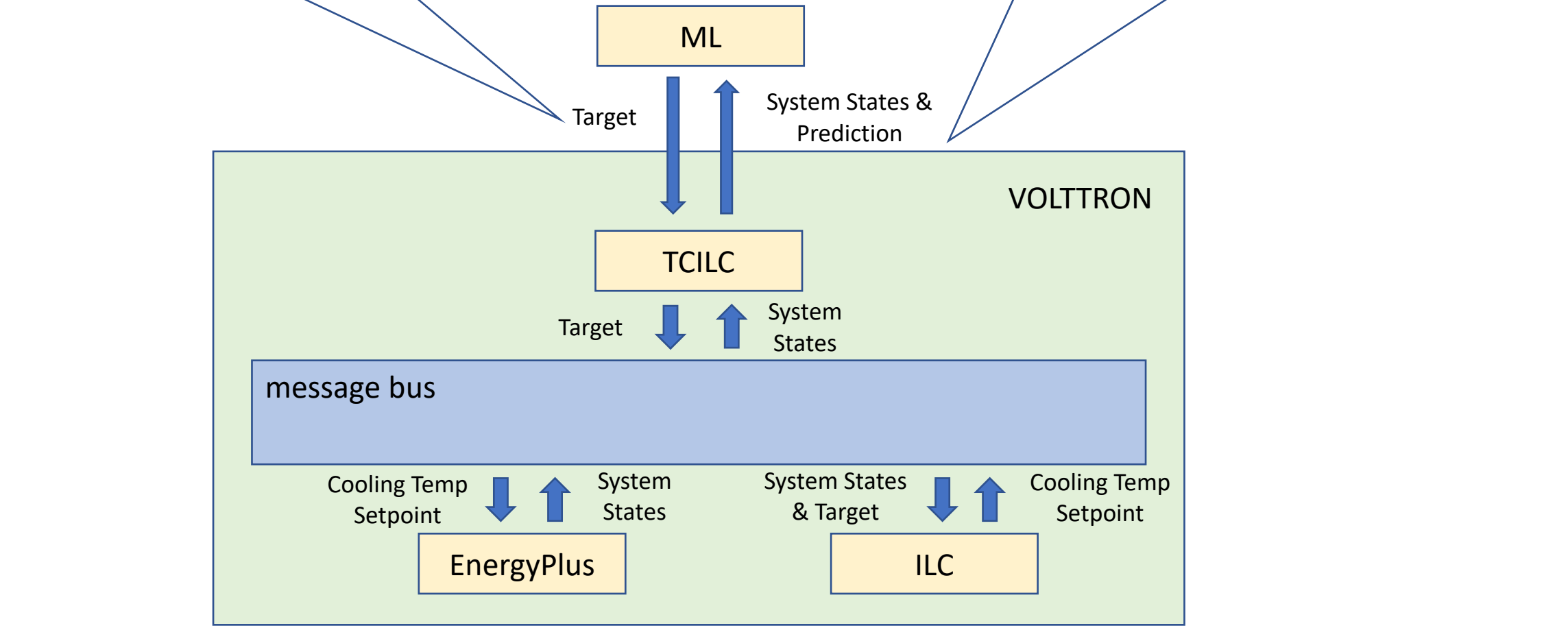
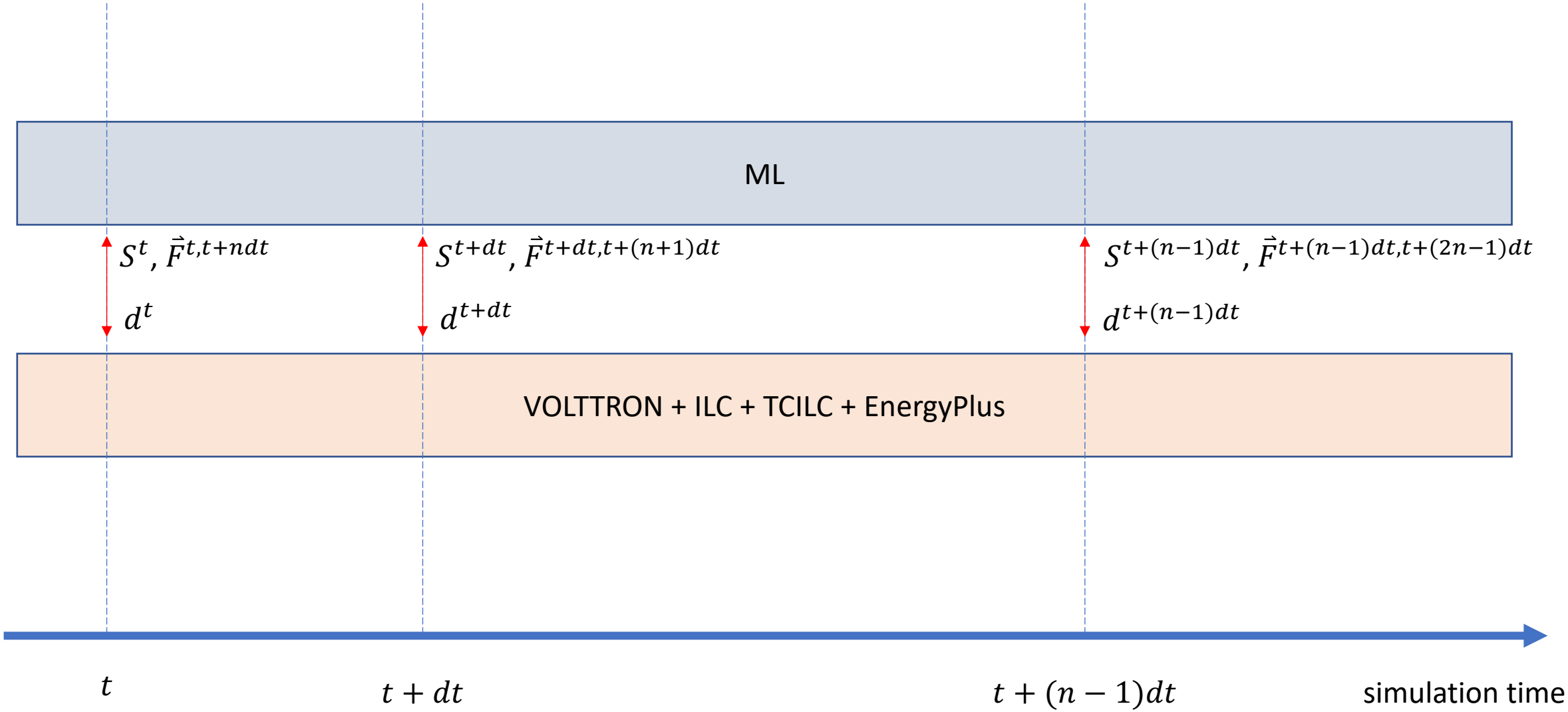
Target (denoted by d^t): the value to which the average building power, over a period from t to $t + dt$, should be kept;

System States (denoted by $\boldsymbol{S^t}$): a vector which includes the zone temperature at t, the zone cooling temperature setpoint at t, and the outdoor temperature at t;
Prediction (denoted by $\vec{\boldsymbol{F}}^{t,t+ndt}$): a vector which includes the predicted outdoor temperature over the period from t to $t + ndt$ and the predicted electricity price over the period from t to $t + ndt$




ML
S^t, $\vec{F}^{t,t+ndt}$
d^t
S^{t+dt}, $\vec{F}^{t+dt,t+(n+1)dt}$
d^{t+dt}
$S^{t+(n-1)dt}$, $\vec{F}^{t+(n-1)dt,t+(2n-1)dt}$
$d^{t+(n-1)dt}$
VOLTTRON + ILC + TCILC + EnergyPlus
t
$t + dt$
$t + (n-1)dt$
simulation time

```
def ML_main_function(u,y):
    '''
        Inputs
        ----------
        u : dict
          {
            outT_arr (float, length:n): [...],      Outdoor temperature for the future n steps
            Tset_arr (float, length:m): [...],      Current zone cooling temperature setpoint for m zones
            T_arr (float, length:m): [...],         Current zone temperature setpoint for m zones
            outT (float, scalar):.                  Current outdoor temperature
          }
        Returns
        -------
        y : dict
          {
            base_power_arr (float, length:n): [...],   Baseline power (zone temperature setpoints are kept unchanged) for the future n steps
            power_upper_arr (float, length:n): [...],  Power flexibility upper bounds for the future n steps
            power_lower_arr (float, length:n): [...]   Power flexibility lower bounds for the future n steps
          }
     '''
```

```
def ML_postprocess(u,y):
    '''
        Inputs
        -------
        u : dict
            {
                base_power_arr (float, length:n): [...],
                power_upper_arr (float, length:n): [...],
                power_lower_arr (float, length:n): [...],
                price_arr (float, length:n) : [...]
            }
        Returns
        -------
        y : dict
            {
                target_power_arr (float, length:n): [...]
            }
    '''
```

Baseline power (zone temperature setpoints are kept unchanged) for the future n steps

Power flexibility upper bounds for the future n steps

Power flexibility lower bounds for the future n steps

Prices for the future n steps

Targets for the future n steps