Citation for published version


Copyright & reuse
Content in the Kent Academic Repository is made available for research purposes. Unless otherwise stated all content is protected by copyright and in the absence of an open licence (eg Creative Commons), permissions for further reuse of content should be sought from the publisher, author or other copyright holder.

Versions of research
The version in the Kent Academic Repository may differ from the final published version. Users are advised to check http://kar.kent.ac.uk for the status of the paper. Users should always cite the published version of record.

Enquiries
For any further enquiries regarding the licence status of this document, please contact:
researchsupport@kent.ac.uk

If you believe this document infringes copyright then please contact the KAR admin team with the take-down information provided at http://kar.kent.ac.uk/contact.html
Abstract: This paper outlines a new language and run-time technology offering formalised design and efficient implementation for highly concurrent dynamic systems. It is based on a careful combination of ideas from Hoare’s CSP (giving compositional semantics, refinement and safety/liveness analysis) and Milner’s π-calculus (giving dynamic network construction and mobility). We have been experimenting with systems developing as layered networks of self-organising neighbourhood-aware communicating processes, with no need for advanced planning or centralised control. The work reported is part of our TUNA (‘Theories Undepinning Nanite Assemblies’) project, in partnership with colleagues from the Universities of York and Surry, which is investigating formal approaches to the capture of safe emergent behaviour in highly complex systems. A particular study modelling artificial blood platelets is described. The software technology scales to millions of processes per processor and distributes over common multiprocessor clusters.

Keywords: concurrency, modelling, emergent behaviour, safety, occam-pi, CSP, π-calculus
Communicating Complex Systems

Peter H. Welch, Frederick R.M. Barnes and Fiona A. Polack

Abstract—This paper outlines a new language and runtime technology offering formalised design and efficient implementation for highly concurrent dynamic systems. It is based on a careful combination of ideas from Hoare’s CSP (giving compositional semantics, refinement and safety/liveness analysis) and Milner’s π-calculus (giving dynamic network construction and mobility). We have been experimenting with systems developing as layered networks of self-organising neighbourhood-aware communicating processes, with no need for advanced planning or centralised control. The work reported is part of our TUNA (‘Theories Undepinning Nanite Assemblies’) project, in partnership with colleagues from the Universities of York and Surry, which is investigating formal approaches to the capture of safe emergent behaviour in highly complex systems. A particular study modelling artificial blood platelets is described. The software technology scales to millions of processes per processor and distributes over common multiprocessor clusters.

Index Terms—concurrency, modelling, emergent behaviour, safety, occam-pi, CSP, π-calculus

I. Introduction and Motivation

... to be completed ...

Introduction to the TUNA project and the complex systems we are simulating. Also why we’re using occam-π [1] to do it — i.e. to support self-organising dynamic systems that we can formally reason about (safety element); and why we would want to reason about them in the first place. Performance details (memory and run-time overheads) and why this enables massive systems to be realised that give a chance for interesting, complex and unplanned behaviour to emerge.

II. Overview of occam-pi

Software systems programmed in occam-pi are built as layered networks of communicating processes. Processes in occam-pi are self-contained, interacting with their external environment through communication channels and a few other synchronisation mechanisms (all built from CSP events). Combined with strict parallel-usage and alias checks performed by the compiler, this gives occam-π programs strong safety guarantees — that can be established formally using Hoare’s CSP [2].

The following subsections provide an overview of the different occam-π mechanisms relevant to this work. The language encompasses much more, however, and we give some hints ...

A. Processes, channels and networks

... to be completed ...

B. Mobile channels, processes and barriers

... to be completed ...

C. Dynamic process creation

... to be completed ...

III. Simulating Complex Systems

... to be completed ...

How we use the occam-π mechanisms to build simulations – the model of self-contained neighbourhood-aware autonomous agents that roam around a virtual world we create for them. The facilities in occam-π allow us to do interesting things, like changing the ‘world’ dynamically, including lazy simulation [3]. How we use mobile barriers to time-step the simulation, as well as providing fairly cheap support for visualisation (phased barriers) [4]. Elaborate on how directly such design mirrors the low-level layers of processes (e.g. platelets, chemicals, physical space) in nature and their low-level interactions ... with benefits on the reality of the modelling and its efficiency (processor activity only to support active processes and their immediate environment).

Then talk about the blood-clot simulation and/or implementation of Steve’s CSP models.

IV. Formal Design and Analysis

... to be completed ...

Something about basic processes, channels and networks. Some integrators in figure 1. Maybe this can be assumed?

Fig. 1. Serial and parallel integrators

B. Mobile channels, processes and barriers

... to be completed ...

Something about mobile channels (for dynamic construction of process networks), mobile processes (for distributing computation) and mobile barriers (for N-way synchronisation).

C. Dynamic process creation

... to be completed ...

Something about FORKING and FORK (for dynamic growing and shrinking of systems).

P.H. Welch and F.R.M. Barnes are members of the Computing Laboratory, University of Kent, Canterbury, Kent, CT2 7NF, England. email: {phw,frmb}@kent.ac.uk

F.A. Polack is a member of the Department of Computer Science, University of York, Yorkshire, YO10 5DD, England. email: fiona@cs.york.ac.uk
these are turned into executable occam-pi (or JCSP) systems.

Something on the CSP models we have for mobile barriers and channels?

V. Conclusions and Future Work

... to be completed ...

What we’ve done and where we’re going.

References


