Oftentimes software systems have trouble decoupling the different domains that they’re built up from, causing software teams to be responsible for multiple layers of the software stack at once.

The Modular Language Stacking paradigm is described and proposed as a solution to domain contamination. The benefits of large scale Modular Language Stacking are hypothesised, and seem to have far reaching positive consequences.

The extremely high level Functional Reactive Programming paradigm is used as a vehicle to understand how a totally pure domain specific language may be developed and how should the description of Modular Language Stacking be refined. From this, the intuition is gathered that extensible, deeply embedded domain specific languages are required. Furthermore the extensible deep EDSLs are to be implemented in terms of other extensible deep EDSLs, achieved through compilation processes on the abstract syntax trees of the EDSLs.

Haskell literature on EDSLs is surveyed to gauge how well we are situated towards actualising Modular Language Stacking. The rich literature orients us and gives us a clear perspective for the next steps.