Retrenchment

R. Banach[1] and M. Poppleton[1,2]

[1] Computer Science Dept., Manchester University, Manchester, M13 9PL, U.K.
[2] School of Mathl. and Inf. Sciences, Coventry University, Coventry, CV1 5FB, U.K.
banach@cs.man.ac.uk , m.r.poppleton@coventry.ac.uk

It has been noticed for some time, that when refinement is used as the sole means of progressing from an abstract model to a concrete one, then certain difficulties plague the development process due to the unforgiving nature of the usual refinement proof obligations. In its familiar downward simulation setting, refinement implies two things. Firstly that whenever the abstract model is able to make a step, the concrete model must also be able to make some step. And secondly that whenever the concrete model actually makes a step, there must be an abstract step that simulates it, in order to preserve the user's fiction, that it is the abstract model that is doing the work. The abstract model says *when* a step must be possible, while the concrete model dictates *how* the outcome may appear. This close link may be counterproductive in certain situations.

Consider natural number arithmetic. Suppose the abstract model contains some numerical state variables, modelled using Peano naturals. The concrete model must reflect the finite nature of real resources, so that concrete variables are finite naturals. Now there is no sensible refinement relation between these models. For consider what happens when the values of some variables are at their bounds. At the abstract level there is no problem as we can always do a calculation. However the concrete model will throw some exception because of the finite bounds. The POs of refinement require that a concrete calculation takes place (because the abstract level can perform a step), but even if there is a concrete calculation, its result will not correspond to the abstract result (by the concrete finiteness assumption).

Refinement works fine in textbook examples, small enough that eg. awkward limits can be captured at the abstract level without pain. In industrial scale situations though, the complexity of the resulting descriptions diminishes their usefulness, and makes refinement applicable close to code only — the bulk of the development effort must remain informal. Retrenchment avoids these problems by loosening the tight correspondence between abstract and concrete enforced by the refinement POs. As well as the usual initialisation and internal consistency POs, retrenchment demands:

$$G(u,v) \wedge P(i,j,u,v) \wedge Op_C(v,j,v',p) \Rightarrow \\ \exists u', o \bullet Op_A(u,i,u',o) \wedge (G(u',v') \vee C(u',v',o,p,\ldots))$$

where i, j are abstract/concrete inputs, u, v are state values, o, p are outputs, Op_A, Op_C are operations, G is the retrieve relation, and P and C are the within and concedes relations. P strengthens G, while C weakens G; '$\ldots$' allows before values to be refered to. (In total correctness, concrete termination is additionally assumed, and abstract termination is derived.) Arbitrary I/O and state mixing

J. Wing, J. Woodcock, J. Davies (Eds.): FM'99, Vol. II, LNCS 1709, pp. 1864–1865, 1999.

between levels facilitates specification evolution. P limits contact between levels, allowing maximum simplicity in the abstract level. C allows exceptions and non-refinement properties to be captured. The added flexibility allows radically different models to be related, eg. a continuous abstract model and a discrete concrete model, as is needed in the development of embedded systems, which usually need to interface to continuous physical models. Retrenchment allows formal techniques and mechanical checkability, to migrate higher into the development process than refinement alone reaches. Contradicting previous properties via C, allows high degrees of complexity to be introduced gradually, stepwise.

Banach R., Poppleton M.; Retrenchment: An Engineering Variation on Refinement. *in*: Proc. B-98, Bert (ed.), Springer, 1998, 129-147, LNCS **1393**.
Also: UMCS Tech. Rep. UMCS-99-3-2, `http://www.cs.man.ac.uk/cstechrep`
See also: `http://www.cs.man.ac.uk/~banach/Recent. publications.html`