**LLM-Driven Task Management and**

**Coordination in Space Asset Operations.**

**Objective** : To leverage large language models (LLMs) to automate the parsing of task requests, enable dynamic task prioritization, and coordinate operations across distributed space assets. The LLM will act as the sole computational backbone for interpreting high-level task descriptions, generating actionable plans, and maintaining synchronization between nodes in a decentralized architecture.

#### **Data:**

1. **Task Requests**:
   * Textual descriptions containing mission details (e.g., task type, priority, and constraints).
   * Example: "Schedule satellite X to capture images of region Y by 18:00 UTC."
2. **Operational Logs**:
   * Historical task descriptions and their corresponding outcomes for fine-tuning the LLM.
   * Includes metadata such as task execution time, resource utilization, and completion status.
3. **Node Updates**:
   * Real-time textual updates from satellite nodes (e.g., “Orbital drift detected, delaying imaging task”).
   * Logs of changes in task states communicated between nodes.

#### **Methodology:**

1. **Task Parsing :**
   * Fine-tune a pre-trained LLM to interpret task requests.
   * Extract actionable parameters (e.g., task type, resources required, priority level).
2. **Dynamic Task Reprioritization**:
   * Use LLMs to evaluate real-time inputs from nodes and reprioritize tasks.
   * Generate updated task schedules in natural language for operator readability.
3. **Inter-node Communication**:
   * Enable LLMs to process and relay textual updates between distributed nodes.
   * Allow each node to understand updates and modify local task execution as needed.
4. **Predictive Adjustments**:
   * Leverage historical task data to enable LLMs to predict task delays or conflicts and suggest adjustments.

#### **Implementation:**

1. **LLM Task Parsing**:
   * Use a transformer-based model (e.g., OpenAI API, Hugging Face) for fine-tuned task parsing.
   * The input is a task description; the output is structured text representing actionable plans.
2. **LLM-Driven Coordination**:
   * Deploy LLMs across nodes (satellites, ground stations) to process real-time updates and synchronize task schedules.
   * Example: A node detects orbital debris and notifies others to reschedule imaging tasks.
3. **Real-Time Adaptation**:
   * Nodes use LLMs to analyze incoming updates (e.g., weather changes, delays) and generate adjusted task schedules.
   * Updated schedules are communicated back in natural language to operators and other nodes.
4. **Decentralized Learning**:
   * Fine-tune LLMs on node-specific data to ensure localized decision-making while maintaining cross-node consistency.
5. **Operator Interaction**:
   * Provide an interface where operators input task requests and receive contextual feedback or clarifications generated by the LLM.
6. **Testing**:
   * Simulate task requests and real-time updates to validate the LLM’s ability to parse, prioritize, and coordinate tasks autonomously.