# QubecTalk Domain Specific Language

A Samuel Pottinger

2024-09-16

**Summary**: Modelers may engage with a domain specific language called QubecTalk in order to construct analyses relevant to the Montreal Protocol. Specifically, this human-readable text-based format can describe scenarios to a simulation engine. Expected to be written in JavaScript, we anticipate that the interpreter for this DSL can execute in browser or locally on a machine. It may run as part of an application with a graphical user interface. However, it may also be invoked directly or through a script from the command line. Additionally, these programs can be written "manually" within a code editor or through a graphical user interface (which does not require authors to engage code directly). This document describes QubecTalk and offers a realistic full featured example of a set of simulations written in the language.

## 1 Purpose

This document details a domain specific language called QubecTalk. This "DSL" is part of a larger system which allows for the construction of mathematical simulations relevant to the Montreal Protocol. These longitudinal[1] projections simulate applications, substances, and possible interventions. In this process, this system seeks to support a broad ecosystem of actors across a longer journey of evaluating and comparing options through analysis outputs.

### 1.1 Audience

Authors[2] of these models range from those seeking a cursory or exploratory projection to experts crafting highly specific or detailed scenarios. To support this wide range of uses, we offer a domain specific language (DSL) called QubecTalk. Analyses which may include multiple simulations are described in single file programs which are run through an interpreter. This "engine" is expected to be

[1] Over time projections which simulate relevant metrics across a discrete set of years as specified by model authors.

[2] In alignment with prior community perspectives, we refer to our audience as authors regardless of skill level. This nomenclature is used in place of "users" as a term. This recognizes and values their agency and their centrality in co-creation within this system.

written in JavaScript / ECMAScript. Authors can interact with this technology through different modalities:

- **Graphical interface**: Some at the start of a project may simply author simulations through a graphical user interface, building out projections and simple policy scenarios without interacting with any code directly.
- **Code editor**: Others may write code to tailor sophisticated simulations to very specific scenarios. These authors may also take advantage of algorithms like Monte Carlo to express uncertainties or explore many potential futures.

Of course, analyses may migrate from quick sketch without editing code to intricate projections engaging the source extensively as ideas are refined and explored. This may also take place socially: one author may start a rough "sketch" of an analysis in a graphical interface later refined by another collaborator working in a code editor. Regardless, this document describes QubecTalk and offers an example of a realistic model translated into this format.

## 1.2 Objectives

QubecTalk serves multiple roles in this context:

- **Internal representation**: In the case that source code is generated and manipulated by a graphical interface to support authors not wishing to program in code, scripts become internal representations of inputs. This allows for saving and loading of configuration in a structured purpose-built format.
- **Expressiveness**: In the case that scripts are further refined by modifying the source code in a more traditional programming context, these scripts may be engaged in tools like full featured editors or version control. This format may be more expressive (ergonomic and flexible) for expert authors seeking to create very sophisticated models.
- **Standardization**: Regardless of how scripts are constructed, these programs offer a unified and structured interface to the engine. This also affords some reusability of this system, allowing for the adoption of other interfaces (graphical or otherwise) in the future.
- **Approachability**: Some authors may be less likely to experience this system from a blank text file and, instead, will edit code started by the graphical user interface. Therefore, this DSL attempts to optimize for readability by audiences which may be less familiar with programming.

This document first outlines a language definition and then offers an example implementation which rebuilds a reference model.

## 1.3 License

This document is available under the Creative Commons CC-BY 4.0 International License as described at https://creativecommons.org/licenses/by/4.0/deed.en.

## 2 Definition

This document first provides an outline of language features.

- We start with discussion of the structure of a QubecTalk program.
- Next, we transition to discussion of system definition including applications and substances.
- Finally, we consider simulation definition and miscellaneous language features.

A program may contain multiple simulations where each includes zero or more policies on top of a "business as usual" simulation state. For more information, see the example.

### 2.1 Structure

Generally there is a definition of a system under a business as usual scenario followed by policies and simulation tasks. This DSL operates through a "stanza" structure:

```
start about
  # Name: "Simulation Name"
  # Description: "Simulation Description"
  # Author: "Simulation Author"
  # Date: "Simulation Date"
end about

start default
  # ...
end default

start policy "Policy 1"
  # ...
end policy

start policy "Policy 2"
  # ...
end policy

start simulations
  # ...
end simulations
```

Each section provides a type and, if appropriate, label for a set of commands.

## 2.2 System

The components of the system interact with each other under policy modification to determine substance consumption and GHG impacts.

### 2.2.1 Applications

Analysis may consider substances across various different applications. Some examples:

- **Refrigeration:** Domestic, small commercial, large commercial, industrial, transport.
- **Air conditioning**: Residential, commercial, central / large, mobile.
- **Medical aerosols**: MDIs, etc.

QubecTalk defines these as follows:

```
define application "domestic refrigeration"

  # Optional variables across substances within an application.

  uses substance "HFC-134a"
    # Logic for a substance
  end substance

end application
```

Note that the same substances may be included across multiple applications. Furthermore, these may be imported or built domestically. When defining these substances, an initial charge should be provided:

```
initial charge with 0.12 kg / unit for manufacture
initial charge with 0.30 kg / unit for import
```

If this is omitted and the substance has sales, an error message may be shown. Note that the initial charge statement would appear where it says `# Logic for a substance` along with other logic for that substance.

### 2.2.2 Sales

The sale of substances typically defines the equipment population. QubecTalk considers sales to fit into two groups: `manufacture` and `import`. Though not currently used, `export` is reserved for future use.

**Domestic**: Starting with an example of specifying substances both produced and consumed domestically. There are typically three important parameters to specify:

- Manufacturing (volume)
- Sales growth (YoY %)
- Initial charge (mt)

This can be established through the following example statements:

```
set manufacture to 350000 mt during year 1
change sales by +5 % each year during years beginning to 6
change import by +3 % each year during years 6 to onwards
```

Note that, if not specifying a year, a statement will apply across the entire timeseries. If no stream is specified, a statement will apply proportionally across sub-streams. For example, in the case of sales, changes apply proportionally across domestic manufacturing and imports.

**Trade**: Imported equipment follows a similar pattern and typical parameters:

- Imported sales (volume)
- Sales growth (YoY %)
- Initial charge (mt)

These are specified using the same statements but indication that the sales flow through the imports channel.

```
set sales to 90000 mt during year 1
change import by +5 % each year
change manufacture by +3 % each year during years 6 to 9
```

Exports are reserved for future use. Again, if no stream is specified, it will apply proportionally across sub-streams (domestic manufacturing and imports).

### 2.2.3 Population

Historic equipment populations are typically inferred by sales before applying a change rate into the future. However, they may be specified manually as well.

**Initializing equipment**: Equipment levels at any year may be specified manually and this approach can be used to initialize the equipment population:

```
set equipment to 2000 units during year 1
```

This sets the current year equipment which assumes the prior year's levels stay the same. To specify a new level for equipment not sold in the current year, use `priorEquipment`.

```
set priorEquipment to 8 * sales during year 1
```

These approaches can be mixed but, generally, authors will apply a multiplier against a specific (typically first) year of sales.

**Adding equipment**: Equipment is added based on sales. For most authors, this is preferred. However, to override this behavior, use `set equipment` to override the value.

**Removing equipment**: Most analyses will remove equipment through deprecation or retirement. Consider these examples:

```
retire 6.7 %
retire 5 % during years 1 to 5
```

If a year is not specified, it will be apply across the entire timeseries. In addition to accepting percentages, an absolute number of units may be given. Note that authors may also choose to use `set equipment` as seen in adding equipment.

**Service schedule**: The language allows for specifying annual recharge quantity from equipment (mt) which is typically applied to an annual percentage of total equipment.

```
recharge 10 % with 0.12 kg / unit
recharge 5 % with 0.12 kg / unit during years 1 to 5
```

One may also recharge a number of units by changing `%` to `units`.

### 2.2.4 Consumption

GHG equivalencies can be made through the `equals` command like so:

```
equals 1430 tCO2e / mt
```

Note that `tCO2e` refers to $CO_2$ equivalent in metric tonnes.

## 2.3 Policies

Policies can be defined by name and typically make changes to the business as usual created within the `default` stanza.

```
start policy
  # ...
end policy
```

Policy names must be unique.

### 2.3.1 Permit / prohibition

Many policies work by placing a limit on sales.

```
cap sales to 75 %
cap import to 750 mt during years 4 to 5
```

Note that sales are the only type which can currently accept the cap. If specifying an aggregate stream, it will be applied proportionally. For example, for sales, the effect will be split proportionally between manufacture and imports.

### 2.3.2 Recycling

Recycling programs are defined as percent refrigerant recovery and loss rate for reuse

```
recover 5 % with 100 % reuse during year 3
recover 10 % with 100 % reuse during years 4 to onwards displacing 50%
```

By default, this will displace imports and exports proportional to their prior value. One may specify a `displacing` rate where 80% means that 80% of the recycled volume offsets virgin production and 20% goes to sales that would not have happened otherwise. Note that rebound (recycling does not displace virgin 1 to 1) requires a set command.

### 2.3.3 Replace

Transition by incentive or other similar mechanisms can be expressed through a `replace` command.

```
replace 5 % of sales with "low gwp"
replace 100 units of sales with "low gwp" during years 3 to onwards
replace 10 mt of manufacturing with "low gwp" during years 3 to onwards
```

This will change the equipment population where percentages are assumed to be percentage of equipment population. If providing mt, this will convert to equipment units. Furthermore, this will displace from manufacturing and imports proportionally if a sales stream is not specified.

## 2.4 Simulation

A set of zero or more policies defines a scenario.

```
simulate "Business as Usual" from years 1 to 20

simulate "Standalone Import Ban"
  using "Import Ban"
from years 1 to 20

simulate "Combined"
  using "Import Ban"
  then "Domestic Permit"
  then "Efficiency Incentive"
from years 1 to 20

simulate "Combined"
  using "Import Ban"
  then "Domestic Permit"
  then "Efficiency Incentive"
from years 1 to 20 across 1000 trials
```

Each scenario will project out a number of years and report consumption of HFCs and alternatives in CO2 tonnes. In order to report changes to business as usual, do not provide any policies.

## 2.5 Language

A few places where other language features may helpful to more expert users.

### 2.5.1 Variables

Full arithmetic expressions with local variables.

```
define x as 5 + 6 * 7 ^ 8
```

These are limited to the scope of the enclosing `start` and `end` pair.

### 2.5.2 Conditional

Conditionals are expressed as Python-style ternary operators.

```
set y to 1 if x == 5 else 0 endif
```

The operators supported: `<`, `>`, `<=`, `>=`, `==`, and `!=` for less than, greater than, less than or equal, greater than or equal, equals, and not equals. Boolean expressions may be combined through and / or (`and`, `or`). True is returned as 1 and false as 0.

### 2.5.3 Uncertainties

Typically uncertainties are applied on demand such as sales growth rates, retirement, and recharge. In general, sampling cam be applied in any expression:

```
sample uniformly from 5 to 10 %
sample normally mean of 5 and std of 2 %
```

For example, one may specify a distribution of possible values for retirement rate:

```
retire sample normally mean of 6.7 and std of 1 %
```

When building these simulations, users should also include a number of trials (`across 1000 trials`).

### 2.5.4 Timeseries

The current year can be found in a variable called `yearsElapsed` and `yearAbsolute` for years since the start of the simulation and absolute year number respectively. These variables can not be set or redefined.

```
yearAbsolute * 5 + 2
```

These values can be used in any expression. Similar variables are also available for other timescales: `monthsElapsed`, `monthAbsolute`, `daysElapsed`, and `dayAbsolute`. These do not reset from year to year but accumulate from the start of the simulation.

### 2.5.5 Get and set

Developers can manually set a value of a stream:

```
set import to 1 mt in year 2
set sales of "HFC-134a" to 1 mt
set manufacture of "HFC-134a" to 1 mt during year 2
```

If year is not specified, it is applied in all years. Values can also be used in expressions.

```
define currentSales as get sales  # current scope
define currentSales as get sales as kg  # set units
define salesOther as get sales of "HFC-134a" as kg  # different substance same app
define salesOtherMobileAC as get sales of "HFC-134a" as kg
```

Setting an aggregate stream like sales will cause the value to be distributed proportionally to the prior value of the sub-streams. In the case of this example, this would be applied across import and domestic manufacturing.

# 3 Units

Various statements can specify units. These are not assignable to variables so are only included in supporting commands.

## 3.1 Supported units

Available units include the following:

- `kg` or `mt`: Volumes as kilograms or megatons
- `tCO2e`: Consumption as tons of CO2e
- `unit` or `units`: Population size as equipment units
- `year` or `years`: Time as year or years elapsed from start of simulation.
- `%`: Percentage

The following are not currently supported but reserved for future use:

- `day` or `days`: Time as day or days elapsed from start of simulation.
- `month` or `months`: Time as month or months elapsed from start of simulation.

These may be expressed as a ratio like `tCO2e / mt`. In the case that the division is by a time unit like `year`, this is assumed to be a compounding value per time unit where the first day / month / year is zero.

## 3.2 Behavior

Interpretation and unit conversion depends on the command:

| **Op** | **Type** | **Units** | **Percent** | **Ratios** |
|---|---|---|---|---|
| Initial Charge | Volume | Volume distributed across population. Consumption converted to volume. | Not supported. | Div by units will multiply by pop size. |
| Set Sales | Volume | Volume. Consumption converted to volume, units converted to volume. Proportional if not sub-stream. | $x = x * \%$ | Div by time multiplies `yearsElapsed`, div by units multiplies population, div by consumption multiplies consumption. |
| Set Equipment | Population | Units. Consumption converted to volume, volume converted to units. | $x = x * \%$ | Div by time multiplies `yearsElapsed`, div by volume multiplies volume, div by consumption multiplies consumption. |
| Set Consumption | Consumption | Consumption distributed across volume streams. | $x = x * \%$ | Div by time multiplies `yearsElapsed`, div by units multiplies population, div by volume converts to div by units. |
| Change Sales | Volume | Volume delta. Consumption converted to volume, units converted to volume. Proportional if not sub-stream. | $x = x * (1 + \%)$ | Div by time multiplies `yearsElapsed`, div by units multiplies population, div by consumption multiplies consumption. |
| Change Equipment | Population | Units delta. Consumption converted to volume, volume converted to units. Proportional. | $x = x * (1 + \%)$ | Div by time multiplies `yearsElapsed`, div by volume multiplies volume, div by consumption multiplies consumption. |

| **Op** | **Type** | **Units** | **Percent** | **Ratios** |
|---|---|---|---|---|
| Change Consumption | Consumption | Consumption delta distributed across volume. | $x = x*(1+\%)$ | Div by time multiplies `yearsElapsed`, div by units multiplies population, div by volume converts to div by units. |
| Retire | Population | Consumption converted to volume, volume converted to units. | $r = x * \%$ | Div by time multiplies `yearsElapsed`, div by volume multiplies volume, div by consumption multiplies consumption. |
| Cap Sales | Volume | Volume. Consumption converted to volume, units converted to volume. Proportional if not sub-stream. | $c = x * \%$ | Div by time multiplies `yearsElapsed`, div by units multiplies population, div by consumption multiplies consumption. |
| Cap Equipment | Population | Units. Consumption converted to volume, volume converted to units. | $c = x * \%$ | Div by time multiplies `yearsElapsed`, div by volume multiplies volume, div by consumption multiplies consumption. |
| Cap Consumption | Consumption | Consumption total distributed across volume streams. | $c = x * \%$ | Div by time multiplies `yearsElapsed`, div by units multiplies population, div by volume converts to div by units. |

| Op | Type | Units | Percent | Ratios |
|---|---|---|---|---|
| Recycle | Volume | Volume. Consumption converted to volume, units converted to volume. Proportional. | Uniform across streams. | Div by time multiplies `yearsElapsed`, div by units multiplies population, div by consumption converts to div by units. |
| Replace Sales | Volume | Volume. Consumption converted to volume, units converted to volume. Proportional if not sub-stream. | Uniform across streams. | Div by time multiplies `yearsElapsed`, div by units multiplies population, div by consumption converts to div by units. |
| Replace Equipment | Volume | Converted to volume. | Uniform across streams. | Div by time multiplies `yearsElapsed`, div by volume converts to units, div by consumption converts to div by volume. |
| Replace Consumption | Volume | Converted to volume. | Uniform across streams. | Div by time multiplies `yearsElapsed`, div by volume converts to units, div by units converts to volume. |
| Get Sales | Volume | Volume (`kg`) | N/A | N/A |
| Get Equipment | Population | Population (`units`) | N/A | N/A |
| Get Consumption | Consumption | Consumption (`tCO2e`) | N/A | N/A |

In addition to the strategies described above, conversion may invert ratio units to achieve a valid configuration. Otherwise, if behavior is not specified above, the correct behavior is undefined. In these cases, units are "unsupported" for a command and will result in an error at runtime. Note that units are converted to consumption and volumes with amortized values.

### 3.3 Proportionality with units

Finally, "proportional" operations on percentages will apply the same to all sub-streams if using a percentage or unit which includes division (ex: `kg / mt`) otherwise the effect size will be proportional to the size of the sub-stream prior to the operation.

## 4 Engine

While the system can operate with constraints through `cap`, updates are applied in the order specified within code. Within the context of streams, one stream can cause another to update with these changes propagating when a stream is changed. The following dependencies see changes made at the start of the chain cause changes downstream. Note that these are not transitive beyond this table so changes to sales does not cause an consumption trigger.

| **Trigger** | **Sales** | **Consumption** | **Population** |
|---|---|---|---|
| Sales (manufacture, import) | Changed directly | Recalc using equals value | Recalc new and total equip after subtract recharge and recycle, retiring at recharge if needed. |
| Consumption | Recalc after add recharge and subtract recycle. | Changed directly | Recalc new and total equip after subtract recharge and recycle, retiring at recharge if needed. |
| Population (equipment) | Recalc after add recharge and subtract recycle. | Recalc after sales update using equals | New equip and total current equip changes but prior equip untouched. |
| Population (priorEquipment) | No change | No change | Change prior equip. New and total equipment recalculated. |

These updates happen automatically within the interpreter and all use the following which are recorded internally with original units.

### 4.1 Saved parameterization

For engine internal updates, some parameters are saved for each substance. Any compatible units are acceptable though examples are included for reference.

| **Parameter** | **Example units** | **Set by** |
|---|---|---|
| GHG intensity | tCO2e / kg | `equals` |
| Recharge population | % | `recharge` |
| Recharge intensity | kg / unit | `recharge` |
| Recovery rate | % | `recover` |
| Yield rate on recycling | % | `recover` |

If multiple statements write to these parameters, the latest value is used when propagating.

## 4.2 Responses

In calculating downstream effects, some streams may be recalculated based on an author-specified change in a triggering stream.

### 4.2.1 Population response

Propagating effects to the equipment population can happen either by changing the new equipment deployment in a year or by changing the prior population size. However, as changing the prior population simply causes the change in population to be recalculated, both operations result in the same outcome:

- Determine retirement from prior population.
- Remove recharge from total available.
- Add recycling to sales to get total available substance using offset rate.
- Convert remaining to added units.
- Determine final delta as added units minus retirement.

Note that retirement applies to prior population size. While the new population delta may be negative, set negative total populations as zero (interpret as net exports).

### 4.2.2 Consumption response

Recalculating consumption largely relies on manufacturing numbers and applies to the substance regardless of where the substance gets used.

- Remove imports from sales.
- Offset manufacturing with recycling (assumed proportional to import / domestic share) using offset rate.
- Convert to consumption.
- Set negative total consumption to zero (treat net negative consumption as credit taken elsewhere).

Note that consumption are applied to point of manufacture (exporter not importer).

## 4.3 Sales response

Sales updating to changes in other variables scales imports and manufacturing proportionally as to meet the requirements of recharge and sales.

- Determine needs for recharge.
- Determine needs for new equipment deployment.
- Subtract recycling Given di.
- Update import and domestic sales proportionally.

Note that negative sales are assumed as exports.

## 5 Example

This example translates a reference model into QubecTalk using the language defined above.

```
start default

  define application "dom refrig"

    uses substance "HFC-134a"
      equals 1430 tCO2e / mt

      # Domestic production
      initial charge with 0.12 kg / unit for manufacture
      set manufacture to 350000 kg during year 1
      change manufacture by +5 % each year during years 1 to 5
      change manufacture by +3 % each year during years 6 to 9

      # Trade
      initial charge with 0.30 kg / unit for import
      set import to 90000 kg during year 1
      change import by +5 % each year during years 1 to 5
      change import by +3 % each year during years 6 to 9

      # Service
      retire 6.7 % each year
      recharge 10 % each year with 0.12 kg / unit

      # Historic units already deployed
      set priorEquipment to (get equipment as units * 7) units during year 1
    end substance

    uses substance "R-600a"
      equals 6 tCO2e / mt
      # Domestic production
      initial charge with 0.05 kg / unit for manufacture
      set manufacture to 200000 kg during year 1
      change manufacture by +8 % each year

      # Trade
      initial charge with 0.05 kg / unit for import
      set import to 10000 kg during year 1
      change import by +8 % each year

      # Service
      retire 6.7 % each year
      recharge 10 % each year with 0.05 kg / unit

      # Historic units already deployed
      set priorEquipment to (get equipment as units * 7) units during year 1
    end substance

  end application

  define application "com refrig"

    uses substance "HFC-134a"
      equals 1430 tCO2e / mt

      # Domestic production
      initial charge with 0.30 kg / unit for manufacture
      set manufacture to 90000 kg during year 1
      change manufacture by +5 % each year during years 1 to 5
      change manufacture by +3 % each year during years 6 to 9

      # Trade
      initial charge with 0.30 kg / unit for import
      set import to 90000 kg during year 1
      change import by +5 % each year during years 1 to 5
      change import by +3 % each year during years 6 to 9

      # Service
      retire 6.7 % each year
      recharge 10 % each year with 0.30 kg / unit

      # Historic units already deployed
      set priorEquipment to (get equipment as units * 7) units during year 1
    end substance

    uses substance "R-600a"
```

```
    equals 6 tCO2e / mt

    # Domestic production
    initial charge with 0.12 kg / unit for manufacture
    set manufacture to 10000 kg during year 1
    change manufacture by +8 % each year

    # Trade
    initial charge with 0.12 kg / unit for import
    set import to 10000 kg during year 1
    change import by +8 % each year

    # Service
    retire 6.7 % each year
    recharge 10 % each year with 0.12 kg / unit

    # Historic units already deployed
    set priorEquipment to (get equipment as units * 7) units during year 1
  end substance

  uses substance "R-404A"
    equals 3922 tCO2e / mt

    # Domestic production
    initial charge with 0.30 kg / unit for manufacture
    set manufacture to 30000 kg during year 1
    change manufacture by +5 % each year

    # Trade
    initial charge with 0.30 kg / unit for import
    set import to 10000 kg during year 1
    change import by +5 % each year

    # Service
    retire 6.7 % each year
    recharge 10 % each year with 0.12 kg / unit

    # Historic units already deployed
    set priorEquipment to (get equipment as units * 7) units during year 1
  end substance

end application

define application "res AC"

  uses substance "R-410A"
    equals 2082 tCO2e / mt

    # Domestic production
    initial charge with 0.90 kg / unit for manufacture
    set manufacture to 175000 kg during year 1
    change manufacture by +5 % each year during years 1 to 5
    change manufacture by +3 % each year during years 6 to 9

    # Trade
    initial charge with 0.90 kg / unit for import
    set import to 20000 kg during year 1
    change import by +5 % each year during years 1 to 5
    change import by +3 % each year during years 6 to 9

    # Service
    retire 6.7 % each year
    recharge 10 % each year with 0.90 kg / unit

    # Historic units already deployed
    set priorEquipment to (get equipment as units * 7) units during year 1
  end substance

  uses substance "HFC-32"
    equals 632 tCO2e / mt

    # Domestic production
    initial charge with 0.68 kg / unit for manufacture
    set manufacture to 85000 kg during year 1
    change manufacture by +8 % each year

    # Trade
    initial charge with 0.68 kg / unit for import
    set import to 9000 kg during year 1
    change sales by +8 % each year

    # Service
    retire 6.7 % each year
    recharge 10 % each year with 0.68 kg / unit

    # Historic units already deployed
    set priorEquipment to (get equipment as units * 7) units during year 1
  end substance

  uses substance "R-290"
    equals 6 tCO2e / mt
```

```
    # Domestic production
    initial charge with 0.68 kg / unit for manufacture
    set manufacture to 10000 kg during year 1
    change manufacture by +5 % each year

    # Trade
    initial charge with 0.68 kg / unit for import
    set import to 10000 kg during year 1
    change sales by +5 % each year

    # Service
    retire 6.7 % each year
    recharge 10 % each year with 0.35 kg / unit

    # Historic units already deployed
    set priorEquipment to (get equipment as units * 7) units during year 1
  end substance

end application

define application "mobile AC"

  uses substance "HFC-134a"
    equals 1430 tCO2e / mt

    # Domestic production
    initial charge with 0.90 kg / unit for manufacture
    set manufacture to 175000 kg during year 1
    change manufacture by +5 % each year during years 1 to 5
    change manufacture by +3 % each year during years 6 to 9

    # Trade
    initial charge with 0.90 kg / unit for import
    set import to 20000 kg during year 1
    change import by +5 % each year during years 1 to 5
    change import by +3 % each year during years 6 to 9

    # Service
    retire 6.7 % each year
    recharge 10 % each year with 0.90 kg / unit

    # Historic units already deployed
    set priorEquipment to (get equipment as units * 7) units during year 1
  end substance

  uses substance "R-1234yf"
```

```
      equals 6 tCO2e / mt

      # Domestic production
      initial charge with 0.90 kg / unit for manufacture
      set manufacture to 85000 kg during year 1
      change manufacture by +8 % each year

      # Trade
      initial charge with 0.90 kg / unit for import
      set import to 9000 kg during year 1
      change import by +8 % each year

      # Service
      retire 6.7 % each year
      recharge 10 % each year with 0.90 kg / unit

      # Historic units already deployed
      set priorEquipment to (get equipment as units * 7) units during year 1
    end substance

  end application

end default


start policy "dom refrig reuse"

  modify application "dom refrig"

    modify substance "HFC-134a"
      recover 5 % with 100 % reuse during year 3
      define level as limit (yearAbsolute - 3) * 10 to [0, 30]
      recover level % with 100 % reuse during years 4 to onwards
    end substance

  end application

end policy


start policy "dom refrig low-GWP"

  modify application "dom refrig"

    modify substance "HFC-134a"
      define level as limit (yearAbsolute - 2) * 20 to [0, 100]
      replace level % of manufacture with "R-600a" during years 3 to onwards
      replace level % of import with "R-600a" during years 3 to onwards
    end substance

  end application

end policy


start policy "com refrig reuse"

  modify application "com refrig"

    modify substance "HFC-134a"
      recover 5 % with 100 % reuse during year 3
      define longTermRecovery as limit (yearAbsolute - 3) * 10 to [0, 30]
      recover longTermRecovery % with 100 % reuse during years 4 to onwards
    end substance

  modify substance "R-404A"
      recover 5 % with 100 % reuse during year 3
      define longTermRecovery as limit (yearAbsolute - 3) * 10 to [0, 30]
      recover longTermRecovery % with 100 % reuse during years 4 to onwards
    end substance

  end application

end policy


start policy "com refrig low-GWP"

  modify application "com refrig"

    modify substance "HFC-134a"
      define level as limit (yearAbsolute - 2) * 20 to [0, 100]
      replace level % of manufacture with "R-600a" during years 3 to onwards
      replace level % of import with "R-600a" during years 3 to onwards
    end substance

    modify substance "R-404A"
      define level as limit (yearAbsolute - 2) * 20 to [0, 100]
      replace level % of manufacture with "R-600a" during years 3 to onwards
      replace level % of import with "R-600a" during years 3 to onwards
    end substance
  end application

end policy


start policy "res AC reuse"

  modify application "res AC"

    modify substance "R-410A"
      recover 5 % with 100 % reuse during year 3
      define longTermRecovery as limit (yearAbsolute - 3) * 10 to [0, 30]
      recover longTermRecovery % with 100 % reuse during years 4 to onwards
    end substance

  modify substance "HFC-32"
      recover 5 % with 100 % reuse during year 3
      define longTermRecovery as limit (yearAbsolute - 3) * 10 to [0, 30]
      recover longTermRecovery % with 100 % reuse during years 4 to onwards
    end substance

  end application

end policy


start policy "res AC low-GWP"

  modify application "res AC"

    modify substance "R-410A"
      define level as limit (yearAbsolute - 2) * 20 to [0, 100]
      replace level % of manufacture with "R-290" during years 3 to onwards
      replace level % of import with "R-290" during years 3 to onwards
    end substance

    modify substance "HFC-32"
      define level as limit (yearAbsolute - 2) * 20 to [0, 100]
      replace level % of manufacture with "R-290" during years 3 to onwards
      replace level % of import with "R-290" during years 3 to onwards
    end substance

  end application

end policy
```

```
start policy "mobile AC reuse"

  modify application "mobile AC"

    modify substance "HFC-134a"
      recover 5 % with 100 % reuse during year 3
      define longTermRecovery as limit (yearAbsolute - 3) * 10 to [0, 30]
      recover longTermRecovery % with 100 % reuse during years 4 to onwards
    end substance

    modify substance "R-1234yf"
      recover 5 % with 100 % reuse during year 3
      define longTermRecovery as limit (yearAbsolute - 3) * 10 to [0, 30]
      recover longTermRecovery % with 100 % reuse during years 4 to onwards
    end substance

  end application

end policy


start policy "mobile AC low-GWP"

  modify application "mobile AC"

    modify substance "HFC-134a"
      define level as limit (yearAbsolute - 2) * 20 to [0, 100]
      replace level % of manufacture with "R-1234yf" during years 3 to onwards
      replace level % of import with "R-1234yf" during years 3 to onwards
    end substance

  end application

end policy


start simulations

  simulate "business as usual" from years 1 to 29

  simulate "dom refrig high ambition"
    using "dom refrig reuse"
    then "dom refrig low-GWP"
  from years 1 to 29
  simulate "com refrig high ambition"
    using "com refrig reuse"
    then "com refrig low-GWP"
  from years 1 to 29

  simulate "res AC high ambition"
    using "res AC reuse"
    then "res AC low-GWP"
  from years 1 to 29

  simulate "mobile AC high ambition"
    using "mobile AC reuse"
    then "mobile AC low-GWP"
  from years 1 to 29

end simulations
```

Note that this translation is literal to demonstrate a one to one mapping with logic from the reference model. That said, some language shortcuts could make this briefer.