Communicating Process Architectures 2013
P.H. Welch et al. (Eds.)
Open Channel Publishing Ltd., 2013

Mutually Assured Destruction (or the Joy of Sync)

Fringe Presentation

Peter H. WELCH [a], Jan Bækgaard PEDERSEN [b] and Frederick R.M. BARNES [a]
[a] *School of Computing, University of Kent, UK*
[b] *Department of Computer Science, University of Nevada Las Vegas, NV, USA*
P.H.Welch@kent.ac.uk, matt.pedersen@unlv.edu, F.R.M.Barnes@kent.ac.uk

Abstract. In contrast to the *Client-Server* pattern, *Mutually Assured Destruction (MAD)* allows the safe opening of communication by either of two processes with the other. Should both processes commit to opening conversation together, both are immediately aware of this and can take appropriate action – there is no deadlock, even though the communications are synchronous. A common need for this pattern is in real-time control systems (e.g. robotics), artificial intelligence, e-commerce, model checking and elsewhere. A typical scenario is when two processes are given a problem to solve; we are satisfied with a solution to either one of them; whichever process solves its problem first kills the other and makes a report; the one that is killed also reports that fact. In this case, the opening communication between the two processes is the only communication (a *kill* signal). If both solve their problem around the same time and try to kill each other, *MAD* is the desired outcome (along with making their reports). A simple and safe solution (verified with FDR) for this pattern with occam *synchronous* communications is presented. This is compared with solutions via *asynchronous* communications. Although these avoid the potential deadlock that a naive strategy with synchronous communications would present, they leave a mess that has to be tidied up and this tidying adds complexity and run-time cost. The *Joy of Sync* arises from the extra semantic information provided by synchronous communications – that if a message has been sent, the receiving process has taken it. With this knowledge comes power and with that power comes the ability to implement higher level synchronisation patterns (such as *MAD* and *Non-blocking Syncs*) in ways that are simple, verifyably safe and impose very low overheads.

Keywords. mutually assured destruction, non-blocking synchronisation, occam-pi, concurrency, synchronisation, simplicity, verification, performance