Claytronics is a project at Carnegie Mellon University to develop programmable matter – bringing the power of programming to physical matter. A Claytronics system consists of millions of tiny computing units called catoms. Each catom is capable of executing code, sensing and communicating with nearby catoms, and moving around its neighbors subject to the laws of physics. The result is an ensemble of particles which can change their physical properties under program control.

Surprisingly, the main challenge to realizing Claytronics is not the underlying hardware, but the programming methodology. Providing effective methods for programming an ensemble of millions of units so that they can reliably, even provably, work together to solve a common task is a significant challenge. We have developed two programming languages, LDP and Meld, with which we could implement some simple, yet useful, behaviors. Each language excelled at some tasks, but not on others. Furthermore, it is unknown whether either language or even their underlying programming styles will scale to large programs.

In this work, we investigate the potential of MSR 3, a programming paradigm combining multiset rewriting, logic and process algebra, as an effective basis for programming Claytronics. MSR 3 natively provides support for concurrency, synchronization, non-determinism, non-monotonicity, and atomicity. It has been used with great success in areas as security protocols and biomolecular systems. MSR 3 appears to extend the computing paradigms of both LDP and Meld. Our main objective is to customize MSR 3 for Claytronics, to support a variety of abstraction levels (from modeling the physical environment to programming meta-modules).

We believe that this will reap rewards not just for programming Claytronics, but will have a direct impact on understanding how best to program all large ensembles, sensor networks, internet protocol routers, autonomous vehicles, power system management, etc.