1. **What is Component-based modelling?**

Component-based modelling focuses on creating architectures from components that are loosely coupled and can be reused. Component-based modelling supports the effective development and communication of robust systems, and can have multiple advantages (MathWorks, 2024) such as:

- Decreased complexity via each component being responsible for a smaller, isolated problem
- Unit testing compatibility via reduced retesting
- Increased reusability via “plug-and-play” components across multiple projects,

2. **Upon what do component-based modelling frameworks depend?**

Component-based modelling frameworks depend on an effective language, connectivity and coordination strategy that enables information flows between the components (Palomar et al., 2016).

3. **Within the context of the work presented in this paper, what is *PyNSim*?**

PyNSim is a Python library that implements a framework compatible with generating resource network simulations using a component-based architecture. Particularly, this component-based architecture enables multi-agent simulations. It has a Simulation class that the user uses to define simulation parameters, Network object that is connected to Node, Institution and Link objects, and an Engine class that drives the simulations.

4. **How does PyNSim achieve its goal when using object-oriented Python programming?**

It has the required components effectively modularised, where the engines interact with the network. The network *components* module contains *Component, Container, Network, Node, Link* and *Institution* classes. To achieve what is needed, PyNSim implements inheritance. The *Component* class is the top-level object from which the *Container, Node* and *Link* classes inherit. The *Network* and *Institution* class inherit from *Container*. The *engine* module performs the necessary calculations, and the *simulation* module enables to user to perform the actual simulation of the system with the required parameters. Although the nature of the components being complex, the effective use of abstraction has hidden the inner workings and can enable rapid prototyping and simulations among network practitioners who are relatively new to scripting (Knox et al., 2018). Encapsulation enables the storing of necessary input data and results of the simulations, and reusability. Particularly, the network component stores the necessary data as the common means of communication between engines.

References:

Knox, S., Meier, P., Yoon, J. & Harou, J.J. (2018) A python framework for multi-agent simulation of networked resource systems. *Environmental modelling & software*, 103: 16-28.

Mathworks (2024) Component-Based Modeling Guidelines. MathWorks. Available from: https://www.mathworks.com/help/simulink/ug/component-based-modeling-guidelines.html [Accessed 1 June 2024]

Palomar, E., Chen, X., Liu, Z., Maharjan, S. & Bowen, J. (2016) Component-based modelling for scalable smart city systems interoperability: A case study on integrating energy demand response systems. *Sensors*, 16(11): 1810.