```
MACHINE
    Battery+Hardware_M3        //    batteryLevel を「外部に展開する値(BM2)」として新しく「物理的な真のバッテリー残量」を導入
REFINES
    Battery+Hardware_M2
SEES
    Battery+Hardware_C2
VARIABLES
    recharge
    solar_panels_opened
    is_charging
    currentPosition
    atGoal
    plan2D_cost
    plan2C_cost        //    〜ここまでM0-M2で定義
    actualBatteryLevel        //    物理的な真のバッテリー残量
    batteryLevel        //    BM2: 展開される残量(実測より5%少なく表示)
    lastCommand        //    HI4: アクチュエータへの移動コマンド
INVARIANTS
    inv1    :    actualBatteryLevel ∈ N        //    BM2:展開する値は実測値の95%(整数演算での近似)
    inv2    :    batteryLevel = actualBatteryLevel − MARGIN_RATE
    inv3    :    lastCommand ∈ N
EVENTS
    INITIALISATION    ≙
      extended
    STATUS
      ordinary
    BEGIN
      act_init_battLevel    :    batteryLevel ≔ MAX_BATTERY        //    バッテリレベルの初期状態は満充電状態
      act_init_recharge    :    recharge ≔ FALSE
      act_init_solar_panels    :    solar_panels_opened ≔ FALSE
      act_init_is_charging    :    is_charging ≔ FALSE
      act_init_currentPosition    :    currentPosition ≔ START_POSITION
      act_atGoal    :    atGoal ≔ FALSE
      act_p2D    :    plan2D_cost :∈ PLAN2D_COST_SET
      act_p2C    :    plan2C_cost :∈ PLAN2C_COST_SET
      act6    :    lastCommand ≔ 0        //    コマンドなし
      act7    :    actualBatteryLevel ≔ MAX_BATTERY_PHYSICAL        //    満充電状態
    END

    Send_Movement_Command    ≙        //    HI5:移動コマンドの送信 (HI4) と実行をコマンド設定とバッテリ消費で表現
    STATUS
      ordinary
    REFINES
      Consume_Battery_Refined
    WHEN
      grd_batt_min    :    batteryLevel > MIN_BATTERY        //    バッテリレベルが最小より大きい場合だけ消費可能
      grd_rechargeOff    :    recharge = FALSE        //    充電が必要なフラグが立っていない場合のみ消費(移動)できる
      grd_batt_check_plan    :    batteryLevel > plan2D_cost + plan2C_cost        //    バッテリレベルが十分かどうか
      grd_need_charge    :    recharge = FALSE
      grd_charging    :    is_charging = FALSE
      grd_under_act_batt_min    :    actualBatteryLevel > MIN_BATTERY_PHYSICAL
      grd_safe_consume    :    (actualBatteryLevel − (plan2D_cost + plan2C_cost)) − MARGIN_RATE > MIN_BATTERY
    THEN
      act_set_command    :    lastCommand ≔ 1        //    HI4: 移動コマンドの送信(値は仮)
      act_consume_act_batt    :    actualBatteryLevel ≔ actualBatteryLevel − (plan2D_cost + plan2C_cost)        //    HI5: 送信されたコマンドは実行され、バッテリレベルが減る
      act_update_batt    :    batteryLevel ≔ (actualBatteryLevel − (plan2D_cost + plan2C_cost)) − MARGIN_RATE        //    BM1/BM2: バッテリー監視と報告値の更新
    END

    Signal_Recharge_Refined    ≙
      extended
    STATUS
      ordinary
    REFINES
      Signal_Recharge_Refined
    WHEN
      grd_arrive_charger    :    currentPosition ∈ chargers        //    ローバーが充電位置にいる場合
      grd_rechargeOff    :    recharge = FALSE        //    リチャージフラグが立っていない
      detect_BattLow    :    batteryLevel < plan2D_cost + plan2C_cost        //    目標地点＋充電地点の移動に必要なバッテリレベルが不足
    THEN
      act1    :    recharge ≔ TRUE
    END

    Start_Charging    ≙
      extended
    STATUS
      ordinary
    REFINES
      Start_Charging
```

```
    WHEN
      grd1   :   recharge = TRUE       //   リチャージフラグがONの場合
    THEN
      act1   :   is_charging ≔ TRUE
      act2   :   solar_panels_opened ≔ TRUE       //   ソーラーパネルを開く(HI6)
    END

    Recharge_Battery_Refined   ≙
    STATUS
      ordinary
    REFINES
      Recharge_Battery
    WHEN
      grd1   :   is_charging = TRUE       //   充電中であること
      grd2   :   batteryLevel < MAX_BATTERY       //   バッテリ電圧がMAX未満であること
      grd_over_act_batt_max   :   actualBatteryLevel + step_charge + MARGIN_RATE ≤ MAX_BATTERY_PHYSICAL
    THEN
      act_charge_act_batt   :   actualBatteryLevel ≔ actualBatteryLevel + step_charge       //   実際のバッテリレベルの更新
      act_update_batt   :   batteryLevel ≔ (actualBatteryLevel + step_charge) − MARGIN_RATE       //   BM1/BM2: バッテリー監視と報告値の更新
    END

    Complete_Charging   ≙
      extended
    STATUS
      ordinary
    REFINES
      Complete_Charging
    WHEN
      grd1   :   is_charging = TRUE
      grd2   :   batteryLevel + step_charge ≥ MAX_BATTERY       //   バッテリレベルがMAXになったら
    THEN
      act1   :   is_charging ≔ FALSE       //   ローバの充電終了
      act2   :   solar_panels_opened ≔ FALSE       //   ソーラパネルも閉じる
      act3   :   recharge ≔ FALSE       //   リチャージフラグもFALSE
      act_update_p2D   :   plan2D_cost :∈ PLAN2D_COST_SET       //   次の移動プランのために、目標地点までのコスト情報を集合から再取得する
      act_update_p2C   :   plan2C_cost :∈ PLAN2C_COST_SET       //   次の移動プランのために、充電地点地点までのコスト情報を集合から再取得する
    END

    Reach_Goal   ≙
      extended
    STATUS
      ordinary
    REFINES
      Reach_Goal
    WHEN
      grd1   :   currentPosition ∈ goals
      grd2   :   atGoal = FALSE
    THEN
      act1   :   atGoal ≔ TRUE       //   HI3: 目標に到達した時のみセット
    END

    Reset_atGoal_Flag   ≙
      extended
    STATUS
      ordinary
    REFINES
      Reset_atGoal_Flag
    WHEN
      grd1   :   atGoal = TRUE
    THEN
      act1   :   atGoal ≔ FALSE
    END

END
```