NATIONAL UNIVERSITY OF SINGAPORE

School of Computing

PH.D DEFENCE - PUBLIC SEMINAR

Title: **Comparative Studies, Formal Semantics and PVS Encoding of CSP#**

Speaker: Ms Shi Ling

Date/Time: 28 October 2014, Tuesday, 10:00 AM to 11:30 AM

Venue: Tutorial Room 9 (COM2 01-08)

Supervisor : Dr Dong Jin Song, Associate Professor, School of Computing

Abstract:

Concurrency becomes an important and necessary property of large and complex systems. Many concurrent systems feature various interactions between execution processes, which are often communications via synchronous/asynchronous message passing or through shared resources. The intricate execution nature and common mission-critical feature of concurrent systems demand rigorous modelling and analysis methods at the early system design stage.

Communicating Sequential Processes (CSP) is a well-known formal specification language to model and analyse concurrent systems. Considerable efforts have been made to extend CSP to support emerging system features like data aspects by integrating declarative specification languages like Z, although the resulting CSP extensions lack automated analysis support.

Recently, Communicating Sequential Programs (CSP#) has been proposed to integrate high-level CSP-like process operators with low-level program constructs on the shared variables. Although these CSP-like extensions support similar types of concurrent systems, there are subtle and substantial differences between them, not only modelling features, but also tool support and verifiability. Our first work is to conduct comprehensive comparisons between CSP# and CSPM (a noticeable CSP extension) from the perspectives of operational semantics and verification capabilities together with eight benchmark systems. These comparisons provide insights for users to select suitable languages/tools for various concurrent systems.

CSP# operational semantics has been defined and used in its PAT model checker. However, it is not compositional, and lacks the support of compositional verification. Our second work is to propose a compositional denotational semantics of CSP# using the Unifying Theories of Programming (UTP). Our denotational semantics blends communication events with state transitions containing shared variables, and captures all possible concurrency behaviours. It also considers the interference of

the environment to process behaviours. We further define a set of algebraic laws capturing the distinct features of CSP#.

Proving our defined algebraic laws is important as such proofs can validate the correctness of the CSP# denotational semantics, although manual proving is tedious and subtle mistakes can easily occur. Moreover, a high grade of automated verification can save much human effort. Therefore, our third work is to encode CSP# denotational semantics into the Prototype Verification System (PVS), an integrated framework for formal specification and verification. Our encoding not only checks the semantics consistency, but also builds up a theoretic foundation for mechanical verification of CSP# models.