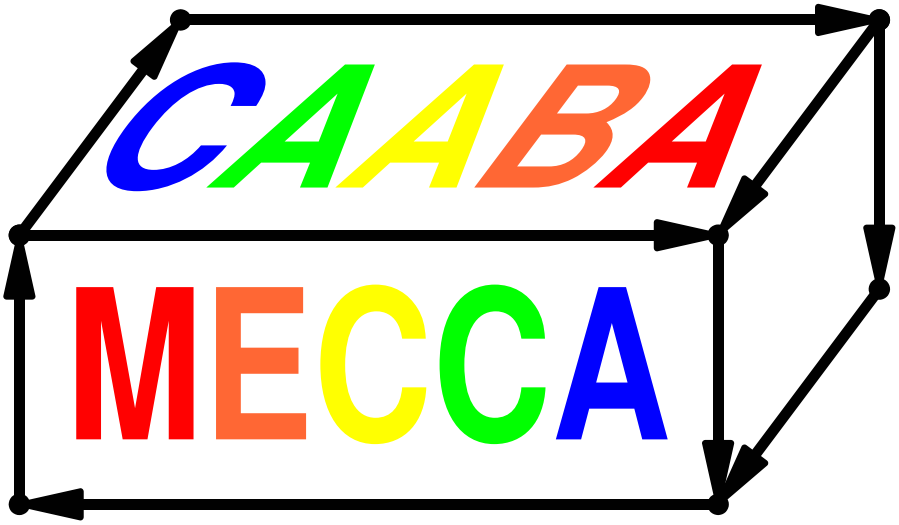

CAABA/MECCA-2.7 User Manual

Chemistry As A Boxmodel Application /
Module Efficiently Calculating the
Chemistry of the Atmosphere

Rolf Sander, Hartwig Harder, Patrick Jöckel, Astrid Kerkweg & Hella Riede

Air Chemistry Department
Max-Planck Institute of Chemistry
PO Box 3060, 55020 Mainz, Germany
sander@mpch-mainz.mpg.de

www.mpch-mainz.mpg.de/~sander/messy/mecca/

Date: 2010-08-09

Contents

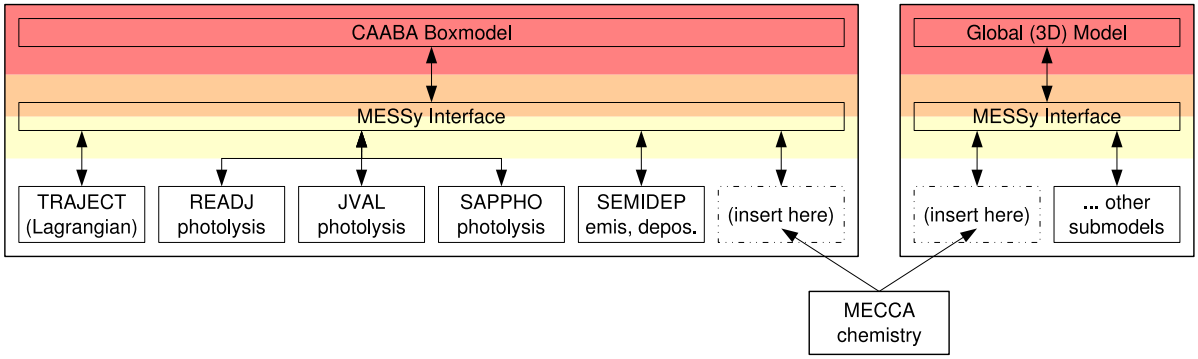

Figure 1: Diagram showing MECCA as part of the CAABA box model or of a global model.

1 Introduction

MECCA (Module Efficiently Calculating the Chemistry of the Atmosphere) is an atmospheric chemistry module that contains a comprehensive chemical mechanism with tropospheric and stratospheric chemistry of both the gas and the aqueous phase (Sander et al., 2005, 2009). For the numerical integration, MECCA uses the KPP software (Sandu and Sander, 2006).

To apply the MECCA chemistry to atmospheric conditions, MECCA must be connected to a base model. As shown in Fig. 1, the base model can be a complex, 3-dimensional model (e.g. Jöckel et al., 2006) but it can also be a simple box model. The connection is established via the MESSy interface (`http://www.messy-interface.org`) developed by Jöckel et al. (2005).

This manual describes how to install and work with MECCA when it is connected to the box model CAABA (Chemistry As A Boxmodel Application). This combination will be referred to as "CAABA/MECCA". The main features of the CAABA box model are shown in Fig. 2. In addition to MECCA chemistry, CAABA also contains several other submodels, e.g. JVAL for calculating J-values, SAPPHO for simplified and parameterized photolysis rates, and SEMIDEP for simplified emission and deposition.

2 Installation

This section can be skipped if CAABA/MECCA is already installed on your computer.

2.1 System Requirements

2.1.1 Linux/Unix

CAABA/MECCA has been tested successfully on several UNIX-like operating systems. The easiest installation is probably on a Linux PC since several auxiliary programs are already included in a typical Linux distribution.

2.1.2 MAC OS X

CAABA/MECCA does not work with the version of the `sed` program that is shipped with MAC OS X. Instead, it is necessary to install the GNU version of `sed`, called `gsed`. This can be done using the MacPorts (`http://www.macports.org/`). To ensure that `gsed` is always executed when `sed` is called, a symbolic link from `sed` to `gsed` can be created, e.g.:

```
sudo ln -s /opt/local/bin/gsed
  /opt/local/bin/sed
```

2.1.3 Windows

A native installation under Windows is neither recommended nor supported. However, it is possible to run the model in a virtual Linux machine on a Windows PC.

((ADD MORE TEXT BASED ON INFO FROM HARTWIG HERE))

2.2 Prerequisites

A Fortran90 compiler (mandatory): Several compilers have been tested successfully: g95 (for Linux), Lahey (for Linux), Intel (for Linux), Compaq (Alpha UNIX). Other compilers can be used as well if they accept standard Fortran90

code. It should be noted that the g95 compiler for Linux is free and can be downloaded from `http://www.g95.org/`.

The Kinetic PreProcessor KPP (mandatory): This flexible numerical integration package by Sandu and Sander (2006) transforms the chemistry mechanism into a set of ordinary differential equations (ODEs) in Fortran90 syntax. MECCA needs the KPP version that is provided in the `mecca/kpp/` directory. Follow the instructions in `mecca/kpp/readme` to install KPP.

Perl, tcsh, awk, sed, and make (mandatory): These UNIX tools are standard on Linux systems. Check that recent versions of them are installed. Especially awk may lead to strange error messages. To test awk, type:
`awk 'BEGIN {print match("X","[^a-z]")}'`
The result should be "1". However, you may get "0" as the result on your system. Supposedly, this is not a bug in awk but a feature. You can solve the problem by setting the environment variable `LC_ALL` to "`C`":
`export LC_ALL=C` (if you use bash)
`setenv LC_ALL C` (if you use tcsh)
When you try the awk test again, it should work fine.

LaTeX (optional): If you have LaTeX installed on your computer, you can print a table (including rate coefficients and references) of the currently selected mechanism (see Sect. 7.3.3 for details).

netCDF library (optional): The netCDF library is needed to create model output in netCDF format. It can be obtained from `http://www.unidata.ucar.edu/software/netcdf/`. Note that several files in the netCDF library are compiler-specific. Thus, it is necessary to create a netCDF library for each compiler and maybe also for each compiler version.

Software for manipulating or displaying netCDF data is listed at: `http://www.unidata.ucar.edu/software/netcdf/software.html`. If you don't have the netCDF library, you can still run the model but produce only ASCII output.

netCDF tools (optional): Several tools are used to analyse the netCDF output when the model is run in the multirun (Sect. 6.1) or Monte-Carlo (Sect. 6.2) mode. Specifically, the NCO programs `ncpdq`, `ncrcat`, and `ncks` from `http://nco.sourceforge.net` and the program `ncclamp` from `http://ncclamp.sourceforge.net` are needed.

ferret (optional): The ferret plotting program is needed to plot the contents of the netCDF output using the ferret scripts in the `jnl/` directory (see Sect. 5 for details). To ensure that ferret finds all necessary files, you have to add "`./tools`" to the `FER_GO` environment variable. The `ferret_paths*template` files show how to do this. For example, when using the tcsh, type:
`setenv FER_GO "$FER_GO ./tools"`

2.3 Installation

Once all prerequisites are fulfilled, you can install CAABA/MECCA by simply unpacking the zip archive:

```
unzip caaba_2.7.zip
```

Next, you have to check that all settings in `Makefile` are correct. If necessary, edit the file: Choose a Fortran90 compiler (`COMPILER`), enter its name (`F90`) and the compiler options (`F90FLAGS`). If you add a new compiler, be sure to activate the C-preprocessor option. To activate netCDF output, you also have to edit the `Makefile`:

- Change the variable `OUTPUT` from `ASCII` to `NETCDF`.
- Enter the correct netCDF library information in `NETCDF_INCLUDE` and `NETCDF_LIB`.

2.4 Troubleshooting

Should there be any problems with the CAABA/MECCA installation, please check the following:

- Confirm that all prerequisites (see above) are fulfilled!
- Confirm that the perl path in the first line of `sfmakedepend` is correct. It should be the same as the output of the command:
  `which perl`
- Confirm that the tcsh paths in the first lines of `xcaaba` and `xmecca` are correct.
- Confirm that the model code was unzipped successfully from the zip archive. Check for potential problems during the unzipping process:
  - Make sure that the directory structure has not changed. Unfortunately, some unzipping programs seem to put all files into one directory, ignoring the original directory structure.

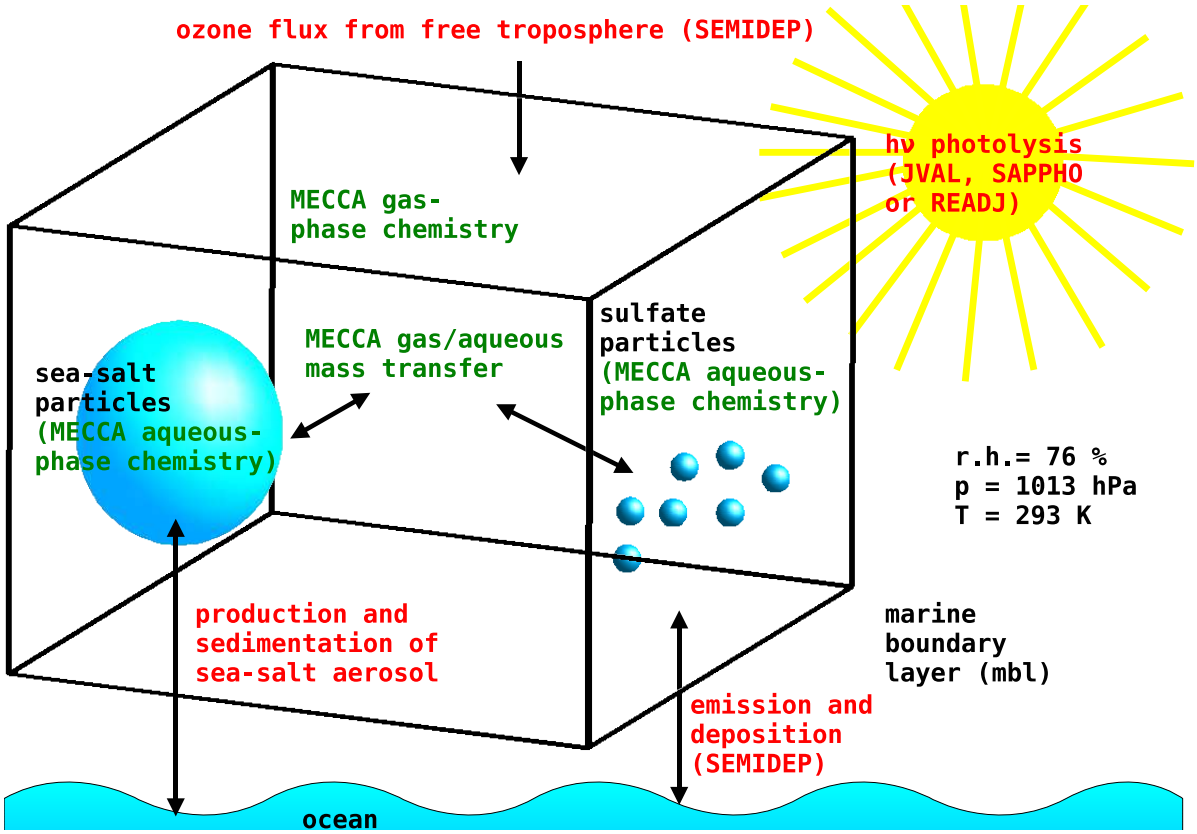

Figure 2: The CAABA box model

– Make sure that links have not been converted to files. For example, the output of the command "`file caaba.nml`" should tell you that `caaba.nml` is a symbolic link to `nml/simple/caaba.nml`.

3 Compiling and running the CAABA/MECCA box model with the shell script `xcaaba`

First, go to the base directory of the model code (note that all path names given in this manual are relative to this base directory):

```
cd caaba_2.7
```

Next, the tcsh script `xcaaba` will guide you through the process of running the box model. To execute it, type:

```
./xcaaba
```

`xcaaba` will ask several questions, and recommended answers are given below. If you only press the Return key, you select the default.

```
Start xmecca?
```

If you answer "y", you can create a new chemical mechanism with `xmecca` as described in detail in Sect. 4. However, for the first tests with CAABA/MECCA it is recommended to answer "n" and use the simple default mechanism.

```
Choose an option:
s = Start from scratch
c = Compile
r = Run existing executable
h = Help
q = Quit
```

Choose "c" to compile the Fortran90 code. After a successful compilation, `xcaaba` asks if you want to run the model:

```
Run CAABA boxmodel with MECCA chemistry?
y = Yes (default)
n = No
m = Multirun (multiple runs)
q = Quit
```

Answer "y" here (see Sect. 6.1 for more information about the "m" = multirun option).

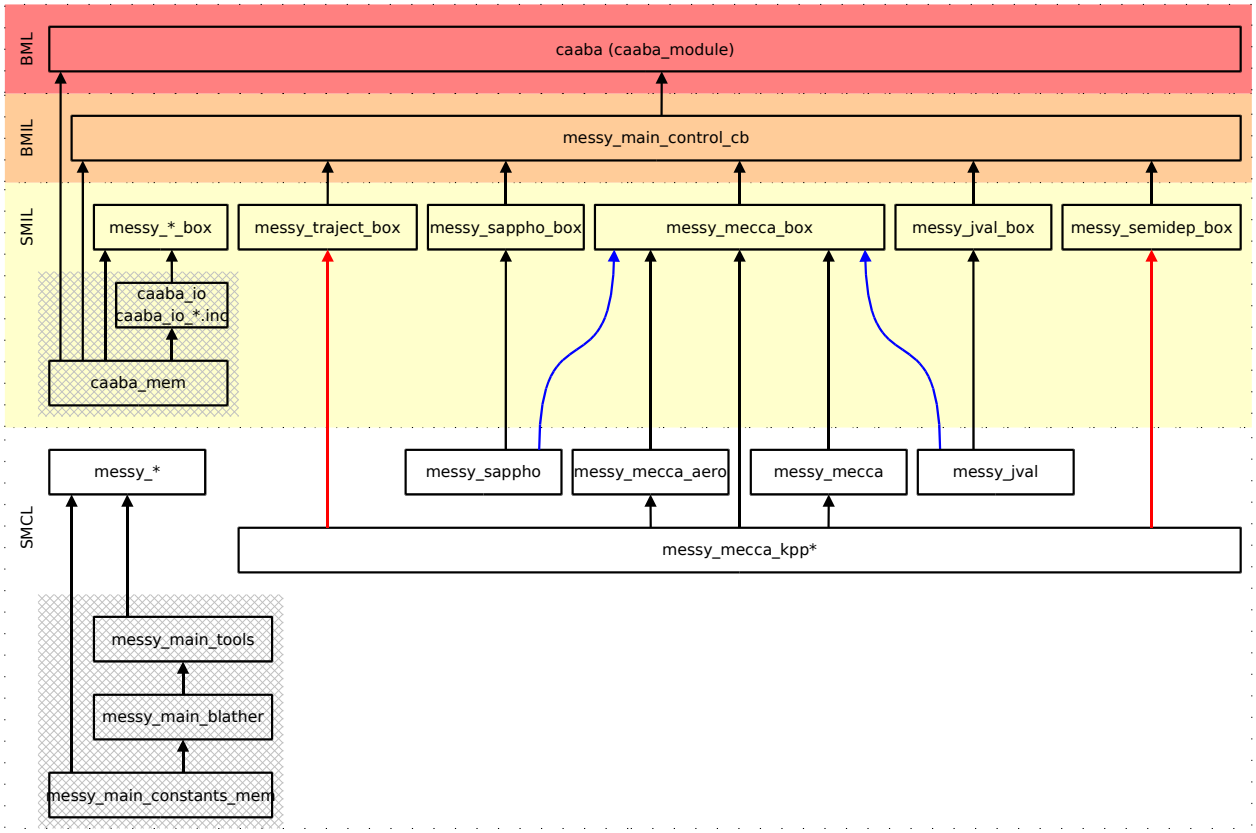

Figure 3: Module structure of MECCA when it is connected to the CAABA box model. The box-model related files are in the colored layers marked with BML, BMIL, and SMIL. The submodel core layer (SMCL) of MECCA is independent of the box model (see Jöckel et al. (2005) for details about the MESSy layers). The arrows start at the module which is exporting the variables and subroutines. They point to the module importing them via the Fortran90 USE instruction. Here, the box `messy_mecca_kpp*` represents all KPP-generated files. The KPP-internal structure is shown in Fig. 4.

```
Choose a namelist file from
the nml/ directory:
...
```

Namelists control the behaviour of CAABA/MECCA during run-time, and editing them allows fine-tuning of the model simulation (see Sect. 7.1). The default is to use the same namelist as last time. For the first tests, the file `simple/caaba.nml` can be chosen. Next, `xcaaba` shows the active contents of the namelist files `caaba.nml` and `mecca.nml`.

```
Run CAABA boxmodel with MECCA chemistry?
```

Answer "y", and the CAABA/MECCA model simulation will start. The flow control is illustrated in Fig: 6. The model day and the current solar zenith angle (sza) are printed on the screen during the model simulation. The default is to integrate 8 days.

```
Save the output and model code
in output/ directory?
```

Answer "y", and `xcaaba` will put the files into a subdirectory with a name based on the date and time of the model simulation, e.g. `output/2009-08-24-16:29:00/`.

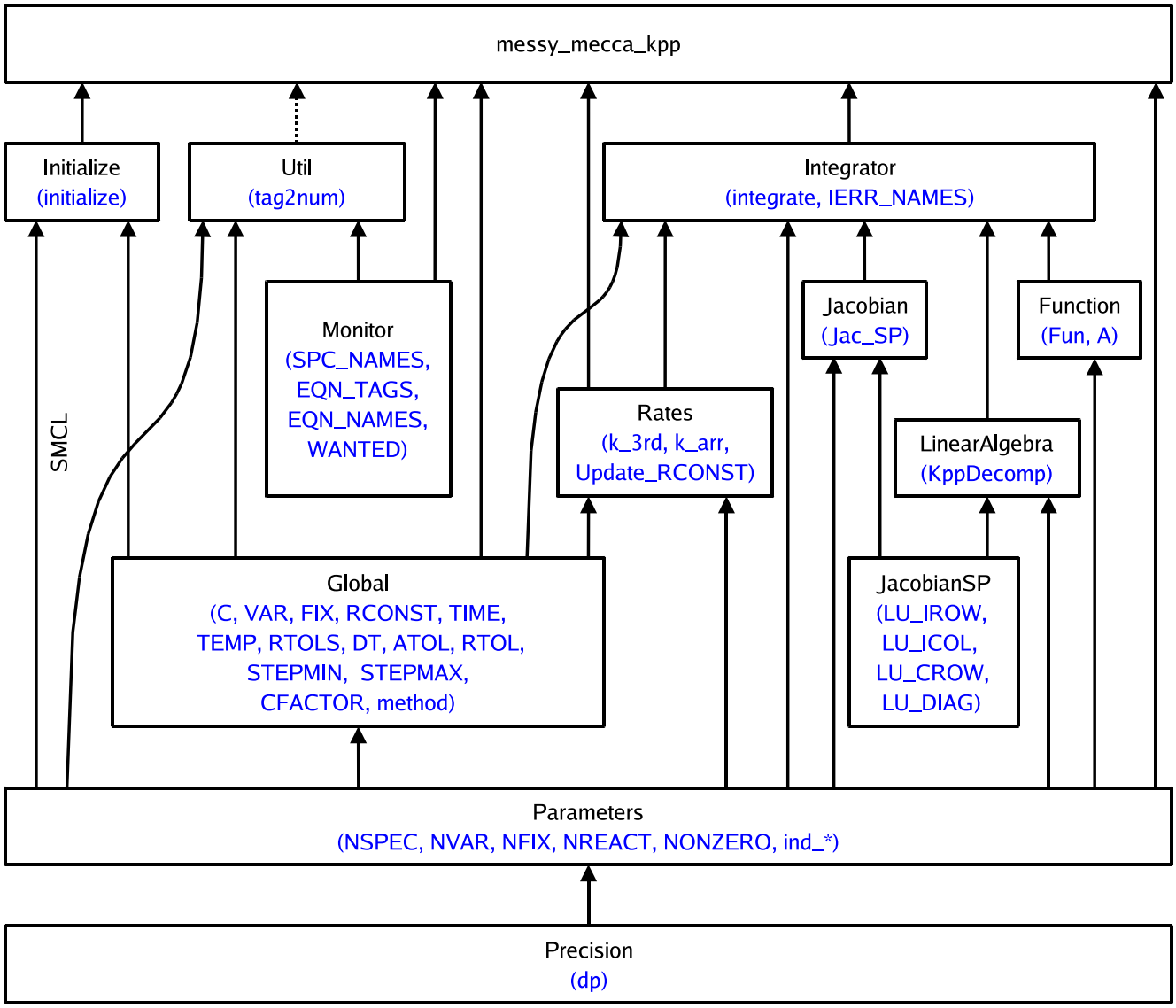

Figure 4: Module structure of KPP-produced Fortran90 files. The arrows start at the module which is exporting the variables and subroutines shown in blue. They point to the module importing them via the Fortran90 USE instruction.

4 Selecting a chemical mechanism with the shell script `xmecca`

MECCA contains a very comprehensive set of chemical reactions in both the gas phase and the aqueous phase. For many applications, using the complete mechanism will consume too much CPU time. Therefore, the shell script `xmecca` has been written which allows to create a custom-made subset of the chemical mechanism interactively. Normally, `xmecca` is called via `xcaaba`. However, you can also start it manually:

```
cd mecca
./xmecca
```

`xmecca` will ask several questions, and recommended answers are given below. If you only press the Return key, you select the default.

```
How many aerosol phases?
```

For a gas-phase only mechanism, type "0". For a mechanism with aqueous-phase chemistry in seasalt and in sulfate particles, type "2". Other values are possible if they have been defined in subroutine `define_aerosol` in `messy_mecca_box.f90`.

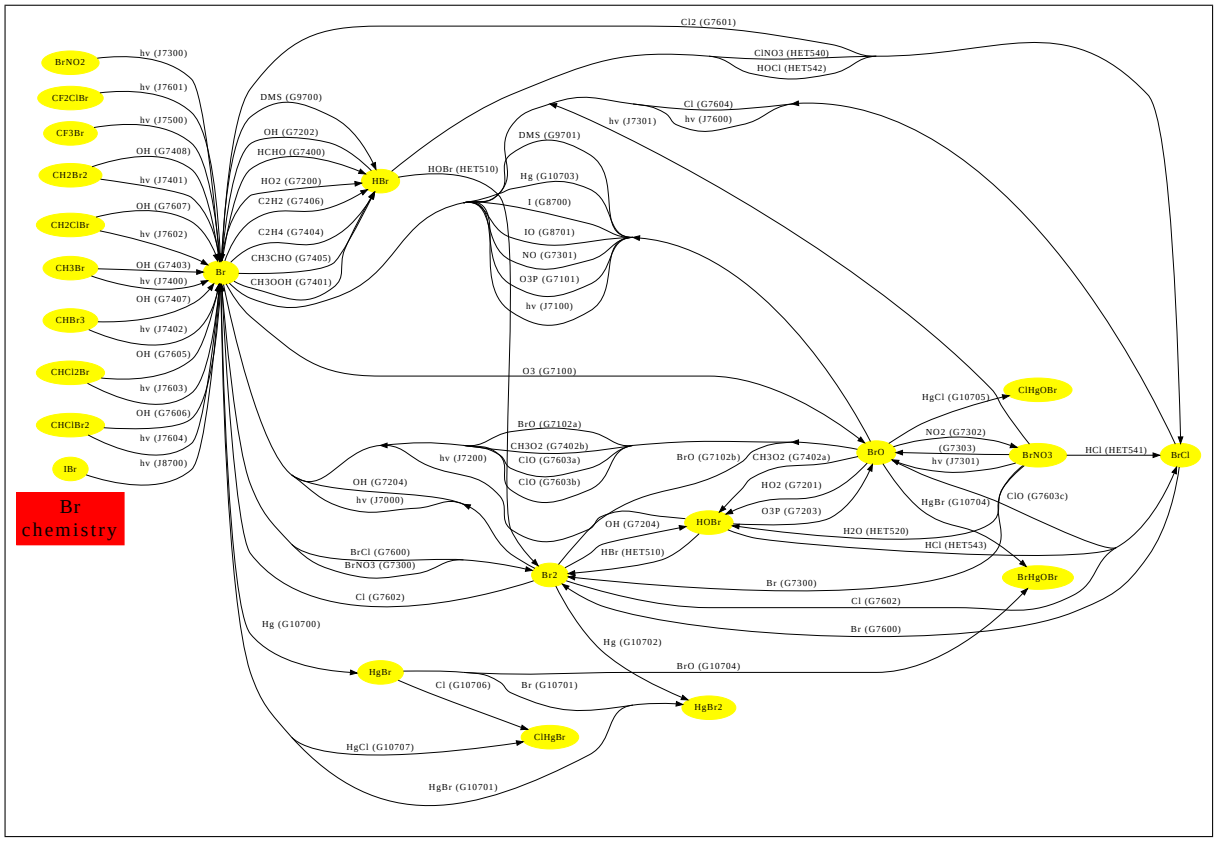

Figure 5: Visualization of the MECCA gas-phase bromine chemistry generated with the graphviz software.

`Modify gas.eqn with a replacement file?`

Answer "0" unless you have written your own replacement file. More information about the replacement feature can be found in the file `rpl/gas.rpl-example`.

`Choose a selection number or type a boolean expression:`

Now you can choose a subset of chemical reactions. A few predefined standard selections are available. For all other purposes, a batch file should be created, as explained at the end of this section. Some of the predefined selections are:

EVAL: A mechanism that was used for the evaluation of the MECCA chemistry in the global model ECHAM5/MESSy (Jöckel et al., 2006).

Minimum tropospheric chemistry: A very small tropospheric mechanism.

Minimum MBL chemistry: A small mechanism that contains aqueous-phase chemistry and should only be used if the number of aerosol phases is > 0.

For details about the selection, see Sect. 4.1.

`Add Monte-Carlo factor to all rate coefficients? [y/n/q, default=n]`

Answer "n" here unless you want to perform Monte-Carlo calculations, as described in Sect. 6.2.

`Add diagnostic tracers to gas.eqn? [q/0/?, default=0]`

Answer "0".

`Calculate accumulated reaction rates of all equations? [y/n/q, default=n]`

Answer "y" if you want to have all accumulated reaction rates in the model output. Otherwise, answer "n".

`Tagging, doubling, both, or none? [t/d/b/n/q, default=n]`

Tagging is under construction, so please answer "n".

`Run KPP?`

Answer "y".

```
Choose an integrator [q=quit,
default=rosenbrock_posdef]:
```

The default integrator is strongly recommended (see Sect.4.2 for details). Next, KPP will create several Fortran90 files.

```
Remove indirect indexing with decomp?
[y/n/q, default=n]
```

If this question shows up, answer "n".

```
Create LaTeX listing of selected mechanism?
[y/n/q, default=n]
```

If you answer "y" here, a table of the current reaction mechanism will be produced. Only the selected reactions will be listed. The table also contains the rate coefficients and their references, as described in Sect. 7.3.3.

```
Create graphviz plots of selected
mechanism? [y/n/q, default=n]
```

If you have the "`dot`" program from the graphviz software installed, you can create graphical visualizations of the reaction mechanism. As an example, the graphviz-generated plot of gas-phase bromine chemistry is shown in Fig. 5. For more information, look at the files `xgraphvizall`, `xgraphviz`, and `spc_extract.awk` in the `mecca/graphviz/` directory.

```
Do you want to delete the temporary
xmecca files?
```

It is okay to delete these temporary files unless you need them for debugging purposes.

When `xmecca` finishes successfully, the Fortran90 code of your selected mechanism has been created. The KPP-produced Fortran90 files (Tab. 1) are moved into the `mecca/smcl/` directory (with lower-case names). An exception is `messy_mecca_kpp_Model.f90`, which is produced by KPP but not needed for MECCA. The modular structure of the KPP-produced Fortran90 files is shown in Fig. 4.

If you need to create a chemical mechanism very often, it is quite tedious to answer all questions every time. To make this easier, you can copy the template `batch/example.bat` to a new name (e.g. `batch/myfile.bat`) and then enter your answers into that batch file. Now you can create a new chemical mechanism in batch mode with

```
./xmecca myfile
```

It is also possible to add the name of the batch file to the `xcaaba` command:

```
./xcaaba myfile
```

4.1 Selecting a set of chemical reactions

All chemical reactions are marked. Each marker consists of several labels which contain information about the altitude (troposphere/stratosphere), the phase where the reaction occurs (gas/aqueous), its relevant chemical elements, and more. See Sect. 7.3.2 for a complete list of labels. To define a set of chemical reactions, you can either choose a pre-defined selection by number or enter a boolean expression based on the labels. Boolean expressions are typed in awk syntax. The most important operators and expressions are:

`&&` = AND
`||` = OR
`!` = NOT
`()` = parentheses
`1` = TRUE
`0` = FALSE

For example, to select all gas-phase reactions (G) except for those including halogens (Cl, Br, I), type:

`G && !Cl && !Br && !I`.

It is important to understand the logic behind this selection mechanism. The expression "`Cl && Br`" selects only those reactions that contain chlorine *and* bromine. Similarly, the expression "`G && Het`" selects only those reactions that occur in the gas phase *and* and are heterogeneous. However, since no reaction has both the "`G`" *and* the "`Het`" label, this results in an empty mechanism. If you want a mechanism that contains both gas-phase and heterogeneous reactions, you must select all reactions that contain either the label "`G`" *or* the label "`Het`", i.e. you must use the expression "`G || Het`".

4.2 Selecting a numerical integrator

Several numerical integrators are defined in the subdirectory `mecca/kpp/int/` and can be used with KPP. The default is the positive definite Rosenbrock solver with automatic time-step control (`rosenbrock_posdef`). It is very robust and capable of integrating very stiff sets of equations (e.g. chemical mechanisms including both gas- and aqueous-phase chemistry). Although a Rosenbrock solver with manual time-step control (`ros2_manual`) is also available, it is strongly recommended not to use it for stiff sets of equations. If you choose it, you do so at your own risk!

5 Plotting the model results with the ferret software

If you have chosen netCDF output, you can plot the model results with the ferret program (`http:`

//ferret.wrc.noaa.gov/Ferret/). Change into the jnl/ directory, then start the program by typing "ferret". When ferret has started, you can plot the gas-phase species of the latest model simulation with the ferret script xxxg.jnl by typing:

```
go xxxg.jnl
```

Similarly, xxxa.jnl can be used to plot aqueous-phase species:

```
go xxxa.jnl
```

The file xxxa.jnl accepts several parameters to modify the plots. The first parameter should be "0d" for plotting box model results. The second parameter can be set to "mpl" or "mpm" in order to plot either aqueous-phase concentrations [mol/L] or mixing ratios [mol(aq)/mol(air)], respectively. The third parameter defines the aerosol bin. With two aerosol bins, "A01" refers to sulfate particles, and "A02" to sea-salt particles. For example, type:

```
go xxxa.jnl 0d mpl A02
```

Photolysis rate coefficients can be plotted with jval.jnl:

```
go jval.jnl
```

If the calculation of accumulated reaction rates had been switched on in xmecca (see Sect. 4), plots of the reaction rates can be made. One possibility is to plot all reactions with:

```
go rxnrates.jnl
```

Alternatively, it is possible to plot only the production and destruction rates for a certain species, e.g. for OH:

```
go rxnrates_scaled.jnl OH
```

To plot results from previous simulations which are saved in the output/ directory, edit the file setmodelrun.jnl and enter the paths of the directories in the "GO _define_sensi" command. To compare model simulations, you can enter two or more "GO _define_sensi" commands in setmodelrun.jnl. To plot the difference between model simulations, activate the line "DEFINE SYMBOL diffplot TRUE" in setmodelrun.jnl.

6 Run CAABA/MECCA in special modes

In the base configuration described so far, CAABA/MECCA calculates the temporal evolution of the chemistry inside an air parcel. This is ideal for sensitivity studies analyzing the effect of individual reactions inside a large chemical mechanism. For other applications, some special modes exist as described below.

6.1 Multiple model simulations and steady-state

The so-called "multirun" mode performs multiple model simulations, each of them terminating when a steady-state has been reached. This is useful to calculate the steady-state concentrations of short-lived species (e.g. OH) when the concentrations of longer-lived species (e.g. non-methane hydrocarbons) are known from measurements. The default termination condition is that the change of OH between two model time steps is less than 0.1 %. If necessary, this can be changed in the function steady_state_reached in messy_mecca.f90. To avoid that the concentrations of long-lived species change from their initial values, they can be fixed in the file mecca/messy_mecca_kpp.kpp by adding them to the "#SETFIX" line. Initial mixing ratios and J-values must be available in netCDF files in the multirun/input/ directory. As an example, the file example.nc is available. To create such input netCDF files from ASCII files, the script asc2ferret4nc.tcsh can be used. Finally, since the multirun mode needs "ncks" from the netCDF Operators (NCO) software, it must be ensured that this program is available. After these preparations, the multirun mode can be entered by running xcaaba and answering the question "Run CAABA boxmodel with MECCA chemistry?" with "m". This will start the multirun.tcsh script in the multirun/ directory. The user can either select one input file or make model simulations for all input files in the multirun/input/ directory. For each input netCDF file, the script loopcaaba.tcsh is called. For each time step contained in the input file, loopcaaba.tcsh performs a CAABA/MECCA model simulation. It first creates a suitable namelist file caaba.nml. Values for temperature and pressure are transferred from the input netCDF file to the namelist file. In addition, loopcaaba.tcsh creates two important settings in caaba.nml: First, the steady-state option is switched on with "l_steady_state_stop = T". Second, the submodel READJ is activated and used with "USE_READJ = T"

and "`photrat_channel = 'readj'`". After the model simulations have finished, a summary of the output is placed in the output directory. The name of the output directory will be based on the name of the input netCDF file, e.g. when the file `example.nc` is used, the output will be in `output/multirun/example/`.

6.2 Monte-Carlo

6.2.1 Performing Monte-Carlo simulations

In the Monte-Carlo mode, several CAABA/MECCA simulations are performed, with each individual simulation using slightly different rate coefficients. To activate it, you first have to create a new chemistry mechanism with `xmecca` (see Sect. 4) and answer the question "Add Monte-Carlo factor to all rate coefficients" with "y". This will start the awk script `mcfct.awk`, which adds Monte-Carlo factors to the rate coefficients in the equation file. Next, the `xcaaba` script can be used to start the simulations. It will start the script `montecarlo.tcsh` in the directory `montecarlo/`. The default is to make 5 model simulations. To choose another value (up to 9999), change the definition of `maxline` in `montecarlo.tcsh`.

6.2.2 Analyzing Monte-Carlo simulations

After performing the model simulations, the resulting netCDF files are merged (using the tools `ncpdq`, `ncclamp`, and `ncrcat`) and then stored in the output directory with a name based on the date and time of the model simulations, e.g. `$outputdir = output/montecarlo/2010-06-24-16:29:00`. The final concentrations and rate coefficients of all simulations are summarized in `caaba_mecca_c_end.nc` and `caaba_mecca_k_end.nc`. Results of the individual simulations can be found in the directories `$outputdir/runs/*`.

Time series

If the model is set up to run for a fixed length (e.g. using the default of `ext_runtime` = 8 days), the time series of all simulations can be plotted together with ferret by activating the lines for Monte-Carlo in `setmodelrun.jnl`. However, these plots become illegible if more than about 5 simulations are made.

Steady-state calculations

The most efficient way to analyze a large number of Monte-Carlo simulations is to use the steady-state option and only compare the final values of the different model simulations, not the individual time series.

The ferret script `montecarlo.jnl` can be used to create scatter plots of concentrations vs rate coefficients. It also plots linear regression lines for all comparisons above a certain threshold of the correlation coefficient (default: $r^2 > 0.05$).

6.2.3 Variation of rate coefficients

In each individual Monte-Carlo simulation j, all rate coefficients k_i are varied by a Monte-Carlo factor:

$$k_{i,j}^{\mathrm{MC}} = k_i \times f_i^{x_{i,j}} \tag{1}$$

Here, $k_{i,j}^{\mathrm{MC}}$ is the rate coefficient of reaction i used in the Monte-Carlo simulation j. It is defined as the product of the recommended value k_i and the Monte-Carlo factor $f_i^{x_{i,j}}$. This Monte-Carlo factor consists of two parts, the uncertainty factor f_i and the exponent $x_{i,j}$:

The uncertainty factor f_i

The uncertainty factor f_i describes the uncertainty of the measured (or estimated) rate coefficient k_i. Its value can usually be found in publications of laboratory studies or summaries like the JPL evaluation (Sander et al., 2006).

The tables of the IUPAC evaluations (e.g. Atkinson et al., 2006) list the decadic logarithm $\lg(f_i)$ of the uncertainty factor, which they call "$\Delta \log k$".

Sometimes an absolute uncertainty is quoted instead of an uncertainty factor, e.g. $k = 2 \pm 0.2$ or $k = 2 \pm 10\%$. In this case we define f_i such that the upper limit is reached when multiplied with k_i, i.e. in the current example $f_i = (2 + 0.2)/2 = 1.1$.

The uncertainty factor is defined in the equation files (`*.eqn`) in a comment starting with the paragraph symbol. Three different syntax types are possible:

- If there is just one § sign, (e.g. "`{§1.1}`"), the value inside the curly braces is the uncertainty factor f_i.
- With two § signs, (e.g. "`{§§0.04}`"), the value inside the curly braces equals $\lg(f_i)$.
- If there is only a § sign ("`{§}`") but no number, the uncertainty factor is set to the default value of $f_i = 1.25$.

The Monte-Carlo exponent $x_{i,j}$

There is one Monte-Carlo exponent $x_{i,j}$ (variable "`mcexp`" in the code) for each rate coefficient k_i and for each individual Monte-Carlo simulation j. The

values of $x_{i,j}$ are normally-distributed random numbers centered around zero, and produced with the Marsaglia polar method (`http://en.wikipedia.org/wiki/Marsaglia_polar_method`). As input for the Marsaglia polar method, uniformly distributed random numbers between 0 and 1 calculated with either the standard Fortran90 function `RANDOM_NUMBER` or the Mersenne Twister algorithm (Matsumoto and Nishimura, 1998) are used.

6.2.4 Changing the uncertainty factors

The uncertainty factors can be changed by modifying the equation files, as shown in Sect. 7.3. Note that predefined rate coefficients (e.g. `k_HO2_HO2`) already contain an uncertainty factor and there must not be an additional factor in the reaction where they are used.

In some cases, it may be useful to vary only one or a few rate coefficients. To do this, it is first necessary to find the correct indices of `mcexp(...)` in `mecca.eqn` (note that these indices may change when creating a new mechanism with `xmecca`). As an example, to vary only the rate coefficients that use `mcexp(40)` and `mcexp(50)`, the following lines can be added to subroutine `mecca_init` in `messy_mecca_box.f90` after `CALL define_mcexp`:

```
DO i=1, MAX_MCEXP
  IF ((i/=40).OR.(i/=50)) mcexp(i) = 0.
ENDDO
```

To verify that the rate coefficients are modified in the Monte-Carlo simulations, it is possible to temporarily activate the subroutine `montecarlo_check` in `template_messy_mecca_kpp.f90` and check the output in `caaba.log`. After these tests, `montecarlo_check` must be switched off again to allow normal model simulations.

6.3 Lagrangian trajectories

CAABA can be used as a Lagrangian trajectory box model (Riede et al., 2009). The usual combination of submodels for this purpose includes the CAABA submodel TRAJECT for the processing of trajectory information, MECCA for atmospheric chemistry, and JVAL for photolysis rate calculation. All important settings for trajectory calculations can be made via the namelist file plus a few external files.

6.3.1 Namelist parameters

A namelist template can be found in `nml/caaba_traject_example.nml`. After copying the namelist file to a new name in the same directory and altering the settings, it will be available when running caaba via `xcaaba`. There are standard and trajectory-exclusive namelist parameters to be set:

- **Submodel switches (mandatory):** The trajectory mode of CAABA requires that `USE_TRAJECT = T`, `USE_MECCA = T`, and `USE_JVAL = T` and/or `USE_SAPPHO = T`. One of the two photolysis rate models is sufficient, but it is also possible to run them in parallel (see also `photrat_channel`). If the use of external photolysis rates is desired, `USE_JVAL = T` is mandatory. For the application in Riede et al. (2009), the submodel `SEMIDEP` was switched off. (`!USE_SEMIDEP = T`).
- **Scenarios (optional):** Scenarios may be used in trajectory mode. When using external input for chemical initialization and photolysis rates, however, they can be ignored (commented out, e.g., `!init_scenario = ''`) or used as complement.
- **Photolysis rate channel:** Choose `photrat_channel = 'jval'` when planning to prescribe photolysis rates.
- **Trajectory input (mandatory):** The path to the netCDF file containing trajectory waypoints should be specified as `input_physc = 'traject/example_traj.nc'`. For its structure, see section 6.3.2.
- **Tracer initialization (optional):** Tracer mixing ratios can be initialized with an external netCDF file by specifying the path to it with `init_spec = 'traject/example_init.nc'`. For its structure, please refer to section 6.3.4.
- **Photolysis rates (optional):** For prescribed photolysis rates, specify the path to the respective file as `input_jval = 'traject/example_jval.nc'`. For its structure, see 6.3.3.
- **Variable names (partly mandatory):** There are default trajectory variable names designated in CAABA. They can be selectively changed by providing alternative variable names. Here is a list of trajectory variables, their default name, and respective examples how to specify an alternative variable name:

quantity	default	alternative
longitude	`LON`	`vlon = 'LON_TR'`
latitude	`LAT`	`vlat = 'LAT_TR'`
pressure	`PRESS`	`vpress = 'P'`
temperature	`TEMP`	`vtemp = 'TM1'`
rel. humidity		`vrelhum = 'rh'`
spec. humidity		`vspechum = 'sh'`

Humidity has no default variable name due to the choice of either providing relative humidity or specific humidity. Thus, it is mandatory to specify exactly one of the two in the namelist. When specific humidity is provided, then both specific humidity and relative humidity are written to the output `caaba_physc.nc`, since CAABA uses relative humidity internally. When relative humidity is provided, only relative humidity will be written to output.

- **Integration time (optional):** `time_step` sets the integration and output time step. See also section 7.1.1.
- **Clipping the trajectory (optional):** Two namelist parameters allow flexible cropping of the model runtime along the trajectory. `runlast` defines the start of the CAABA simulation counted backwards in time from the trajectory end, i.e. `runlast = 4.5` means "calculate the last 4.5 days of the trajectory". The unit is days. The parameter `ext_runtime` defines the overall model simulation time. Thus, `runlast` and `ext_runtime` combined clip out any desired part of the trajectory.

6.3.2 Trajectory input file

The trajectory information is provided to CAABA via an external netCDF file specified in the namelist by `input_physc`. A sample file is available at `traject/example_traj.nc`. The file should contain a time origin in 'seconds/minutes/hours/days since yyyy-mm-dd hh:mm:ss', where the seconds in the time string are optional, for example: `"MINUTES since 2000-01-19 08:00:00"`. The file must contain at least two trajectory waypoints and the following time-dependent variables:

quantity	default name	unit
longitude	`LON`	degrees east
latitude	`LAT`	degrees north
pressure	`PRESS`	Pascal
temperature	`TEMP`	Kelvin
(rel. humidity)		1
(spec. humidity)		kg/kg

Of the two humidity quantities, only one needs to be present.

6.3.3 Photolysis rate file

It is possible to prescribe photolysis rate coefficients via netCDF file. An example is available at `traject/example_jval.nc`. At the moment, only photolysis rates for the species NO_2 can be read into the model and must have the variable name `J_NO2`. The files specified in `input_jval` (e.g. `example_jval.nc`) and `input_physc` (e.g. `example_traj.nc`) must both refer to exactly the same trajectory as the photolysis rate values are read into the model at the same times as other trajectory information.

6.3.4 Tracer initialization file

There are default initial values included in MECCA for a variety of species and simulation aims (see `init_scenario`). However, for several consecutive simulations with changing initialization, the possibility to define an initialization file using `init_spec` is convenient. The tracer initialization file is a netCDF file with one point in time, at which selected species' mixing ratios are defined in mol/mol. The point in time itself is not important and not checked when reading the initial mixing ratios. All tracers that are not specified in `init_spec` are initialized according to default or a chosen `init_scenario`. An example for the initialization file can be found at `traject/example_init.nc`.

6.3.5 Trajectory mode output

Output along the trajectory is written to `caaba_messy.nc`. There are some special variables written out in addition to the default `caaba_messy.nc` output. They are listed in the second part of the table.

variable	unit	notes
`lon`	(dummy 3-D x-coordinate)	
`lat`	(dummy 3-D y-coordinate)	
`lev`	(dummy 3-D z-coordinate)	
`time`	<unit>since ...	time
`lon_tr`	deg east	longitude
`lat_tr`	deg north	latitude
`press`	Pa	pressure
`temp`	K	temperature
`relhum`	1	relative humidity (RH)
`spechum`	kg/kg	specific humidity (q)
`sza`	deg	solar zenith angle
`J_NO2_x`	1/s	NO_2 photolysis rate
`localtime`	same as time	local time
`year_loc`		year of local time
`month_loc`		month of local time
`day_loc`		day of local time
`hour_loc`		hour of local time
`min_loc`		minute of local time
`sec_loc`		second of local time

The specific humidity q is only written to output if it was provided as input. In that case, the relative humidity RH is calculated using the WMO definition

(Jacobson, 1999):

$$\mathrm{RH} = \frac{\omega_v}{\omega_{vs}} = \frac{p_{\mathrm{H_2O}}(T)}{p - p_{\mathrm{H_2O}}(T)} \times \frac{p - p_{\mathrm{sat}}(T)}{p_{\mathrm{sat}}(T)} \qquad (2)$$

with ω_v = water vapor mass mixing ratio and ω_{vs} = saturation water vapor mass mixing ratio. We calculate these as:

$$\omega_v = \frac{q}{1-q} \qquad (3)$$

$$\omega_{vs} = \frac{M(\mathrm{H_2O})}{M(\mathrm{air})} \times \frac{p_{\mathrm{sat}}(T)}{p - p_{\mathrm{sat}}(T)} \qquad (4)$$

using the pressure p, the temperature-dependent saturation water vapor pressure $p_{\mathrm{sat}}(T)$, and the molar masses M of water and dry air.

7 Modifying CAABA/MECCA

The CAABA/MECCA model simulation can be modified by changing the namelist files (`*.nml`), the species files (`*.spc`), the equation files (`*.eqn`) and the Fortran90 files (`*.f90`).

7.1 Namelist files

Fortran90 namelist files allow modifications of the model simulation without having to recompile the source code.

7.1.1 The CAABA namelist file `caaba.nml`

The file `caaba.nml` contains the namelist `&CAABA`. Here individual parts of the CAABA model (the so-called "MESSy submodels") can be switched on or off. It is important that the following switches are set to "T" (=true):

```
USE_MECCA   = T
USE_SAPPHO  = T
USE_SEMIDEP = T
```

To use the photolysis rate coefficients from SAPPHO in MECCA, set:

```
photrat_channel = 'sappho'
```

Alternatively, you can switch on the JVAL submodel with `USE_JVAL = T` and then select `photrat_channel = 'jval'`. It is fine to switch on both the JVAL and the SAPPHO submodel, which can be useful for a comparison. However, only the values selected by `photrat_channel` are used for the MECCA chemistry.

You can define the model start, runtime, and time step, e.g.:

```
startday    = 90.
ext_runtime = '10 days'
time_step   = '15 minutes'
```

If you don't set these, the default is a model start on Julian day 80, a model simulation duration of 8 days, and an output time step of 20 minutes. The `time_step` value can be given as any integer or floating point and in the units of `seconds`, `minutes`, or `hours`.

As an alternative, it is possible to stop the model simulation when a steady state has been reached. This is normally used in the "multirun" mode (Sect. 6.1):

```
l_steady_state_stop = T
```

The default location of the model simulation (latitude, longitude) is 45° N and 0° E. It can be changed here, e.g.:

```
degree_lat = 82  ! Alert
degree_lon = 297 ! Canada
```

Changing it only affects photolysis calculations (via the zenith angle calculations).

The values of temperature (`temp`), pressure (`press`), relative humidity (`relhum`), and the height of the boundary layer (`zmbl`) can be defined, e.g.:

```
temp   = 293.    ! [K]
press  = 101325. ! [Pa]
relhum = 0.81    ! [1]
zmbl   = 1000.   ! [m]
```

The values shown here are the default values as defined in `caaba_mem.f90`.

In the submodel SEMIDEP, emissions are distributed evenly into the well-mixed boundary layer of height `zmbl`. Note that CAABA is only a box model, and changing `zmbl` has no other effects than this.

It is possible to initialize chemical species from a netcdf file. To activate this feature, define a valid input file name, e.g.:

```
init_spec = 'inputfile.nc'
```

If the submodel READJ is switched on, a netcdf file containing J-values must be defined. In addition, an index can be defined if the netcdf file contains data for more than one time step, e.g.:

```
init_j = 'readj_input.nc'
init_j_index = 25
```

Table 1: List of CAABA/MECCA Fortran90 files

CAABA box model related files	
`caaba.f90`	main box model file
`caaba_io.f90`	input/output
`caaba_io_netcdf.inc`	netCDF input/output
`caaba_io_ascii.inc`	ASCII input/output
`caaba_mem.f90`	declaration of CAABA variables
`messy_main_control_cb.f90`	flow control
`messy_jval_box.f90`	connection of JVAL to CAABA
`messy_mecca_box.f90`	connection of MECCA to CAABA
`messy_mecca_dbl_box.f90`	(under construction)
`messy_mecca_tag_box.f90`	(under construction)
`messy_readj_box.f90`	connection of READJ to CAABA
`messy_sappho_box.f90`	connection of SAPPHO to CAABA
`messy_semidep_box.f90`	simplified emission and deposition, including connection to CAABA
`messy_traject_box.f90`	trajectory calculations
static core files	
`messy_cmn_photol_mem.f90`	common definitions for photolysis
`messy_main_constants_mem.f90`	physical constants
`messy_main_blather.f90`	print utilities
`messy_main_rnd.f90`	random number generation
`messy_main_rnd_lux.f90`	Luxury random numbers
`messy_main_rnd_mtw.f90`	Mersenne-Twister random numbers
`messy_main_timer.f90`	timer
`messy_main_tools.f90`	auxiliary functions
`messy_main_tools_kp4_compress.f90`	(file exists but is not not used with CAABA)
`messy_jval.f90`	calculation of J-values
`messy_jval_jvpp.inc`	include file for JVAL
`messy_readj.f90`	read J-values
`messy_sappho.f90`	simplified and parameterized photolysis rate coefficients
static MECCA core files in the `mecca/smcl/` directory	
`messy_mecca.f90`	MECCA core
`messy_mecca_aero.f90`	aerosol chemistry
`messy_mecca_khet.f90`	(file exists but is not used with CAABA)
KPP- and `xmecca`-produced files in the `mecca/smcl/` directory	
`messy_mecca_kpp.f90`	a wrapper for the KPP files
`messy_mecca_kpp_function.f90`	ODE function
`messy_mecca_kpp_global.f90`	global data headers
`messy_mecca_kpp_initialize.f90`	initialization
`messy_mecca_kpp_integrator.f90`	numerical integration
`messy_mecca_kpp_jacobian.f90`	ODE Jacobian
`messy_mecca_kpp_jacobiansp.f90`	Jacobian sparsity
`messy_mecca_kpp_linearalgebra.f90`	sparse linear algebra
`messy_mecca_kpp_monitor.f90`	equation info
`messy_mecca_kpp_parameters.f90`	model parameters
`messy_mecca_kpp_precision.f90`	arithmetic precision
`messy_mecca_kpp_rates.f90`	user-defined rate laws
`messy_mecca_kpp_util.f90`	utility input-output

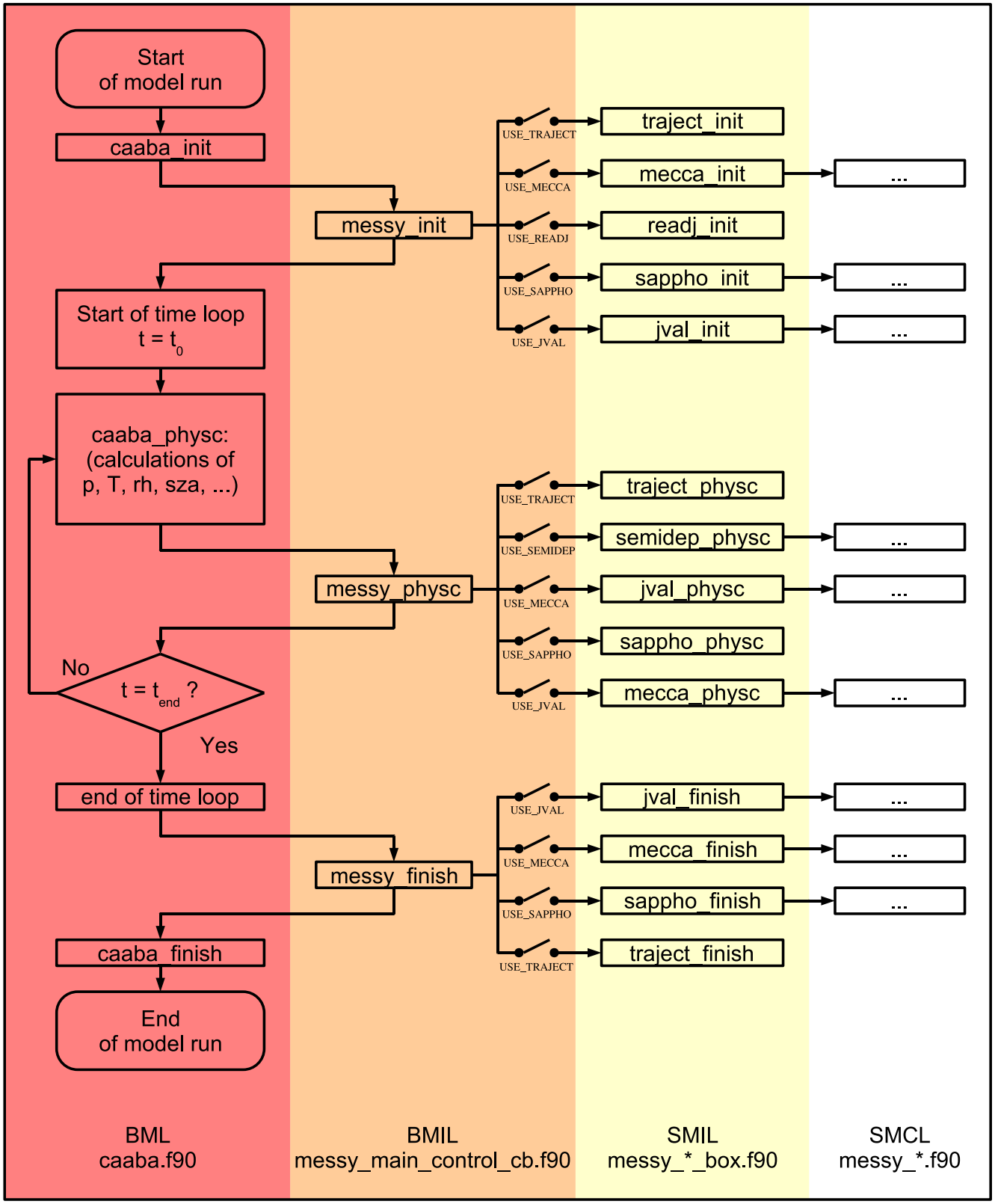

Figure 6: Flow control of a CAABA box model simulation

To facilitate running CAABA under different boundary conditions, so-called "scenarios" can be defined. Currently, there are scenarios for photolysis (`photo_scenario`), initialization (`init_scenario`), emission (`emission_scenario`), and dry deposition (`drydep_scenario`):

```
photo_scenario    = 'MBL'
init_scenario     = 'MBL'
emission_scenario = 'MBL'
drydep_scenario   = 'MBL'
```

Possible values ("`MBL`" is used here as an example) for the scenarios can be found in the variable `list_of_scenarios` in `caaba.f90`. Note that values have not yet been assigned for all scenarios. New data can be added to these files:

scenario	subroutine	file
`photo`	`jvalues`	`messy_sappho.f90`
	`jval_init`	`messy_jval_box.f90`
`init`	`x0`	`messy_mecca_box.f90`
`emission`	`emission`	`messy_semidep_box.f90`
`drydep`	`drydep`	`messy_semidep_box.f90`

Finally, it is possible to skip the chemistry calculations with KPP completely. This is only useful for debugging:

```
l_skipkpp = T
```

7.1.2 The MECCA namelist file `mecca.nml`

The file `mecca.nml` contains the namelists `&CTRL_KPP` and `&CTRL` (the namelist `&CPL` is not used in connection with CAABA). `&CTRL_KPP` is used for fine-tuning the numerical integration. The default selection `icntrl(3) = 2` should normally be suitable.

7.1.3 The JVAL namelist file `jval.nml`

The file `jval.nml` contains several namelists. When JVAL is run as a submodel for CAABA, only the control namelist (`&CTRL`) is used. Normally, there is no need to change the default setting of "`time_control = 'constant'`".

7.2 The species files `gas.spc` and `aqueous.spc`

The files `*.spc` declare chemical species for KPP. All species that may occur in an equation must be declared here. Additional dummy species may also be declared here.

Gas-phase species are declared in `gas.spc`. Examples for gas-phase species are `O2`, `O1D`, and `NO2`. The names of lumped species start with "`L`". The names of species with two or more carbon atoms are taken from the master chemical mechanism (MCM).

MECCA also includes aqueous species which are declared in `aqueous.spc`. The names of cations end with "`p`" for plus. The names of single-charge anions end with "`m`" for minus. Doubly-charged anions end with "`mm`". Examples for aqueous species are `H2O2`, `Hp`, `NO3m`, and `SO4mm`.

All aqueous-phase species have the suffix "`_a##`", which is a placeholder for the aerosol phase number. `xmecca` replaces it by either "`_a01`" (accumulation soluble) or "`_a02`" (coarse soluble). This allows separate chemistry calculations for aerosol particles of different size and composition.

All species are defined here with `#DEFVAR`, i.e. KPP considers them as prognostic variables. To treat a species as a constant (e.g. CO_2), it can be added to the `#SETFIX` command in the file `messy_mecca_kpp.kpp`.

7.3 The equation files `gas.eqn` and `aqueous.eqn`

The equation files `*.eqn` define the chemical reaction mechanism for KPP. After making any changes to the equation files, it is always necessary to execute KPP via `xmecca` again. Each reaction occupies one line in this file. An example is:

```
<G1000> O2 + O1D = O3P + O2 : {%StTrG}
   3.3E-11*EXP(55./temp); {&1945}{§1.1}
```

The line starts with the reaction number, which is enclosed in angle brackets "`<...>`" (see Sect. 7.3.1). The second part (up to the colon) defines the reaction, and the third part (between the colon and the semicolon) defines the rate coefficient. The lines may also contain comments. Comments in equation files are either enclosed in curly braces, or the comment line starts with `//`. When using `xmecca`, some comments have a special meaning. Comments starting with the percent symbol "`{%...}`" are markers (see Sect. 7.3.2). Comments starting with the ampersand "`{&...}`", the "at"-symbol "`{@...}`", or the dollar "`{$...}`" are used to store information for the listing of reactions, as explained in Sect. 7.3.3. Comments starting with the paragraph symbol "`{§...}`" are defining uncertainties of the rate coefficients for Monte-Carlo calculations (see Sect. 6.2).

If the definition of a rate coefficient is very complex, it can be stored in a Fortran90 variable and the variable is

put into the `gas.eqn` file. For example, the rate of the self reaction of HO_2 is quite complex since it depends on humidity. It is predefined and the reaction line can be simplified to:

```
HO2 + HO2 = H2O2 : k_HO2_HO2;
```

The declaration and definition of `k_HO2_HO2` are also in the `gas.eqn` file. They can be found in the so-called KPP "inline types" `F90_GLOBAL` and `F90_RCONST`, e.g.:

```
#INLINE F90_GLOBAL
  REAL :: k_HO2_HO2
#ENDINLINE

#INLINE F90_RCONST
  k_HO2_HO2 = (1.5E-12*EXP(19./temp)+ &
               1.7E-33*EXP(1000./temp)*cair)* &
               (1.+1.4E-21* &
               EXP(2200./temp)*C(KPP_H2O))
#ENDINLINE
```

Another method to add reaction rates with complex dependencies are Fortran90 functions. This is done for example for the oxidation of S(IV) by H_2O_2 (`k_SIV_H2O2`). A function call is given as the rate in the `*.eqn` file. These functions are defined with the "inline type" `F90_RATES`:

```
#INLINE F90_RATES
  ELEMENTAL REAL(dp) FUNCTION k_SIV_H2O2 &
    (k_298,tdep,cHp,temp)
    ...
  END FUNCTION k_SIV_H2O2
#ENDINLINE
```

7.3.1 Reaction numbers

Each reaction in an equation file has a unique reaction "number" (number is not quite correct, since letters are included as well), which is enclosed in angle brackets, e.g.: "`<G1000>`". The reaction number starts with one or more upper case letters denoting the type of reaction. The following types exist:

A	aqueous-phase reactions
H	Henry's law (dissolution and evaporation)
EQ	equilibria in the aqueous phase (forward and backward reactions of acid/base and other equilibria)
G	gas-phase reactions
J	J-values of photolysis reactions
HET	heterogeneous reactions (e.g. on polar stratospheric clouds)

The type is followed by a sequence of 3 or 4 digits. The first digit is the number of the main element of the reaction. The following numbers are used:

1)	O	Oxygen
2)	H	Hydrogen
3)	N	Nitrogen
4)	C	Carbon
5)	F	Fluorine
6)	Cl	Chlorine
7)	Br	Bromine
8)	I	Iodine
9)	S	Sulfur
10)	Hg	Mercury

Out of those elements that occur in a reaction, the one with the highest number is called the main element. Accordingly, the second digit is determined by the element with the second highest number (or set to zero if there is no second element in the reaction). There is one exception in this numbering scheme: For the carbon group, the second digit is the number of C atoms in the largest organic molecule.

The following digits have no special meaning. If a reaction branches into several pathways, a suffix "a", "b", "c", ... is added.

7.3.2 Markers and labels

Each reaction must contain a marker. A marker contains several labels. The syntax is "`{%...}`" where the dots represent the labels. Labels are used to select specific reactions, as described above (Sect. 4.1). The labels are placed in the marker without separators. The following labels are available and should appear in this order:

1. altitudes at which the reaction occurs (mandatory, include at least one)
   - `St` = Reactions relevant in the stratosphere
   - `Tr` = Reactions relevant in the troposphere
2. phase (mandatory, include exactly one)
   - `Aa##` = Aqueous, aerosol (`##` is a placeholder for the 2-digit aerosol phase number)
   - `G` = Gas phase reactions
   - `Het` = heterogeneous reactions (e.g. on polar stratospheric clouds)
3. elements (include all elements that occur in the reaction, except for H and O)
   - `N` = Nitrogen
   - `C` = Carbon with > 1 C atom (only used for C,N,O species but not for halogenated or sulfur-containing organics)
   - `F` = Fluorine
   - `Cl` = Chlorine
   - `Br` = Bromine
   - `I` = Iodine
   - `S` = Sulfur
   - `Hg` = Mercury

4. other
   - `J` = Photolysis reactions
   - `Mbl` = Minimum reaction mechanism for MBL chemistry
   - `Sc` = Scavenging chemistry mechanism
   - `Scm` = Scavenging chemistry mechanism, minimum selection

See Sect. 7.5.8 for a description how to add new labels to `xmecca`.

7.3.3 Creating a table of the chemical mechanism

To ensure that the documentation of the chemical mechanism is always up to date, the necessary information is contained inside the species and equation files. If you have the programs pdfLaTeX and BibTeX installed on your system, you can generate a table of the chemical mechanism in pdf format.

The awk scripts `spc2tex.awk` and `eqn2tex.awk` convert information from the selected reactions into a LaTeX table. BibTeX citations are included in comments starting with an ampersand "`{&...}`". If there is a second ampersand "`{&&...}`", additional information about reactions can be found in `meccanism.tex` as a footnote to the tables. Comments starting with the at symbol "`{@...}`" or the dollar "`{$...}`" can be used to put LaTeX commands directly into the `*.eqn` files. `eqn2tex.awk` produces several `*.tex` files which are included into `meccanism.tex`.

7.4 Fortran90 files

The CAABA/MECCA simulations can be modified by changing the Fortran90 files (see Tab. 1 for a list of files). The modular structure of the Fortran90 files is shown in Fig. 3. Most of the files need only be changed by model developers. Those that are also interesting for model users, are briefly explained below.

7.4.1 `caaba.f90`

This file contains the main Fortran90 code ("`PROGRAM caaba`").

7.4.2 `caaba_mem.f90`

This file contains variable declarations which are needed by several CAABA files.

7.4.3 `messy_main_control_cb.f90`

Flow control. Editing this file is only necessary when a new submodel is added.

7.4.4 `messy_jval_box.f90`

This file contains the connection of JVAL to CAABA.

7.4.5 `messy_jval.f90` and `messy_jval_jvpp.inc`

These files contain the calculation of J-values.

7.4.6 `messy_mecca_box.f90`

The chemical composition of seawater is defined in `SUBROUTINE mecca_init`. Aerosol properties (radius, liquid water content (LWC), and their chemical composition) are defined in `SUBROUTINE define_aerosol`. Initial mixing ratios of chemical species are defined in `SUBROUTINE x0`. Depending on which scenario was chosen in the CAABA namelist file (see Sect. 7.1.1), one of the initialization subroutines `x0_*` will be used.

7.4.7 `messy_sappho_box.f90`

This file contains the connection of SAPPHO to CAABA.

7.4.8 `messy_sappho.f90`

Simplified parameterized photolysis rate coefficients are defined here.

7.4.9 `messy_semidep_box.f90`

Simplified emission fluxes and deposition velocities are defined here.

7.4.10 `messy_mecca_aero.f90`

Several variables needed to calculate rate coefficients are defined in `messy_mecca_aero.f90`. The accommodation coefficients (`alpha`) and the mean velocity (`vmean`) are used for the calculation of the mass transfer coefficients (`ykmt`). Together with the inverse dimensionless Henry's law coefficients (`yhenry`), they are needed to calculate equilibria between the gas and the aqueous phase. Heterogeneous reactions are described with the forward (`k_exf`) and backward (`k_exb`) rate coefficients. The variable `xaer` is set to 1 or 0 to switch aqueous-phase chemistry on or off, respectively. The factor `cvfac` converts the aqueous-phase unit mol/L (refering to the volume of the liquid) to the gas-phase unit molecules/cm^3 (referring to the gas-phase volume).

7.5 How to expand the chemical mechanism

This section contains brief descriptions for experienced model developers explaining where to make changes to the code for certain model expansions. In the descriptions, "xyz" is used as an example for the name of the addition.

7.5.1 Adding a new gas-phase species

- gas.spc:
  Add the new species, its elemental composition, the name in LaTeX syntax, and a comment, e.g.:
  `NC4H10 = 4C + 10H ; {@C_4H_<10>} {n-butane}`
  Note that curly brackets needed by LaTeX must be entered as angle brackets.
- jnl/xxxg.jnl:
  Add one line per new species. Check if the new species is part of an existing familiy, e.g. add new reactive bromine species to Brx.
- jnl/tools/_kppvarg.jnl:
  Add one line per new species.

7.5.2 Adding a new aqueous-phase species

- aqueous.spc:
  Add the new species, the name in LaTeX syntax, and a comment, e.g.:
  `SO4mm_a## = IGNORE; {@SO_4^<2->\aq} {sulfate}`
  The suffix _a## is mandatory. The elemental composition is currently ignored. Note that curly brackets needed by LaTeX must be entered as angle brackets.
- jnl/xxxa.jnl:
  Add one line per new species.
- jnl/_families_a.jnl:
  Check if the new species is part of an existing familiy, e.g. add new bromine species to Brtot.
- jnl/tools/_kppvara.jnl:
  Add one line per new species.

7.5.3 Adding a new gas-phase reaction

First, choose an appropriate reaction number. To avoid that several developers assign the same number to different new reactions, it is strongly recommended that a preliminary reaction number is used initially. This can be done by adding the developer's initials as a suffix, e.g. John Doe would use G0001JD, G0002JD, G0003JD, and so on. When the new code is merged with other development branches, the final reaction numbers will be assigned.

- gas.eqn:
  - Add one line per new reaction.
  - Add Monte-Carlo uncertainty factor.
- latex/meccanism.tex:
  If necessary, add a footnote about the new reaction here.

7.5.4 Adding a new gas-phase photolysis reaction

First, choose an appropriate reaction number, as explained in Sect. 7.5.3.

- gas.eqn:
  - Add one line per new reaction.
  - Add Monte-Carlo uncertainty factor.
- latex/meccanism.tex:
  If necessary, add a footnote about the new reaction here.

Check if the necessary photolysis rate coefficient is already provided by SAPPHO, READJ, and/or JVAL. If not, add it:

- messy_cmn_photol_mem.f90:
  - Add a new index of photolysis ip_XYZ at the end of the list.
  - Increase IP_MAX.
  - Add the name to jname.
- messy_sappho_box.f90:
  Add XYZ to CALL open_output_file and CALL write_output_file.
- messy_sappho.f90:
  Add a simple definition for jx(ip_XYZ).
- messy_jval_box.f90:
  Add one line.
- messy_jval_jvpp.inc:
  Calculate the definition with jvpp or add it manually here.

7.5.5 Adding a new aqueous-phase reaction

First, choose an appropriate reaction number, as explained in Sect. 7.5.3.

- aqueous.eqn:
  - Add one line per new reaction.
  - Add Monte-Carlo uncertainty factor.
- latex/meccanism.tex:
  If necessary, add a footnote about the new reaction here.

7.5.6 Adding a new Henry's law equilibrium

First, choose an appropriate reaction number, as explained in Sect. 7.5.3.

- `aqueous.eqn`:
  - Add two lines per new equilibrium, one for the forward and one for the backward reaction.
  - Add Monte-Carlo uncertainty factors.
- `messy_cmn_gasaq.f90`:
  - Add molar mass:
    `CALL add_species('XYZ', ...)`
  - Add the Henry's law coefficient:
    `CALL add_henry('XYZ', ...)`
  - Add the accommodation coefficient:
    `CALL add_alpha('XYZ', ...)`

  Using these data, the subroutines `mecca_aero_trans_coeff`, `mecca_aero_henry`, and `mecca_aero_calc_k_ex` in `messy_mecca_aero.f90` will automatically calculate the rate coefficients `k_exf` and `k_exb` for `aqueous.eqn`.
- `latex/meccanism.tex`:
  If necessary, add a footnote about the new equilibrium here.

7.5.7 Adding a new acid-base equilibrium

First, choose an appropriate reaction number, as explained in Sect. 7.5.3.

- `aqueous.eqn`:
  - Add two lines per new equilibrium, one for the forward and one for the backward reaction.
  - Add Monte-Carlo uncertainty factors.
- `latex/meccanism.tex`:
  If necessary, add a footnote about the new equilibrium here.

7.5.8 Adding a new label

First, choose a name for the new label. The name must start with an upper case letter and can be followed by one or more lower case letters or numbers. Element symbols must not be used because they are reserved for reactions of that element. For example, since S is sulfur, the symbol S could not be used for the stratosphere. To avoid that several developers introduce new labels with the same name for different purposes, it is strongly recommended that a preliminary label is used initially. This can be done by adding the developer's initials as a prefix, e.g. John Doe would use `Jd1`, `Jd2`, `Jd3`, and so on. When the new code is merged with other development branches, a final label name can be assigned.

- `xmecca`:
  In the generation of `awkfile1`, add another `locate` function, and print the new label to the logfile.

7.5.9 Adding a new emission

- `messy_semidep_box.f90`:
  Add one line to `emission_default` (or one of the other `emission_*` subroutines).

7.5.10 Adding a new deposition

- `messy_semidep_box.f90`:
  Add one line to `drydep_default` (or one of the other `drydep_*` subroutines).

7.6 How to add a new MESSy submodel

- Choose a name (up to 7 alphanumerical characters, starting with a letter). Here, "`xyz`" is used as an example.
- `caaba_mem.f90`:
  `LOGICAL :: USE_XYZ = .FALSE.`
- `messy_xyz.f90`:
  Put all generic subroutines here, i.e. all subroutines that are used for the CAABA boxmodel as well as for a global model.
- `messy_xyz_box.f90`:
  Put CAABA-specific code here. Generic code is not included directly here. Instead, the generic subroutines in `messy_xyz.f90` are called from here. This file contains up to four subroutines:
  - If the submodel needs an initialization, put subroutine `xyz_init` here.
  - If the submodel performs calculations during the time loop, put subroutine `xyz_physc` here.
  - If the submodel prints results, put subroutine `xyz_result` here.

  - If the submodel needs to close any open files at the end of the model simulation, put subroutine `xyz_finish` here.

- `messy_main_control_cb.f90`:
  - Add "`USE_XYZ`" to "`USE caaba_mem`"
  - If subroutine `xyz_init` exists, add:
    `USE messy_xyz_box, ONLY: xyz_init`
    `IF (USE_XYZ) CALL xyz_init`
    to subroutine `messy_init`.
  - If subroutine `xyz_physc` exists, add:
    `USE messy_xyz_box, ONLY: xyz_physc`
    `IF (USE_XYZ) CALL xyz_physc`
    to subroutine `messy_physc`.
  - If subroutine `xyz_result` exists, add:
    `USE messy_xyz_box, ONLY: xyz_result`
    `IF (USE_XYZ) CALL xyz_result`
    to subroutine `messy_result`.
  - If subroutine `xyz_finish` exists, add:
    `USE messy_xyz_box, ONLY: xyz_finish`
    `IF (USE_XYZ) CALL xyz_finish`
    to subroutine `messy_finish`.

- `caaba.f90`:
  Edit subroutine `caaba_read_nml`:
  - Add "`USE_XYZ`" to "`USE caaba_mem`".
  - Add "`USE_XYZ`" to namelist `/CAABA/`.
  - Print value of `USE_XYZ` (see "selected MESSy submodels")
  - If applicable, perform consistency checks for interaction of new submodel with other submodels.

- `nml/default/caaba.nml`:
  Add sensible default values for `USE_XYZ` and possibly other options.

- `manual/caaba_mecca_manual.tex`:
  Mention new submodel in this user manual (Sect. 7.4, Tab. 1, and Fig. 6).

8 Revision history

The major changes between different CAABA/MECCA versions are listed here. A very detailed listing can be found in the file `CHANGELOG`.

New in version 2.6

- In its default configuration, the model now simulates very simple methane chemistry.
- Scenarios in namelists make it easier to define boundary conditions (Sect. 7.1.1).
- Multiple model simulations can be performed, each of them terminating when a steady-state has been reached (Sect. 6.1).
- The Monte-Carlo method can be used to investigate the effect of uncertainties of the rate coefficients (Sect. 6.2).
- The new isoprene oxidation mechanism MIM2 by Taraborrelli et al. (2009) has been added.
- The graphviz software can be used to visualize the complex reaction mechanism.
- Plots of all reaction rates and scaled plot of production/destruction of specific species is now possible with the ferret plotting program.
- The new submodel READJ can read photolysis rate coefficients ("J-values") from a netcdf file.

New in version 2.5

- Output of more information at the start of a model simulation.
- It is not necessary anymore to have the netcdf library for running CAABA/MECCA with ASCII output. The corresponding output functions are now in the `caaba_io*` files.
- For clarity, different initializations and emissions are put into individual subroutines.
- `xmecca`-generated infos are now available as f90 strings in `messy_mecca_kpp_global.f90`.
- A reaction mechanism for Hg chemistry has been added.

New in version 2.4

- `xmecca` runs in batch mode with `*.bat` files.
- Aerosol chemistry is switched on or off automatically (depending on the selected reaction mechanism) via `l_aero`.
- New directory structure: The former `boxmodel/` subdirectory is now the main CAABA directory. The MECCA code is now in the subdirectory `mecca/` in the main CAABA directory.
- The kpp program is now included in the `kpp/` directory in the CAABA/MECCA distribution.
- JVAL was added as a new submodel.
- `xmecca` skips parts that are only needed for global model applications when it is used in the CAABA distribution.

References

Atkinson, R., Baulch, D. L., Cox, R. A., Crowley, J. N., Hampson, R. F., Hynes, R. G., Jenkin, M. E., Rossi, M. J., and Troe, J.: Evaluated kinetic and photochemical data for atmospheric chemistry: Volume II - gas phase reactions of organic species, Atmos. Chem. Phys., 6, 3625–4055, 2006.

Jacobson, M. Z.: Fundamentals of atmospheric modeling, Cambridge University Press, Cambridge, 1999.

Jöckel, P., Sander, R., Kerkweg, A., Tost, H., and Lelieveld, J.: Technical Note: The Modular Earth Submodel System (MESSy) – a new approach towards Earth System Modeling, Atmos. Chem. Phys., 5, 433–444, 2005.

Jöckel, P., Tost, H., Pozzer, A., Brühl, C., Buchholz, J., Ganzeveld, L., Hoor, P., Kerkweg, A., Lawrence, M. G., Sander, R., Steil, B., Stiller, G., Tanarhte, M., Taraborrelli, D., van Aardenne, J., and Lelieveld, J.: The atmospheric chemistry general circulation model ECHAM5/MESSy1: consistent simulation of ozone from the surface to the mesosphere, Atmos. Chem. Phys., 6, 5067–5104, 2006.

Matsumoto, M. and Nishimura, T.: Mersenne twister: a 623-dimensionally equidistributed uniform pseudorandom number generator, ACM Trans. Model. Comput. Simul., 8, 3–30, 1998.

Riede, H., Jöckel, P., and Sander, R.: Quantifying atmospheric transport, chemistry, and mixing using a new trajectory-box model and a global atmospheric-chemistry GCM, Geosci. Model Dev., 2, 267–280, 2009.

Sander, R., Kerkweg, A., Jöckel, P., and Lelieveld, J.: Technical Note: The new comprehensive atmospheric chemistry module MECCA, Atmos. Chem. Phys., 5, 445–450, 2005.

Sander, R., Gromov, S., Harder, H., Jöckel, P., Kerkweg, A., Kubistin, D., Riede, H., Taraborrelli, D., and Tost, H.: The atmospheric chemistry box model CAABA/MECCA-3.0, Geosci. Model Dev. Discuss., (in preparation), 2009.

Sander, S. P., Friedl, R. R., Golden, D. M., Kurylo, M. J., Moortgat, G. K., Keller-Rudek, H., Wine, P. H., Ravishankara, A. R., Kolb, C. E., Molina, M. J., Finlayson-Pitts, B. J., Huie, R. E., and Orkin, V. L.: Chemical Kinetics and Photochemical Data for Use in Atmospheric Studies, Evaluation Number 15, JPL Publication 06-2, Jet Propulsion Laboratory, Pasadena, CA, 2006.

Sandu, A. and Sander, R.: Technical Note: Simulating chemical systems in Fortran90 and Matlab with the Kinetic PreProcessor KPP-2.1, Atmos. Chem. Phys., 6, 187–195, 2006.

Taraborrelli, D., Lawrence, M. G., Butler, T. M., Sander, R., and Lelieveld, J.: Mainz Isoprene Mechanism 2 (MIM2): an isoprene oxidation mechanism for regional and global atmospheric modelling, Atmos. Chem. Phys., 9, 2751–2777, 2009.