SCIENCES & BIOENGINEERING SCIENCES

The Research Group
Software Languages Lab

has the honor to invite you to the public defense of the PhD thesis of

Isaac Nyabisa Oteyo

to obtain the degree of Doctor of Sciences

Title of the PhD thesis:

DisCoPar-Kilimo: A Low-Code Development Environment Geared Towards Smart Agriculture Applications

Promotors:
Prof. Dr. Elisa Gonzalez Boix (VUB)
Prof. Dr. Wolfgang De Meuter (VUB)
Dr. Angel Luis Scull Pupo (VUB)

The defense will take place on **Friday, January 26, 2024, at 17h30 in Auditorium I.0.02.**

The defense can also be followed through a live stream: https://shorturl.at/cvX47

Members of the jury
Prof. Dr. Ann Nowé (VUB, chair)
Prof. Dr. Abdellah Touhafi (VUB, secretary)
Prof. Dr. Robert Hirschfeld (University of Potsdam, Germany)
Prof. Dr. Engineer Bainomugisha (Makerere University, Uganda)

Curriculum vitae

Isaac Nyabisa Oteyo received his master's degree in 2017 from Makerere University, Uganda. Afterwards, he started his PhD studies under the supervision of Prof. Dr. Elisa Gonzalez Boix, Prof. Dr. Wolfgang De Meuter and Dr. Angel Luis Scull Pupo in October 2017 at the Software Languages Lab (SOFT), Department of Computer Science, Vrije Universiteit Brussel. His PhD focused on low-code development environments (LCDEs) for smart agriculture applications. The research leading to his PhD dissertation has been presented and published at international and national workshops, conferences, and journals. He has two journal papers, three conference papers, and two workshop papers as first author. Lastly, he has one conference paper as second author. His PhD work was funded by VLIR-UOS IUC for global North and global South collaboration.

Abstract of the PhD research

Smart agriculture applications (SAAs) are used to improve farming activities in modern farms. Today, designing and implementing SAAs is difficult and costly. Typically, SAAs are systems that orchestrate distributed components deployed in microcontrollers, smartphones, and cloud services. It requires highly skilled engineers to implement SAAs as it entails carefully handling distribution to enable the different parts that compose the system to communicate. However, "skilled developers are hard to find". Moreover, the network infrastructure in rural areas is limited and unreliable resulting in inexistent or intermittent connections. This thesis aims to simplify the implementation of SAAs and study software techniques to enable domain experts to implement them.

Smart agriculture applications are often constructed using conventional programming languages that require software development knowledge which domain experts often do not have. Low-code development environments (LCDEs) have emerged as an alternative for users lacking technical expertise. The LCDEs offer visual programming environments with "ready-to-use" components, making software development more accessible to all technical skill levels. However, certain limitations hinder the widespread usage of LCDEs for the development of SAAs. First, the state-of-the-art LCDEs lack components for specifying edge computations on devices installed in the environment (e.g., crop fields and farms) and assume that sensors gather data and send it to the cloud or server. Second, there are no components to support coordinating the communication between the different parts that compose SAAs e.g., coordinating the communication between the edge and mobile components. Lastly, the output of the computations done on the devices installed in the environment needs to be sent to other components on the mobile phones or server over communication networks which can fail. The existing LCDEs have limited support for handling failures between the different components that compose SAAs when the networks become intermittent or unavailable.

In this dissertation, we first identify four categories of components aimed at facilitating the development of SAAs using a low-code development environment. The components encompass functionalities to observe and monitor prevailing environmental conditions, support computation on the devices installed in the environment i.e., computation at the edge, coordinate the communication between the different components that compose SAAs and lastly, effectively handle partial network failures. We prototype the four component categories by extending an existing low-code development environment, DisCoPar, with "ready-to-go" components geared for SAAs. The resulting low-code development environment, which we named DisCoPar-Kilimo, adopts a flow-based programming environment where applications are represented as interconnected nodes that transmit data to one another. The nodes represent software components that perform specific computational tasks in the application. The components are presented in a palette from which domain experts can pick and use them to construct software. The key contributions of this dissertation consist of a set of properties and readily deployable components for building SAAs. Technically, we also contribute DisCoPar-Kilimo, a domain-specific low-code development environment for implementing SAAs. To assess the effectiveness of DisCoPar-Kilimo, we employed a scenario-based approach and successfully implemented seven representative smart agriculture scenarios on it. Four of these scenarios were based on the identified properties while the remaining three were derived from existing literature and our experiences with agricultural extension workers (i.e., farmers) in Kenya. The implemented scenarios demonstrate how DisCoPar-Kilimo can be used to construct SAAs and its flexibility in implementing SAAs. From the implemented scenarios, DisCoPar-Kilimo can be used intuitively by domain experts as it hides away application development issues like memory management and coordinating communication, which can take considerable time to configure manually. In conclusion, DisCoPar-Kilimo is unique and can be considered the first of a kind in new and future-generation LCDEs for SAAs.