```
mcdp {
    # We need to fly for this duration
    provides endurance [s]
    # While carrying this payload
    # (in addition to battery)
    provides extra_payload [kg]
    # And providing this extra power
    provides extra_power [W]

    # Sub-design problem: choose the battery
    sub battery = dp {

        provides capacity [J]
        requires mass [kg]

    implemented-by load BatteryDP
    }
    # Sub-design problem: actuation
    sub actuation = abstract mcdp {
        # actuators need to provide this lift
        provides lift [N]
        # and will require power
        requires power [W]
        # simple model: quadratic
        c = 0.002 W/N^2
        power >= lift * lift * c
    }
    # Co-design constraint: battery must be large enough
    power = actuation.power + extra_power
    energy = power * endurance
    battery.capacity >= energy

    # Co-design constraint: actuators must be powerful enough
    gravity = 9.81 m/s^2
    weight = (battery.mass + extra_payload) * gravity
    actuation.lift >= weight

    # suppose we want to optimize for size of the battery
    requires mass for battery
}
```