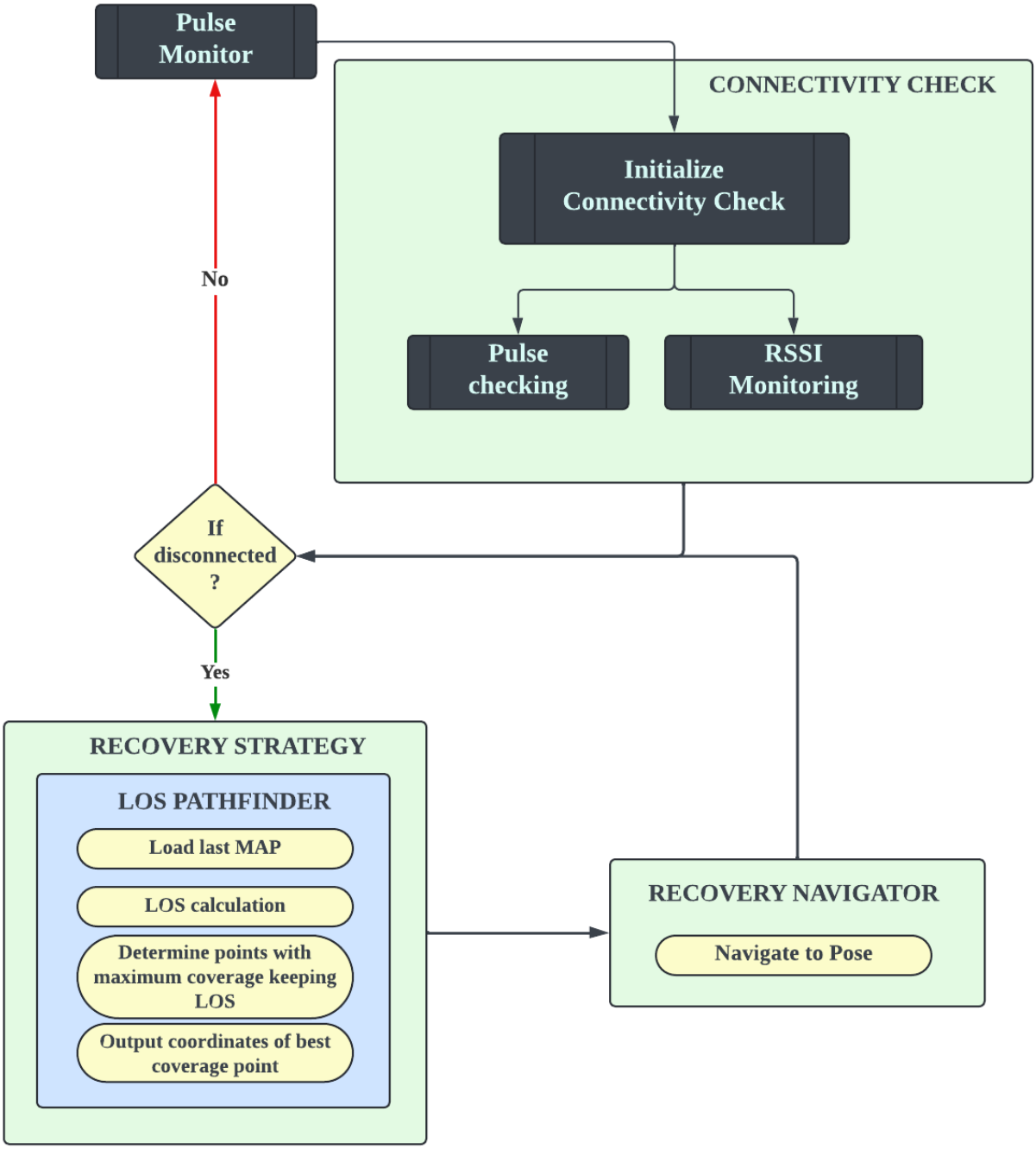
RECOVERY PATHFINDER PACKAGE CONFIGURATIONS



Pulse Monitor:

- Main module of the pathfinder package which initiates all the required modules.
- Contains 1 second timer to check the connectivity of the robot.
- This module can be used to change the disconnectivity checking approach as given below.

```
########################## SELECT ONLY ONE OF THE FOLLOWING CONNECTIVITY CHECKER OPTIONS ########################

self.pulseChecker       = pulseChecker(self)        # Checks for published pulse from each robot
#self.RSSIMonitor       = RSSI(self)                # Use RSSI values of lattepanda board's network interface card

#################################################################################################################
```

- Contains logic of main state machine for recovery.
  - RECOVERYSTATUS.CHECKPULSE
  - RECOVERYSTATUS.RECOVERY
  - RECOVERYSTATUS.NAVIGATE

```python
def monitor_callback(self):
    if (self.RECOVERYMODE == RECOVERYSTATUS.RECOVERY):
        try:
            self.logger.info("Calculating optimum path")
            self.keyPoints = self.LOSpathFinder.getRecoveryPath()
            self.logger.info(f"Keypoints: {self.keyPoints}")


            self.RECOVERYMODE = RECOVERYSTATUS.NAVIGATE

        except Exception as err:
            print("monitor callback error: ",err)
            if (self.recoveryAttempt <= self.MAX_RECOVERY_ATTEMPTS):
                self.logger.info(f"Starting next recovery attempt : {self.recoveryAttempt}")
                self.recoveryAttempt += 1
            else:
                self.logger.warn("Recovery Failed")
                self.RECOVERYMODE = RECOVERYSTATUS.CHECKPULSE
```

Pulse Checker:

- Publishes a header msg containing timestamp and robots ID.
- Module subscribes to all header msgs coming from all the connected robots.
- If robot doesnot receive any header msgs, detect it as robot disconnected from the network.
- 5 second time out is defined before going for recovery stage.

```python
def __init__(self, node):
    self.node              = node
    self.isLost            = None
    timer_period           = 1.0     # seconds
    self.pulse_publisher   = self.node.create_publisher(Header, 'pulse', 2)
    self.pulse_timer       = self.node.create_timer(timer_period, self.pulse_callback)
    self.pulse_check_timer = self.node.create_timer(5.0, self.pulse_check)
```

NOTE: Due to decentralized approach of sending header msgs (pulses) affected by the local time of each robot, a difference between first and current readings are considered.

RSSI Monitor:

- Used to monitor received signal strength by on board computer's network interface card.(Lattepanda board)
- This needs to be improved to work with RAJANT router's network interface.
- Threshold for disconnectivity is defined as -70dbm

```
class RSSI():
    def __init__(self, node):
        self.node           = node
        self.signalThresh   = -70      # dbm
        self.timer          = self.node.create_timer(5.0, self.getSignalLevel)
        self.interface      = 'wlp3s0' # Replace with your wireless interface name
        self.signalAcquired = False
```

Recovery Navigator:

- Uses the the cooridinates given by recovery path calculations to send a goal through action client while receiving progress.
- Uses navigateToPose but actual orientation is neglected.

NOTE: After goal suceeded, robot will check the connectivity status and proceed with the state machine. This helps to reduce back and forth movements of robots at the edge of the map.

To further improve this we could us3 key locations along the path to best coverage point and check connectivity at each location.
(Key points/ junctions are already calculated by LOSPAthfinder module)