Choreographing Cyber-Physical Distributed Control Systems for the Energy Sector

Energy Systems are facing a significant change in the way their management and control is conceived. With the introduction of distributed and renewable energy based resources, a shift to a more distributed operation paradigm is emerging, overturning the conventional top-down design and operation principles. This shift creates a demand for distributed control systems (DCS) to facilitate a more adaptive and efficient operation of power networks. One key challenge here is to ensure the required reliability of distributed control systems. Whereas proven strategies exist for reliable control for coordination of physical actions, with increasing distribution of such control, the reliability and degradation properties in response to communications issues become more important. We build on the notion of Quality Choreographies, a formal model for the development of failure-aware distributed systems, and discuss how quality choreographies respond to the needs presented by DCS. We demonstrate their applicability by modelling the Bully Algorithm, one of the de-facto election algorithms used in coordination of DCS.

Enforcing Availability in Failure-Aware Communicating Systems

Choreographic programming is a programming-language design approach that drives error-safe protocol development in distributed systems. Motivated by challenging scenarios in Cyber-Physical Systems (CPS), we study how choreographic programming can cater for dynamic infrastructures where the availability of components may change at runtime. We introduce the Global Quality Calculus (GCq), a process calculus featuring novel operators for multiparty, partial and collective communications; we provide a type discipline that controls how partial communications refer only to available components; and we show that well-typed choreographies enjoy progress.
Declarative interpretations of session-based concurrency

Session-based concurrency is a type-based approach to the analysis of communication-intensive systems. Correct behavior in these systems may be specified in an operational or declarative style: the former defines how interactions are structured; the latter defines governing conditions. In this paper, we investigate the relationship between operational and declarative models of session-based concurrency. We propose two interpretations of session π-calculus processes as declarative processes in linear concurrent constraint programming (lcc). They offer a basis on which both operational and declarative requirements can be specified and reasoned about. By coupling our interpretations with a type system for lcc, we obtain robust declarative encodings of π-calculus mobility.

Protocol-Based Verification of Message-Passing Parallel Programs

We present ParTypes, a type-based methodology for the verification of Message Passing Interface (MPI) programs written in the C programming language. The aim is to statically verify programs against protocol specifications, enforcing
properties such as fidelity and absence of deadlocks. We develop a protocol language based on a dependent type system for message-passing parallel programs, which includes various communication operators, such as point-to-point messages, broadcast, reduce, array scatter and gather. For the verification of a program against a given protocol, the protocol is first translated into a representation read by VCC, a software verifier for C. We successfully verified several MPI programs in a running time that is independent of the number of processes or other input parameters. This contrasts with alternative techniques, notably model checking and runtime verification, that suffer from the state-explosion problem or that otherwise depend on parameters to the program itself. We experimentally evaluated our approach against state-of-the-art tools for MPI to conclude that our approach offers a scalable solution.

*General information*
State: Published
Organisations: Department of Applied Mathematics and Computer Science , Language-Based Technology, Universidade de Lisboa, Imperial College London
Pages: 280-298
Publication date: 2015

*Host publication information*
Publisher: Association for Computing Machinery
ISBN (Print): 978-1-4503-3689-5
BFI conference series: Systems, Programming, Languages and Applications: Software for Humanity (5000252)
Main Research Area: Technical/natural sciences
Program verification, Parallel programming, MPI, Session types, Dependent types

*Foundations of Communication-Centred Programming: Calculi, Logics & Types*
Service and process-oriented systems are relatively new technologies that promise effective business processes and flexible and adaptable enterprise IT systems. The key factor in service systems is the ability to decompose a business process into a distributed system, where each participant implements parts of the functions in a business process, and interactions between participants are performed via message passing. In recent years, service- and process-oriented applications have rapidly become the norm for distributed enterprises and they have led to a number of new programming languages and standards, collectively referred to as communication-centred programming. Despite their adoption, these languages and standards for communication-centred programming are still young and unstable, and not grounded on a solid theoretical foundation. There are at least two significant dimensions when describing communication-centred programs: First, the global/local views used to describe interactions, and second, the imperative/declarative specifications styles used. With respect to views:

a global view considers the system as a whole, describing specifications as sequences of message exchanges among participants, and a local view describes the system as a concurrent composition of processes, implementing each participant in the system. While the global view is what is usually provided as specification, the local view is a necessary step towards a distributed implementation. On specification styles: If processes are defined imperatively, then the control flow is defined explicitly (e.g.: as a flow graph of interactions/commands). In a declarative approach processes are described as a collection of conditions (e.g.: logical formulae) they should fulfil in order to be considered correct. Until now, research in these two dimensions have evolved rather independently from each other. This dissertation collects works devoted to foundational studies in communication-centred programming. Specifically, the dissertation revolves around process calculi as the main analytical tool for service-oriented systems. Process calculi are formal languages conceived for the description and analysis of concurrent systems, providing a rigorous framework where distributed systems can be accurately analysed, by means of reasoning techniques to verify essential properties of a system. By means of process calculi, we provide formal relations between global and local views, and declarative/imperative specifications. This is achieved by extending previous works on the area with additional information in model specifications, like timing constraints and compensable behaviour. Finally, we provide process specifications with reasoning techniques (specification logics, type systems, simulation techniques) that allow one to verify the behaviour of a service specification with respect to trustworthy properties in the system.

*General information*
State: Published
Prizes:

**Otto Mønsteds Fund: Travel grant**
Hugo-Andrés López-Acosta (Recipient)
Department of Applied Mathematics and Computer Science, Language-Based Technology

**Description**
Travel grant to support the presentation of a paper at the 2015 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications, OOPSLA 2015, part of (SLASH) 2015. Pittsburgh, PA, USA, October 25-30, 2015

**Details**
Awarded date: 25 Oct 2015
Granting Organisations: Otto Mønsteds Fond
Prize: Prizes, scholarships, distinctions