementat ca o extensie a Hets, care adaugă mai mult de 7000 de linii de cod Haskell la sursele Hets.

Work package 2.1: Parser pentru limbajul generic de specificare. Parserul pentru limbajul nostru de specificare a fost implementat ca o metoda Haskell care are ca argument metoda folosită pentru parsingul specificaţiilor logicii de bază a hibridizării. Ambele notaţii ale limbajului de specificare, atât cea pentru matematicieni cât şi cea pentru dezvoltatori, sunt suportate.

Work Package 2.2: Hibridizarea logicilor. Am implementat o metodă generică de a genera noi instanţe ale clasei de tipuri **Logic** din Hets pe baza definiţiilor declarative de hibridizări de logici. Aceste instanţe sunt generate sub formă de cod sursă Haskell, care este apoi compilat pentru a face noua logică hibridizată disponibilă pentru specificare în sistem.

Work Package 2.3: Hybridizarea translatărilor. În mod similar, o metodă generică analizează definiţia de hibridizare a unei translatări din logica de bază în logica de ordinul întâi şi generează cod Haskell continând o noua instanţă a clasei de tipuri **Comorphism** pentru translatări din Hets. Dupa o compilare a codului generat, noua translatare este disponibilă pentru demonstraţii prin translatare în logica de ordinul întâi în sistem.

Work Package 2.4: Suport pentru metodologia de specificare. Clasa de tipuri **Logic** conţine un număr de metode necesare în implementarea semanticii specificaţiilor structurate din DOL. Instanţele acestei clase de tipuri pe care le generăm pentru instituţii hibridizate conţin definiţii pentru aceste metode.

2.3 Work Area 3: Documentaţie şi exemple

Work Package 3.1: Ghid de utilizare. Ghidul de utilizare al H este disponibil prin intermediul paginii Web a proiectului, la adresa
`http://imar.ro/~diacon/PN-III-P2-2.1-PED-2016-0494.html`

Work Package 3.2: Librăria de exemple. Librăria de exemple este găzduită pe serverul Ontohub, la adresa `https://ontohub.org/forver`, şi poate fi descărcată şi ca un Git repository la adresa `git://ontohub.org/forver.git`.

Work Package 3.3: Studiul de caz. Studiul de caz prezintă specificarea şi verificarea formală unui sistem de control pentru un boiler cu aburi, un exemplu de dimensiuni medii care a fost folosit ca un benchmark pentru a testa uşurinţa de a folosi in cazuri practice tehnicile majore de specificare şi verificare formală a programelor mari şi a sistemelor complexe.
Specificaţia noastra ilustrează toate caracteristicile limbajului de specificare introdus în proiect, atât la nivelul logicii hibridizate folosite pentru a specifica sistemul, cât şi la nivelul specificaţiilor structurate şi a verificării. În formalizarea noastră, sistemul are 5 moduri si 9 evenimente, iar întreaga specificaţie are mai mult de 800 de linii. Studiul de caz este disponibil la `https://ontohub.org/forver/Sbcs.dol`.

2.4 Work Area 4: Diseminarea rezultatelor

Pagina Web a proiectului, care include un link la pagina Web a sistemului, este disponibilă la adresa `http://imar.ro/~diacon/PN-III-P2-2.1-PED-2016-0494.html`.

Două articole în conferinţe sunt momentan în pregătire. Un articol de descriere a sistemului (4 pagini), bazat în linii mari pe manualul de utilizare, va fi trimis la conferinţa CALCO (`https://www.coalg.org/calco-mfps-2019/`), în secţiunea de tool-uri. Astfel, prototipul dezvoltat în proiect va fi promovat în comunitatea academică din care fac parte cei doi membri ai proiectului. CALCO are loc la fiecare doi ani, iar ediţia următoare va fi in iunie 2019. Un articol mai lung care prezintă studiul de caz va fi trimis la 3rd World Congress on Formal Methods (`http://formalmethods2019.inesctec.pt/`) care va avea de asemenea loc în anul următor.

3 Direcţii de lucru în viitor

Codul sursă al H se află momentan sub review din partea dezvoltatorilor Hets, cu scopul de a fi integrat cu ramura principală de dezvoltare a Hets. Astfel, orice mod-

ificare majoră din Hets va trebui să nu afecteze funcţionalitatea H dezvoltată în acest proiect. Mai mult, această integrare aduce şi alte beneficii importante în termeni de uşurinţa utilizării sistemului. În primul rând, utilizatorii H îl vor putea folosi prin intermediul interfeţei Web a Hets, Ontohub, astfel având disponibil suport pentru repositories Git de specificaţii şi nu vor trebui să instaleze local sistemul. În al doilea rând, dezvoltatorii Hets plănuiesc sa creeze un mediu integrat de dezvoltare de tip IDE pentru Hets, bazat pe plug-in-uri pentru editorul Atom. Pe baza integrării, acest editor va putea fi folosit şi pentru dezvoltare în H.

4 Potenţial pentru exploatare comercială

Prototipul dezvoltat în acest proiect suportă specificarea sistemelor reconfigurabile, ale căror moduri de execuţie se pot schimba ca răspuns la diferite interacţiuni cu mediul. Sistemele safety-critical prezintă adesea un astfel de comportament. Un domeniu important de interes este cel al dispozitivelor medicale, unde importanţa cerinţelor de siguranţă este crucială. Logicile hibridizate au fost deja folosite pentru specificarea si verificarea formală a comportamentului dispozitivelor medicale, de exemplu [2] folosind o pompă de insulină cu infuzie continuă ca studiu de caz. De aceea, vom urmări să contactăm producătorii acestor dispozitive pentru a aplica metodologia noastră de specificare şi verificare în domeniul medical.

References

[1] Jean-Raymond Abrial, Egon Börger, and Hans Langmaack, editors. *Formal Methods for Industrial Applications, Specifying and Programming the Steam Boiler Control (the Book Grow out of a Dagstuhl Seminar, June 1995).*, London, UK, UK, 1996. Springer-Verlag.

[2] Alexandre Madeira, Renato Neves, Luís Soares Barbosa, and Manuel A. Martins. A method for rigorous design of reconfigurable systems. *Sci. Comput. Program.*, 132:50–76, 2016.

[3] Till Mossakowski, Mihai Codescu, Fabian Neuhaus, and Oliver Kutz. The distributed ontology, modeling and specification language - DOL. In Arnold Koslow and Arthur Buchsbaum, editors, *The Road to Universal Logic*, volume 2, pages 489–520. Birkhäuser, 2015.