SystemAdmin

System Initialization

create(shortName, longName, ratePerKWh) → :PowerCompany
Initialize PowerCompany with basic configuration

create(plantId, location, buildCost, productionCostPerKWh) → :PowerPlant
Create power generation facility

addPowerPlant(PP1)
powerPlants.add(PP1)
success

create(substationId, location, buildCost, maintenanceCost) → :Substation
Create distribution substation

addSubstation(SUB1)
canAddSubstation()

alt [substations.size() < 20]

checkDistance(PP1.location)
location.isWithinDistance(PP1.location, MAX_PLANT_DISTANCE)

alt [within distance]
substations.add(SUB1)
true

[too far]
false
(distance constraint violated)

[max substations reached]
false
(max 20 substations)

create(transformerId, location, installationCost, maintenanceCost) → :Transformer
Create transformer unit

addTransformer(TRANS1)
canAddTransformer()

alt [transformers.size() < 10]

checkDistance(SUB1.location)
location.isWithinDistance(SUB1.location, MAX_SUBSTATION_DISTANCE)

alt [within distance]
transformers.add(TRANS1)
true

[too far]
false
(distance constraint violated)

[max transformers reached]
false
(max 10 transformers)

create(accountNumber, name, location, customerType) → :Customer
Register new customer

addCustomer(CUST1)
canAddCustomer()

alt [customers.size() < 5]

checkDistance(TRANS1.location)
location.isWithinDistance(TRANS1.location, MAX_TRANSFORMER_DISTANCE)

alt [within distance]
customers.add(CUST1)
true

[too far]
false
(distance constraint violated)

[max customers reached]
false
(max 5 customers)

addCustomer(CUST1)
customers.add(CUST1)
success

create(employeeId, name, startDate, hourlyWage) → :Employee
Hire employee

addEmployee(EMP1)
employees.add(EMP1)
success

create(cycleId, startDate, endDate) → :BillingCycle
status = ACTIVE
Initialize first billing cycle

System Running

System is operational
Power generation and consumption tracking begins

System Shutdown

closeCycle()
status = CLOSED
cycle closed

shutdown()
stop()
disconnect()
disconnect()
finalizeAccount()
terminate()

System shutdown complete

SystemAdmin

:PowerCompany   :PowerPlant   :Substation   :Transformer   :Customer   :Employee   :BillingCycle

SystemAdmin   :PowerCompany   :PowerPlant   :Substation   :Transformer   :Customer   Customers

Start Billing Period

create(cycleId, startDate, endDate) → :BillingCycle

status = ACTIVE
New billing cycle begins

billing cycle created

Power Generation and Tracking

loop [during billing period]

recordPowerProduction(kWh)

powerProduced += kWh
Track total power generated for this cycle

production recorded

calculateProductionCost()

cost = powerProduced × productionCostPerKWh

updateCosts(cost)

totalCosts += cost

create(transactionId, date, cost, PRODUCTION_COST) → :Transaction

transactions.add(TXN)

cost updated

Maintenance Costs

loop [for each substation]

getMaintenanceCost()

maintenanceCost

recordMaintenanceCost(maintenanceCost)

totalCosts += maintenanceCost

create(transactionId, date, cost, MAINTENANCE_COST) → :Transaction

transactions.add(TXN)

loop [for each transformer]

getMaintenanceCost()

maintenanceCost

recordMaintenanceCost(maintenanceCost)

totalCosts += maintenanceCost

create(transactionId, date, cost, MAINTENANCE_COST) → :Transaction

transactions.add(TXN)

Customer Power Usage

loop [during billing period]

recordPowerUsage(kWh)

powerUsage += kWh
Track customer consumption

usage recorded

End Billing Period and Calculate Bills

closeCycle()

status = PROCESSING

cycle closed

processBillingCycle(BC)

loop [for each customer in customers]

forEach(customer)

calculateBill(ratePerKWh)

bill = powerUsage × ratePerKWh

totalAmountDue += bill

billAmount

billAmount

totalRevenue += billAmount

create(transactionId, date, billAmount, CUSTOMER_BILL) → :Transaction

transactions.add(TXN)
Record billing transaction

resetPowerUsage()

powerUsage = 0
Reset for next cycle

Calculate Financial Summary

calculateTotalCosts()

costs = sum of all cost transactions

calculateProfit()

profit = totalRevenue − totalCosts

Financial Summary:
- Total Revenue: $X from customer bills
- Total Costs: $Y (production + maintenance)
- Net Profit: $(X−Y)

billing cycle processed
profit = $Z

status = CLOSED

Customer Payment (Optional)

opt [customer makes payment]

makePayment(amount)

totalAmountDue −= amount

payment received

recordPayment(amount)

create(transactionId, date, amount, CUSTOMER_PAYMENT) → :Transaction

transactions.add(TXN)

Reset for Next Cycle

resetProduction()

powerProduced = 0
Ready for next billing cycle

Billing cycle complete
System ready for next period

SystemAdmin   :PowerCompany   :BillingCycle   :PowerPlant   :Substation   :Transformer   :Customer   :Transaction   Customers