**Assignment 1 – SRE**

Name - Sanoj

Roll NO: 2K21/SE/159

Question 1: Explain Aspect oriented requirement engineering. Why do we need it?

Aspect-Oriented Requirements Engineering (AORE) is a way to make computer programs better. It helps us deal with things in programs that are tricky and affect many parts of the program.

**Key Concepts of AORE:**

1. **Concerns**: In AORE, concerns are the primary units of analysis. Concerns represent different aspects of the software system's functionality, and they can be categorized as either core concerns (essential functionality) or cross-cutting concerns (issues that affect multiple parts of the system).
2. **Join Points**: Join points are specific points in the software where a cross-cutting concern might be applied. For example, in a banking application, a join point could be the point where a user logs in.
3. **Advice**: Advice represents the behavior associated with a cross-cutting concern. In AOP, advice can be added to specific join points to modify the system's behavior without changing the core functionality.

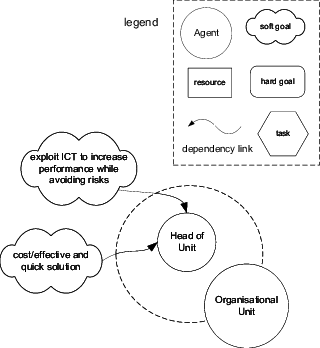
**Why Do We Need AORE:**

1. **Modularity**: Traditional requirements engineering often results in intertwined, tangled requirements that make it challenging to understand and manage cross-cutting concerns. AORE promotes modularity by isolating these concerns, making it easier to reason about and change them independently.
2. **Maintainability**: As software evolves, changes to cross-cutting concerns can impact multiple parts of the system. AORE allows for the encapsulation of these concerns, reducing the risk of introducing errors when modifying the system.
3. **Reusability**: By explicitly identifying and modularizing cross-cutting concerns in the requirements phase, AORE promotes the reuse of these concerns across different projects or within the same project.
4. **Traceability**: AORE enhances traceability between the requirements and their implementation. Developers can more easily see how requirements map to aspects, making it easier to verify that the system meets its intended functionality.
5. **Conflict** **Resolution**: AORE helps identify conflicts or contradictions between requirements early in the development process, reducing the likelihood of costly changes later in the project.

In summary, **Aspect-Oriented Requirements Engineering (AORE)** is a methodology that extends the principles of **Aspect-Oriented Programming** to the requirements engineering phase. It helps manage cross-cutting concerns more effectively, improving modularity, maintainability, reusability, traceability, and conflict resolution in software development projects.

Question 2: Explain agent-based requirement engineering. What are its drawbacks?

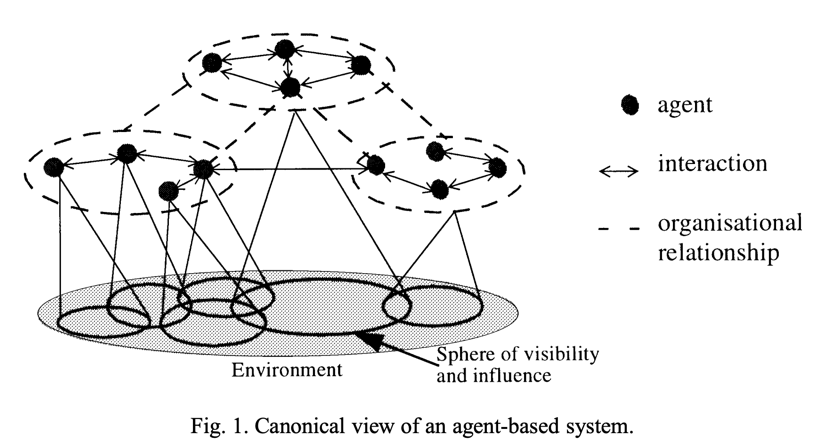
**Agent-Based Requirement Engineering** is an approach used in software development to model and analyze complex systems by focusing on the interactions between autonomous agents. In this context, "agents" are entities or components that have their objectives, knowledge, and behaviors and can act independently to achieve their goals within the software system. These agents can represent various elements, such as users, subsystems, or even hardware components.



**Agent Based Requirement Engineering – The Initial Model**

**Key Concepts in Agent-Based Requirement Engineering:**

* **Agents**: Agents are the central entities in this approach. They can be human users, software components, or hardware devices, and they are characterized by their goals, knowledge, and actions.
* **Interactions**: The focus is on modeling the interactions or communications between these agents. These interactions can be through messages, data exchanges, or even collaborative decision-making.
* **Autonomy**: Agents have a degree of autonomy, meaning they can make decisions independently to achieve their goals. They don't always rely on a central controller to tell them what to do.
* **Emergent Behavior**: The behavior of the entire system emerges from the interactions and autonomous actions of individual agents. This emergent behavior can be analyzed to understand the system's overall performance.



**Drawbacks of Agent-Based Requirement Engineering:**

* **Complexity**: Agent-based modeling can be quite complex, especially for systems with many interacting agents. Managing and understanding these interactions can become challenging.
* **Validation and Verification**: Verifying and validating agent-based models can be difficult. Ensuring that the model accurately represents real-world behavior can be a significant challenge.
* **Resource Intensive**: Developing agent-based models can be resource-intensive, both in terms of time and computational power. This can make it impractical for certain projects.
* **Lack of Standardization**: Unlike some other software engineering methodologies, there is no widely accepted standard for agent-based modeling. This lack of standardization can make it harder to communicate and collaborate with others in the field.
* **Steep Learning Curve**: Understanding and applying agent-based modeling techniques may require a steep learning curve for those not familiar with the approach. Training and expertise in this methodology are often necessary.
* **Overhead**: In some cases, introducing agents and their interactions can introduce unnecessary overhead, leading to increased complexity without significant benefits.

In summary, agent-based requirement engineering is an approach that focuses on modeling complex systems by representing them as interactions between autonomous agents. While it has its advantages in capturing emergent behavior and real-world dynamics, it also comes with drawbacks such as complexity, challenges in validation, and resource requirements that need to be carefully considered when choosing this approach for a software development project.

**References:**

1. An agent-based requirements engineering framework for complex socio-technical systems by [**Paolo Bresciani**](https://www.researchgate.net/profile/Paolo-Bresciani-4?_tp=eyJjb250ZXh0Ijp7ImZpcnN0UGFnZSI6InB1YmxpY2F0aW9uIiwicGFnZSI6InB1YmxpY2F0aW9uIn19) and [**Paolo Donzelli**](https://www.researchgate.net/profile/Paolo-Donzelli?_tp=eyJjb250ZXh0Ijp7ImZpcnN0UGFnZSI6InB1YmxpY2F0aW9uIiwicGFnZSI6InB1YmxpY2F0aW9uIn19). [Link](https://www.researchgate.net/publication/252277701_An_agent-based_requirements_engineering_framework_for_complex_socio-technical_systems)
2. On agent-based software engineering by Nicholas R. Jennings. [Link](http://www2.econ.iastate.edu/tesfatsi/jennings.pdf)
3. What is an Aspect in Requirements Engineering? By [**Hermann Kaindl**](https://www.researchgate.net/profile/Hermann-Kaindl?_tp=eyJjb250ZXh0Ijp7ImZpcnN0UGFnZSI6InB1YmxpY2F0aW9uIiwicGFnZSI6InB1YmxpY2F0aW9uIn19)**.** [Link](https://www.researchgate.net/publication/220738002_What_is_an_Aspect_in_Requirements_Engineering)
4. Aspect-Oriented Requirements Engineering: Approaches and Techniques by [T**awfig Abdelaziz**](https://www.researchgate.net/profile/Tawfig-Abdelaziz?_tp=eyJjb250ZXh0Ijp7ImZpcnN0UGFnZSI6InB1YmxpY2F0aW9uIiwicGFnZSI6InB1YmxpY2F0aW9uIn19). [Link](https://www.researchgate.net/publication/327732449_Aspect-Oriented_Requirements_Engineering_Approaches_and_Techniques)