WETENSCHAPPEN & BIO-INGENIEURSWETENSCHAPPEN

De Onderzoeksgroep

Software Languages Lab

nodigt U graag uit op de openbare verdediging van het proefschrift van

Kevin De Porre

ter behaling van de graad van Doctor in de Wetenschappen

Titel van het proefschrift:
When sequential code meets replicated data

Promotor:
Prof. dr. Elisa Gonzalez Boix (VUB)

De verdediging heeft plaats op

Maandag 19 december 2022 om 17u in aula D.2.01

Contacteer Kevin De Porre op kdeporre@vub.be indien u de presentatie wilt volgen via Microsoft Teams.

Samenstelling van de jury

Prof. dr. Bart de Boer (VUB, voorzitter)
Prof. dr. Bas Ketsman (VUB, secretaris)
Prof. dr. Nikos Deligiannis (VUB)
Prof. dr. Annette Bieniusa (Technical University of Kaiserslautern)
dr. Martin Kleppmann (Technische Universität München)

Curriculum vitae

Kevin De Porre behaalde de graad van MSc in de Computerwetenschappen in 2018 aan de VUB en ontving de BrEA prijs voor de computerwetenschappen voor zijn proefschrift. In datzelfde jaar startte hij een doctoraat aan het Software Language Lab (SOFT) gefinancierd door een SB-beurs van het Fonds Wetenschappelijk Onderzoek - Vlaanderen.

Zijn onderzoek richtte zich op programmeertalen en technieken die programmeurs helpen bij het ontwerpen, implementeren en verifiëren van highly available gedistribueerde systemen met behulp van data replicatie. Zijn onderzoek resulteerde in vier publicaties in internationale peer-reviewed tijdschriften en conferenties, en twee bijdragen in internationale peer-reviewed workshops.

Abstract van het doctoraatsonderzoek

Hedendaagse applicaties draaien veelal bovenop een geo-gedistribueerd systeem, zijnde een systeem dat gegevens repliceert over meerdere machines (replica's genaamd) die zich op strategische posities over de hele wereld bevinden. Dergelijke systemen zorgen doorgaans voor goede prestaties en hoge beschikbaarheid doordat replica's onafhankelijk kunnen worden aangepast. Replica's kunnen echter gelijktijdige updates in verschillende volgordes uitvoeren, wat tot conflicten kan leiden. Programmeurs moeten deze conflicten voorzien en oplossen, hetgeen zeer ingewikkeld is.

Om te vermijden dat programmeurs conflicten handmatig moeten oplossen, stelden onderzoekers gerepliceerde varianten voor sequentiële datatypes voor, genaamd Replicated Data Types (RDT's). RDT's lijken op traditionele datastructuren maar implementeren intern mechanismen om conflicten op te sporen en op te lossen. RDT's kunnen verschillende semantiek hebben afhankelijk hoe conflicten worden opgelost.

In dit proefschrift argumenteren we dat geavanceerde toepassingen op maat gemaakte RDT's vereisen die zijn afgestemd op de behoeften van de toepassing. Bestaande RDT's kunnen echter niet worden aangepast omdat conflictoplossing hardgecodeerd is. Dit dwingt programmeurs om hun eigen RDT's te bouwen of bestaande RDT's aan te passen met behulp van ad-hoc mechanismen voor conflictdetectie en -oplossing. Het is welbekend dat ad-hoc mechanismen foutgevoelig zijn en resulteren in fragiele systemen. Bovendien worden deze nieuwe of gewijzigde RDT's zelden geverifieerd vanwege de complexiteit van software verificatie, waardoor bugs waarschijnlijk onopgemerkt blijven.

Om de aangehaalde problemen aan te pakken, onderzoeken we in deze dissertatie programmeertaalabstracties en technieken voor de ontwikkeling van correcte RDT's. Dit leidde tot een tweeledige oplossing. Ten eerste stellen we een algemene aanpak voor om bestaande sequentiële datatypes te repliceren zodat men geen gespecialiseerde RDT's dienen te bouwen voor elke use case. Met onze aanpak kunnen programmeurs sequentiële datatypes uitbreiden met een *declaratieve specificatie* van de gewenste semantiek voor conflictoplossing, uitgedrukt door applicatie specifieke invarianten. We analyseren deze datatypes statisch om alle conflicten op te sporen en oplossingen te vinden die aansluiten bij de gewenste semantiek. Tijdens de programma uitvoering serialiseert ons nieuwe replicatieprotocol bewerkingen op een efficiënte manier zodat replica's naar dezelfde staat convergeren en toepassing specifieke invarianten behouden met minimale coördinatie. Om onze aanpak te valideren, hebben we een uitgebreid portfolio aan RDT's en verschillende applicaties gebouwd die gelijkaardige prestaties als state-of-the-art oplossingen vertonen.

Ten tweede stellen we VeriFx voor, een high-level programmeertaal om ad-hoc RDT's te implementeren en te verifiëren. Programmeurs implementeren nieuwe RDT's in VeriFx en krijgen automatisch een bewijs dat hun implementatie correct is of een tegenvoorbeeld indien dit niet het geval is. Geverifieerde RDT's kunnen worden vertaald naar mainstream talen om ze in een bestaand systeem te integreren. Om de effectiviteit van onze aanpak te demonstreren hebben we VeriFx gebruikt om een portfolio van meer dan 40 RDT's te implementeren en te verifiëren. Dit portfolio bevat welgekende RDT's uit de literatuur over Conflict-free Replicated Data Types (CRDT's) en Operational Transformation (OT) alsook commerciële databases.

Onze resultaten leiden tot twee belangrijke inzichten. Ten eerste is het mogelijk om efficiënte RDT's te bouwen met toepassing specifieke semantiek, zonder handmatig conflicten op te moeten lossen. Ten tweede kan de implementatie en verificatie van RDT's worden verenigd in een high-level taal zodat software-ingenieurs zonder diepgaande kennis van verificatie toch RDT's kunnen implementeren en automatisch verifiëren.