```
import dates_utility as utility
from models_base import ModelsBase
from state_vector_numpy import StateVectorNumpy as StateVector
from state_matrix_numpy import StateMatrixNumpy as StateMatrix
from state_matrix_sp_scipy import StateMatrixSpSciPy as SparseStateMatrix

class MyModel(ModelsBase):
    _model_name = "MyModel"
    _default_model_configs = dict(model_name=_model_name)

    def __init__(self, model_configs=None, output_configs=None):
        """ Constructor; MyModel class implementation. """
        # Aggregate passed configurations with default configurations
        model_configs = utility.aggregate_configurations(model_configs, DummyModel._default_model_configs)
        self.model_configs = utility.aggregate_configurations(model_configs, ModelsBase._default_model_configs)
        self._output_configs = utility.aggregate_configurations(output_configs, ModelsBase._default_output_configs)

    def state_vector(self):
        """ initialize an empty state vector """
        return StateVector(np.zeros(self.state_size()))

    def state_matrix(self, create_sparse=False):
        """ initialize an dense/sparse empty state matrix """
        state_size = self.state_size()
        if create_sparse:
            return SparseStateMatrix(sparse.lil_matrix((state_size, state_size)))
        else:
            return StateMatrix(np.zeros((state_size, state_size)))
```