Correctness of migration in a subset of Obliq

Hans Hüttel Josva Kleist
Uwe Nestmann
Department of Computer Science, Fredrik Bajers Vej 7E
DK-9220 Aalborg Ø, Denmark

September 1999

A particular style of programming mobile objects was proposed by Cardelli in the lexically-scoped distributed programming language Obliq. Lexical scoping in distributed settings facilitates the analysis of programs as the binding of variables is only determined by their location in the program text, and not by the execution site.

In Obliq, immutable values can be copied from site to site, but mutable values are stationary – an object that migrates does not actually move. Instead, a *clone* of the object is created at the target site. The original object becomes an *alias* for the new object.

Our aim is to describe this style of migration formally for a subset of Obliq that we call Øjeblik, and to prove it correct.

It turns out to be it essential for the safety of migration to provide a proper model of aliasing. One must describe the conditions under which the available operations may be accepted at the intended point of action. This is crucial for aliasing models since an operation that has to be *self-inflicted* can only be carried out if it addresses the same object in which it itself is active. One must find an aliasing model with a balance between the right amount of self-infliction control over incoming requests and the right amount of uncontrolled forwarding that allows an alias node to behave unobservable to its external clients. We suggest a *serialized forwarder* model, which differs from both Obliq's implementation and the existing formal semantics given by Talcott. Moreover, we show that any aliasing model is doomed to fail in the case of self-inflicted migration.

We also give an overview of the ideas of the proof of correctness of (non-self-inflicted) migration. We prove correctness via an encoding into a π-calculus with label-selective names. In our proof, we take advantage of the many tools that the π-calculus theory provides. We formalize the notion of self-infliction via the concept of keys, i.e. π-calculus names that can only be used for matching. No other names may be used for matching, so the sometimes tricky handling of matching in π-calculus is rather harmless on our setting.