1 Dimensions and programming

The idea of using dimension as a error-checking mechanism for programming has been suggested by a number of people during the last forty years. Most writing on the subject follows the lead of [KDBL78] and claims that the earliest mention of the idea was [?]. The claim appears to be that dimension should provide an augmentation to the existing type-systems and that such provision would provide an extra layer of error-protection that a compiler could in theory be built to check.

[?] and [?] discuss classic software problems in safety critical systems caused by confusion about whether values were imperial or metric and argue that dimensionality in the programming language would have had caught such errors. It is for these reasons that [?] rejects the idea of run-time checking of the consistancy of dimension.

Although they might have use in formal verification of software; I would like to draw attention to a possible use in making source code more readable. This use is also pointed out in [KDBL78] who recognise that attaching units to variables can provide "increased readablity".

2 Dimensionality in observables in EM

I think that the tkeden tool would be an ideal environment to explore the power of programming with dimension. Firstly, the discipline of Empirical Modelling makes great use of observation as a concept. This leads to discussions of a model's observables rather than a program's variables.

Observation relates the visible state of a model with experience. By this we mean that observables convey a meaning richer than a value – which can be the result of an unfinished computation. Observables directly relate to the referent of a model. That is, observables in a model will have a context that is more than just a value. Dimension units might be found to provide some of that context and help enhance tkeden's support for observable-oriented modelling.

References

[KDBL78] Michael Karr and III David B. Loveman. Incorporation of units into programming languages. *Communications of the ACM*, 21(5):385–391, 1978.