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. 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. In either case, 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 realistic full featured examples 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 are likely to be diverse:

- **Depth of modeling**: Range from those seeking a cursory or exploratory projection to experts crafting highly specific or detailed scenarios.
- **Prior expertise**: Span various levels of expertise with regards to substances of interest.
- **Comfort with programming**: May be experienced software developers or may be less comfortable with code even if they have familiarity with programming protal systems like spreadsheet software.

[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.

This diversity guides the design of QubecTalk to allow for positive experiences regardless of background and goals.

1.2 Objectives

QubecTalk serves multiple roles:

- **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 programs 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 examples including an implementation which rebuilds a reference model.

1.3 Modalities

Analyses which may include multiple simulations are described in single file programs which are run through an interpreter. This "engine" is 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.

1.4 License

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: each includes zero or more policies on top of a "business as usual" simulation state. For more information, see the examples.

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 stanza is optional. The hash mark (`#`) denotes comments and the interpreter will ignore anything after a hash mark on a line of code.

2.2 System

QubecTalk allows authors to describe the components of the system to be simulated. These interact with each other inside the interpreter's engine in order to calculate substance consumption, equipment population, and GHG impacts. This engine also dynamically converts between units for author flexibility.

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 and emissions. QubecTalk considers the sales "stream" to fit into two groups ("substreams"): `manufacture` and `import`. Though not currently used, `export` is reserved for future use.

- **Domestic**: The `manufacture` substream refers to substances both produced and consumed domestically. This will count towards emissions.
- **Trade**: Sales of imported substances are tracked under the `import` substream. These are not included in emissions.

Consider the following example snippet:

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

Note the following:

- If not specifying a year, a statement will apply across the entire timeseries.
- If a meta stream like "sales" is specified, a statement will apply proportionally across substreams. For example, in the case of sales above, changes apply proportionally across domestic manufacturing and imports.
- See below for units but absolute volumes, percentages, or ratios can be used in these and similar commands.

While not included in the example snippet and currently not supported, the export substream (`export`) is reserved for future use and would count towards emissions.

2.2.3 Population

New equipment is typically inferred by sales. However, this is not a requirement and the equipment population may be specified manually as well.

Setting equipment: Equipment levels at any year may be specified manually. This could be helpful in specifying historic equipment population levels from before a simulation started.

To specify a new level for equipment not sold in the current year but present in the population, use `priorEquipment`. This will not incur the initial charge sales but will enter those units into the recharge / service pool. This snippet[3] assumes that the historic equipment population is seven times the number of new units sold in the first simulated year:

```
set priorEquipment to (get equipment as units * 7) units during year 1
```

In addition to `priorEquipment`, equipment levels for a year (`equipment`) may be specified manually as well. In this case, the engine assumes that the delta from

[3]The parentheses in this example are added for readability but are optional.

`priorEquipment` to `equipment` represents new equipment sold in the current year.

```
set equipment to 2000 units 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.

```
change sales by +5 % / year
```

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

One may also recharge a number of units by changing `%` to `units` or other volume metric.

2.2.4 Emissions

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

```
emit 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 "example name"
  # ...
end policy
```

Policy names must be unique but cannot include quotes (`"`). In the above example, the policy is given the name `example name`.

2.3.1 Permit / prohibition

Many policies work by placing a limit on streams such as sales.

```
cap emissions to 100 tCO2e
cap sales to 75 % during year 2
cap import to 750 mt during years 4 to 5
```

If specifying an aggregate stream, it will be applied proportionally across sub-streams. In this example snippet, capping `sales` causes the cap to apply across `manufacture` and `import`.

2.3.2 Recycling

Recycling programs are defined as refrigerant recovery from servicing / recharge followed by some loss / yield for reuse.

```
recover 5 % with 100 % reuse during years 3 to onwards
recover 10 % with 100 % reuse during years 4 to 7 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. This is sometimes referred to as "rebound consumption" or "induced demand" but generally describes interventions in which new secondary material does not displace primary material in a 1 to 1 way. If not specified, this command assumes no rebound consumption / induced demand. Note that, while the current implementation does not allow for recovery from retired equipment, this funcitonality is reserved for future use.

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
```

The units given are flexible as with other commands. For example, if providing `mt`, this will convert to equipment units.

2.4 Simulation

A set of zero or more policies defines a scenario. Each `simulate` command creates a new scenario and the order of policies is the order in which they are applied within the engine.

```
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 variables are available.

```
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. QubecTalk does not have a dedicated boolean type.

2.5.3 Uncertainties

Typically uncertainties are applied on demand such as sales growth rates, retirement, and recharge. However, 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[4]:

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

When building these simulations, users should also include a number of trials (`across 100 trials`).

2.5.4 Timeseries

Current simulation time can found in variables:

- `yearsElapsed` for years since the start of the simulation.
- `yearAbsolute` for absolute year number.

These variables can not be set or redefined. Consider the following example snippet:

```
yearAbsolute * 5 + 2
```

These values can be used in any expression. Similar variables are not yet available for other timescales (`monthsElapsed`, `monthAbsolute`, `daysElapsed`, and `dayAbsolute`) but are reserved for future use.

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 commands.

[4]The parentheses in this snippet are added for readability but are optional.

3.1 Supported units

Available units include the following:

- `kg` or `mt`: Volumes as kilograms or megatons
- `tCO2e`: Emissions 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.

3.2 Behavior

Interpretation and unit conversion depends on the command:

Op	Type	Units	Percent	Ratios
Initial Charge	Volume	Volume distributed across population. Emissions converted to volume.	Not supported.	Div by units will multiply by pop size.
Set Sales	Volume	Volume. Emissions converted to volume, units converted to volume. Proportional if not sub-stream.	$x = x * \%$	Div by time multiplies time elapsed since last sim step, div by units multiplies population, div by emissions multiplies emissions.
Set Equipment	Population	Units. Emissions converted to volume, volume converted to units.	$x = x * \%$	Div by time multiplies time elapsed since last sim step, div by volume multiplies volume, div by emissions multiplies emissions.

Op	Type	Units	Percent	Ratios
Set Emissions	Emissions	Emissions distributed across volume streams.	$x = x * \%$	Div by time multiplies time elapsed since last sim step, div by units multiplies population, div by volume converts to div by units.
Change Sales	Volume	Volume delta. Emissions converted to volume, units converted to volume. Proportional if not sub-stream.	$x = x*(1+\%)$	Div by time multiplies time elapsed since last sim step, div by units multiplies population, div by emissions multiplies emissions.
Change Equipment	Population	Units delta. Emissions converted to volume, volume converted to units. Proportional.	$x = x*(1+\%)$	Div by time multiplies time elapsed since last sim step, div by volume multiplies volume, div by emissions multiplies emissions.
Change Emissions	Emissions	Emissions delta distributed across volume.	$x = x*(1+\%)$	Div by time multiplies time elapsed since last sim step, div by units multiplies population, div by volume converts to div by units.
Retire	Population	Emissions converted to volume, volume converted to units.	$r = x * \%$	Div by time multiplies time elapsed since last sim step, div by volume multiplies volume, div by emissions multiplies emissions.

Op	Type	Units	Percent	Ratios
Cap Sales	Volume	Volume. Emissions converted to volume, units converted to volume. Proportional if not sub-stream.	$c = x * \%$	Div by time multiplies time elapsed since last sim step, div by units multiplies population, div by emissions multiplies emissions.
Cap Equipment	Population	Units. Emissions converted to volume, volume converted to units.	$c = x * \%$	Div by time multiplies time elapsed since last sim step, div by volume multiplies volume, div by emissions multiplies emissions.
Cap Emissions	Emissions	Emissions total distributed across volume streams.	$c = x * \%$	Div by time multiplies time elapsed since last sim step, div by units multiplies population, div by volume converts to div by units.
Recycle	Volume	Volume. Emissions converted to volume, units converted to volume. Proportional.	Uniform across streams.	Div by time multiplies time elapsed since last sim step, div by units multiplies population, div by emissions converts to div by units.
Replace Sales	Volume	Volume. Emissions converted to volume, units converted to volume. Proportional if not sub-stream.	Uniform across streams.	Div by time multiplies time elapsed since last sim step, div by units multiplies population, div by emissions converts to div by units.

Op	Type	Units	Percent	Ratios
Replace Equipment	Volume	Converted to volume.	Uniform across streams.	Div by time multiplies time elapsed since last sim step, div by volume converts to units, div by emissions converts to div by volume.
Replace Emissions	Volume	Converted to volume.	Uniform across streams.	Div by time multiplies time elapsed since last sim step, 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 Emissions	Emissions	Emissions (`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 between emissions and volumes with initial charge values. Furthermore, while alternative simulation time steps are reserved for future use, the time step ("time elapsed since last sim step") is always 1 year even in the first year. 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

This section describes engine internal logic as changes in one stream impact others. Most users of QubecTalk will not need to understand these details and can skip to the examples below. However, this behavioral description is for completeness.

4.1 Triggers

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. The following dependencies see changes made at the start of the chain cause changes downstream. Note that these are not transitive beyond this table to avoid circular triggers.

Trigger	**Sales**	**Emissions**	**Population**
Sales (`manufacture`, `import`)	Changed directly	Recalc using emit value	Recalc new and total equip after subtract recharge and recycle, retiring at recharge if needed.
Emissions (`emissions`)	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 emit	New equip and total current equip changes but prior equip untouched.
Prior Population (`priorEquipment`)	No change	No change	Change prior equip. New equipment "delta" updated to maintain original total.

These updates happen automatically within the interpreter. For more information, see the trigger response logic below.

4.2 Trigger response logic

Downstreams are recalculated based on author-specified upstream changes as described in this section.

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, negative total populations are treated as zero (interpret as net exports).

4.2.2 Emissions response

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

- Ignore imports and secondary material within sales.
- Convert to emissions.
- Set negative total emissions to zero (treat net negative emissions as credit taken elsewhere).

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

4.2.3 Sales response

Sales changes are propogated to imports and domestic manufacturing proportionally. In any case, changes to either `sales` or its substreams (`manufacture`, `import`) cause new equipment deployment to be recalculated after accounting for recharge.

- Determine needs for recharge.
- Determine new equipment deployment delta.
- Offset with recycling.
- If needed, update import and domestic sales proportionally.

Note that negative sales are assumed as exports.

4.3 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	`emit`
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.

5 Examples

The following offer demonstrative examples of QubecTalk. These are also used in automated tests of the language's interpreter.

5.1 Basic simulations

Before looking at policies, examples start with a single business as usual scenario to highlight language basics.

5.1.1 Single substance

This simplistic introductory example defines a single substance.

```
start default

  define application "test"

    uses substance "test"
      initial charge with 5 kg / unit for manufacture
      set manufacture to 100 mt
      emit 5 tCO2e / mt
    end substance

  end application

end default


start simulations

  simulate "business as usual" from years 1 to 1

end simulations
```

This runs for 1 year and expects 20000 units, 500 tCO2e, 100000 kg maufacture.

5.1.2 Change

Though changes can be specified in various units, this example shows a simple percent change per year.

```
start default

  define application "test"

    uses substance "test"
      initial charge with 1 kg / unit for manufacture
```

```
      set manufacture to 100 mt
      change manufacture by +10 % / year
      emit 5 tCO2e / mt
    end substance

  end application

end default


start simulations

  simulate "business as usual" from years 1 to 2

end simulations
```

This runs for 2 years and expects output of 550 tCO2e in year 2.

5.1.3 Retire

QubecTalk removes equipment through "retirement" which takes units out of the deploymed population.

```
start default

  define application "test"

    uses substance "test"
      initial charge with 1 kg / unit for manufacture
      set manufacture to 100 mt
      retire 10 %
      emit 5 tCO2e / mt
    end substance

  end application

end default


start simulations

  simulate "business as usual" from years 1 to 2

end simulations
```

This example runs for 2 years and expects output of 450 tCO2e in year 2.

5.1.4 Recharge

Servicing applies a recharge, consuming some substance sales.

```
start default

  define application "test"

    uses substance "test"
      set manufacture to 100 mt
      initial charge with 1 kg / unit for manufacture
      recharge 10% with 1 kg / unit
      emit 5 tCO2e / mt
    end substance

  end application

end default


start simulations

  simulate "business as usual" from years 1 to 2

end simulations
```

This example results in 190000 units total across years 1 and 2.

5.1.5 Multiple emissions

This next snippet shows combining emissions for multiple substances.

```
start default

  define application "test"

    uses substance "a"
      initial charge with 1 kg / unit for manufacture
      set manufacture to 100 mt
      emit 5 tCO2e / mt
    end substance

    uses substance "b"
      initial charge with 1 kg / unit for manufacture
      set manufacture to 100 mt
      emit 10 tCO2e / mt
    end substance
```

```
  end application

end default


start simulations

  simulate "business as usual" from years 1 to 1

end simulations
```

This example runs for 1 year and expects output of 1500 tCO2e.

5.2 Policies

Individual or combined policies can run in scenarios created through the `simulate` command.

5.2.1 Cap

This example applies a simple percentage cap.

```
start default

  define application "test"

    uses substance "test"
      initial charge with 1 kg / unit for manufacture
      set manufacture to 100 mt
      emit 5 tCO2e / mt
    end substance

  end application

end default


start policy "intervention"

  modify application "test"

    modify substance "test"
      cap manufacture to 50%
    end substance

  end application
```

```
end policy


start simulations

  simulate "result" using "intervention" from years 1 to 1

end simulations
```

This runs for 1 year and expects output of 250 tCO2e.

5.2.2 Recycling

This next simple recycling scheme assumes no rebound consumption.

```
start default

  define application "test"

    uses substance "test"
      initial charge with 2 kg / unit for manufacture
      set manufacture to 100 mt
      recharge 50% with 1 kg / unit
      emit 5 tCO2e / mt
    end substance

  end application

end default


start policy "intervention"

  modify application "test"

    modify substance "test"
      recover 50 % with 100 % reuse
    end substance

  end application

end policy


start simulations

  simulate "result" using "intervention" from years 1 to 2
```

```
end simulations
```

This example results in a total of 937.5 tCO2e across years 1 and 2.

5.2.3 Replace

This example demonstrates a simple transition between compatible substances.

```
start default

  define application "test"

    uses substance "a"
      initial charge with 1 kg / unit for manufacture
      set manufacture to 50 mt during year 1
      emit 10 tCO2e / mt
    end substance

    uses substance "b"
      initial charge with 1 kg / unit for manufacture
      set manufacture to 50 mt during year 1
      emit 5 tCO2e / mt
    end substance

  end application

end default


start policy "intervention"

  modify application "test"

    modify substance "a"
      replace 50 % of manufacture with "b"
    end substance

  end application

end policy


start simulations

  simulate "result" using "intervention" from years 1 to 1
```

```
end simulations
```

In year 1, this example results in 625 tCO2e.

5.2.4 Combination

This next example demonstrates policy stacking.

```
start default

  define application "test"

    uses substance "test"
      initial charge with 1 kg / unit for manufacture
      set manufacture to 100 mt during year 1
      emit 5 tCO2e / mt
    end substance

  end application

end default


start policy "a"

  modify application "test"

    modify substance "test"
      cap manufacture to 50%
    end substance

  end application

end policy


start policy "a"

  modify application "test"

    modify substance "test"
      cap manufacture to 50%
    end substance

  end application

end policy
```

```
start simulations

  simulate "result" using "a" then "b" from years 1 to 1

end simulations
```

This example results in 250 tCO2e in year 1.

5.3 Features

The following examples show individual language features.

5.3.1 Conditional and variable

This example demonstrates use of a conditional statement.

```
start default

  define application "test"

    uses substance "test"
      initial charge with 1 kg / unit for manufacture
      define testVar as 5
      set manufacture to 100 if testVar > 10 else 50 mt endif
      emit 5 tCO2e / mt
    end substance

  end application

end default


start simulations

  simulate "business as usual" from years 1 to 1

end simulations
```

This runs for one year and results in 250 tCO2e.

5.3.2 Monte Carlo

This example runs a simple Monte Carlo.

```
start default
```

```
  define application "test"

    uses substance "test"
      initial charge with 1 kg / unit for manufacture
      set manufacture to 100 mt
      emit sample uniformly from 3 to 7 tCO2e / mt
    end substance

  end application

end default


start simulations

  simulate "business as usual" from years 1 to 1 across 100 trials

end simulations
```

This will run for 1 year and results in emissions between 300 to 700 tCO2e.

5.4 Reference model

This final example translates a reference model into QubecTalk.

```
start about
  # Name: "Basis for Models"
  # Description: "Translation of in our business BN3."
  # Author: "A Samuel Pottinger"
  # Date: "2024-09-20"
  # Notes: "Based on work from Balaji Natarajan."
end about


start default

  define application "dometic refrigeration"

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

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

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

    # Service
    retire 6.7 % / year
    recharge 10 % / 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"
    emit 6 tCO2e / mt

    # Domestic production
    initial charge with 0.05 kg / unit for manufacture
    set manufacture to 200000 mt during year 1
    change manufacture by +8 % / year

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

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

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

  uses substance "low gwp"
    emit 0 tCO2e / mt
  end substance

end application

define application "commercial refrigeration"

  uses substance "HFC-134a"
    emit 1430 tCO2e / mt
```

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

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

  # Service
  retire 6.7 % / year
  recharge 10 % / 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"
  emit 6 tCO2e / mt

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

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

  # Service
  retire 6.7 % / year
  recharge 10 % / 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"
  emit 3922 tCO2e / mt

  # Domestic production
  initial charge with 0.30 kg / unit for manufacture
```

```
    set manufacture to 30000 mt during year 1
    change manufacture by +5 % / year

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

    # Service
    retire 6.7 % / year
    recharge 10 % / 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 "low gwp"
    emit 0 tCO2e / mt
  end substance

end application

define application "residential AC"

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

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

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

    # Service
    retire 6.7 % / year
    recharge 10 % / 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"
  emit 632 tCO2e / mt

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

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

  # Service
  retire 6.7 % / year
  recharge 10 % / 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"
  emit 6 tCO2e / mt

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

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

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

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

uses substance "low gwp"
  emit 0 tCO2e / mt
end substance
```

```
end application

define application "mobile AC"

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

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

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

    # Service
    retire 6.7 % / year
    recharge 10 % / 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"
    emit 6 tCO2e / mt

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

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

    # Service
    retire 6.7 % / year
    recharge 10 % / 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 "low gwp"
      emit 0 tCO2e / mt
    end substance

  end application

end default


start policy "domestic refrigeration prohibition"

  modify application "dometic refrigeration"

    modify substance "HFC-134a"
      cap manufacture to 75 % during year 3
      cap manufacture to 40 % during year 4
      cap manufacture to 0 % during years 5 to onwards

      cap import to 75 % during year 3
      cap import to 40 % during year 4
      cap import to 0 % during years 5 to onwards
    end substance

  end application

end policy


start policy "domestic refrigeration reuse"

  modify application "dometic refrigeration"

    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 "domestic refrigeration low-GWP"

  modify application "dometic refrigeration"

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

  end application

end policy


start policy "commercial refrigeration prohibition"

  modify application "commercial refrigeration"

    modify substance "HFC-134a"
      cap manufacture to 0 % during years 3 to onwards
      cap import to 0 % during years 3 to onwards
    end substance

    modify substance "R-404A"
      cap manufacture to 0 % during years 3 to onwards
      cap import to 0 % during years 3 to onwards
    end substance

  end application

end policy


start policy "commercial refrigeration reuse"

  modify application "commercial refrigeration"

    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 "commercial refrigeration low-GWP"

  modify application "commercial refrigeration"

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

  end application

end policy


start policy "residential AC prohibition"

  modify application "residential AC"

    modify substance "R-410A"
      cap manufacture to 0 % during years 3 to onwards
      cap import to 0 % during years 3 to onwards
    end substance

    modify substance "HFC-32"
      cap manufacture to 0 % during years 18 to onwards
      cap import to 0 % during years 18 to onwards
    end substance

  end application
```

```
end policy


start policy "residential AC reuse"

  modify application "commercial refrigeration"

    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 "residential AC low-GWP"

  modify application "commercial refrigeration"

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

  end application

end policy


start policy "mobile AC prohibition"
```

```
  modify application "mobile AC"

    modify substance "HFC-134a"
      cap manufacture to 0 % during years 3 to onwards
      cap import to 0 % during years 3 to onwards
    end substance

  end application

end policy


start policy "mobile AC reuse"

  modify application "commercial refrigeration"

    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 "commercial refrigeration"

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

  end application
```

```
end policy


start simulations

  simulate "business as usual" from years 1 to 29

  simulate "domestic refrigeration high ambition"
    using "domestic refrigeration prohibition"
    then "domestic refrigeration reuse"
    then "domestic refrigeration low-GWP"
  from years 1 to 29

  simulate "commercial refrigeration high ambition"
    using "commercial refrigeration prohibition"
    then "commercial refrigeration reuse"
    then "commercial refrigeration low-GWP"
  from years 1 to 29

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

  simulate "mobile AC high ambition"
    using "mobile AC prohibition"
    then "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 more concise.