ABSTRACT

Supporting Selective Formalism in CSP++ with Process-Specific Storage

Alicia Gumtie
University of Guelph, 2012

Advisor:
Dr. W. B. Gardner

Communicating Sequential Processes (CSP) is a formal language whose primary purpose is to model and verify concurrent systems. The CSP++ toolset was created to embody the concept of selective formalism by making machine-readable CSPm specifications both executable (through the automatic synthesis of C++ source) and extensible (by allowing the integration of C++ user-coded functions). However, these user-coded functions were limited by their inability to share data with each other, which meant that their application was constrained to solving simple problems in isolation. We extend CSP++ by providing user-coded functions in the same CSP process with safe access to a shared storage area, similar in concept and API to Pthreads' thread-local storage, enabling cooperation between them and granting them the ability to undertake more complex tasks without breaking the formalism of the underlying specification. This feature's utility is demonstrated in our line-following robot case study.

ABSTRACT

CELLPILOT: AN EXTENSION OF THE PILOT LIBRARY FOR CELL BROADBAND ENGINE PROCESSORS AND HETEROGENEOUS CLUSTERS

Natalie Girard
University of Guelph, 2012

Advisors:
Dr. W. Gardner, Dr. G. Grewal

The CellPilot library provides a uniform communication programming model, based on Pilot's process/channel approach, for clusters of Cell Broadband Engine processors. Pilot, a thin layer on top of the Message Passing Interface (MPI) library, allows processes to read/write messages on channels defined between pairs of processes on the cluster, but Pilot alone does not help a Cell programmer cope with the considerable complexities of intra-Cell communication.

With CellPilot, programmers still design software in terms of processes, but they can now be located on a Cell node's Power Processor Elements (PPEs), Synergistic Processing Elements (SPEs), or non-Cell node within a heterogeneous Cell cluster, and communication is accomplished via channels between process pairs. Programs are coded in terms of reading and writing on those channels, whereupon CellPilot transparently applies whichever communication mechanisms are required to transport the message, regardless of its endpoints. This gives the programmer a way to handle inter-process communication while avoiding low-level I/O operations and the use of multiple libraries.

ABSTRACT

MISE EN SCENE: A SCENARIO-BASED APPROACH TO GENERATING CSP SYSTEM TRACES

John Douglas Carter
University of Guelph, 2006

Advisor:
Professor W. B. Gardner

The “Requirements to Design to Code” (R2D2C) project of NASA’s Software Engineering Laboratory is based on inferring a formal specification expressed in Communicating Sequential Processes (CSP) from system requirements supplied in the form of CSP traces. The traces, in turn, are to be derived from scenarios, a user-friendly medium often used to describe the required behavior of computer systems under development. This work, called Mise en Scene, defines a new scenario medium (Scenario Notation Language, SNL) suitable for control-dominated systems, coupled with a two-stage process for automatic translation of scenarios to a new trace medium (Trace Notation Language, TNL) which encompasses CSP traces. An extensive survey of the “scenario” concept and scenario-based approaches to system engineering is also presented.