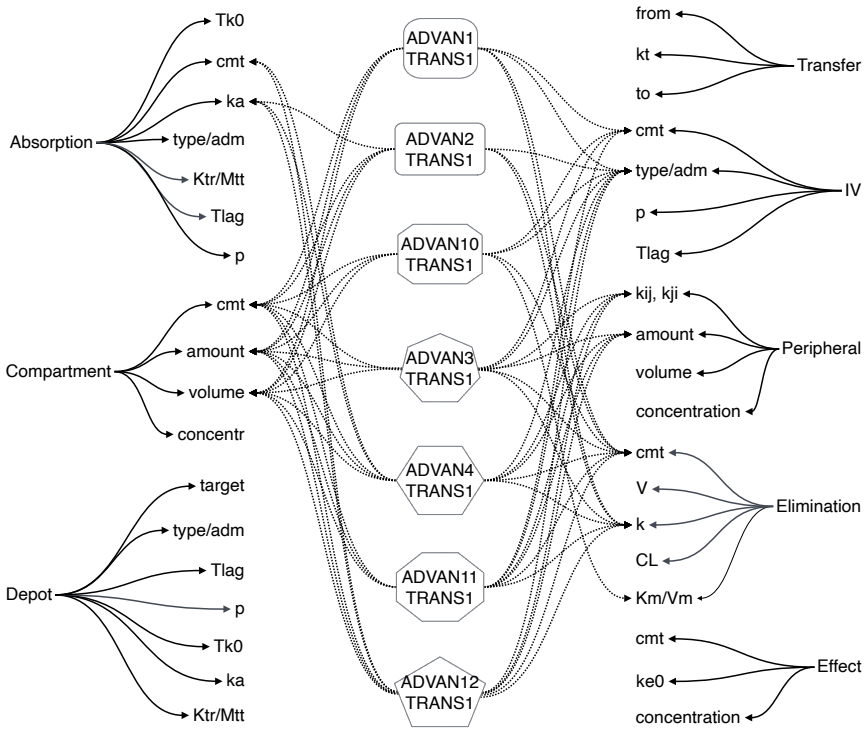
Technical Report

PK Macros in PharmML 0.6

Maciej J Swat[1], Sarala Wimalaratne[1], Niels Rode Kristensen[2], Roberto Bizzotto[3], Marc Lavielle[4]

[1]*EMBL - European Bioinformatics Institute, Cambridge, UK*,
[2]*Novo Nordisk A/S, Bagsværd, Denmark*,
[3]*National Research Council, Institute of Neuroscience, Padova, Italy*,
[4]*Inria Saclay, France*



March 12, 2015

Contents

Chapter 1

Introduction

A long standing problem is how to define effectively PK models, without using ODEs, in a tool-independent way. There are many tool-specific solutions, each of them using their own ways to define PK models. After an thorough analysis of a number of available approaches, the MLXTRAN PK macro system has been selected as the one which should be supported by PharmML [3]. It has the following features

- Tool-independent, but easily translatable to tool-specific libraries
- User friendly and easy to learn for users new to the field
- Flexible and providing consistent specification of compartmental PK models
- More models can be specified without ODEs, compared to nmadvan/PREDPP
- No limitations will be imposed, compared to nmadvan/PREDPP:
  - All PREDPP library models with underlying analytical solutions (ADVAN1-4, 11-12) can be specified. Templates are provided for these to help NONMEM users, see Section 3.1.
  - The PREDPP library models applying matrix exponential-based solutions (ADVAN 5, 7) are already being defined in a way very similar to the macro-based solution proposed.
  - The PREDPP library models applying numerical ODE solution (ADVAN 6, 8-9, 10, 13) can still be specified by using ODEs explicitly (or by using/mixing with macro-definitions).

Adopting the PK macros should allow seamless translation of models between MDL and PharmML and MLXTRAN. Therefore, in the PharmML implantation of the macros we tried to stay as close as possible to the original MLXTRAN specification. We had to make sure that the macros, their attributes and possible assignments are defined in a unambiguous way. In the sections 2.2 and 2.3 we explain the rules to be followed in their implementation and the translation process to and from PharmML.

1.1 The macro idea

Figure 1.1 visualises the principle of the PK macros solution and it's use. The module within the inner MDL/PharmML box shows few example PK macros and their attributes using which a PK model can be formulated. In this case four macros, with one or more arguments, are used to encode a basic *oral 1-compartment model with linear elimination* consisting of rolling items

1. depot compartment, with amount Ad,
2. central compartment, with amount Ac and volume V,
3. oral administration, 1^{st} order absorption, ka, and
4. linear elimination, with constant rate k.

This is the *only* information needed to be encoded in PharmML for this particular PK model. It corresponds to the following macro

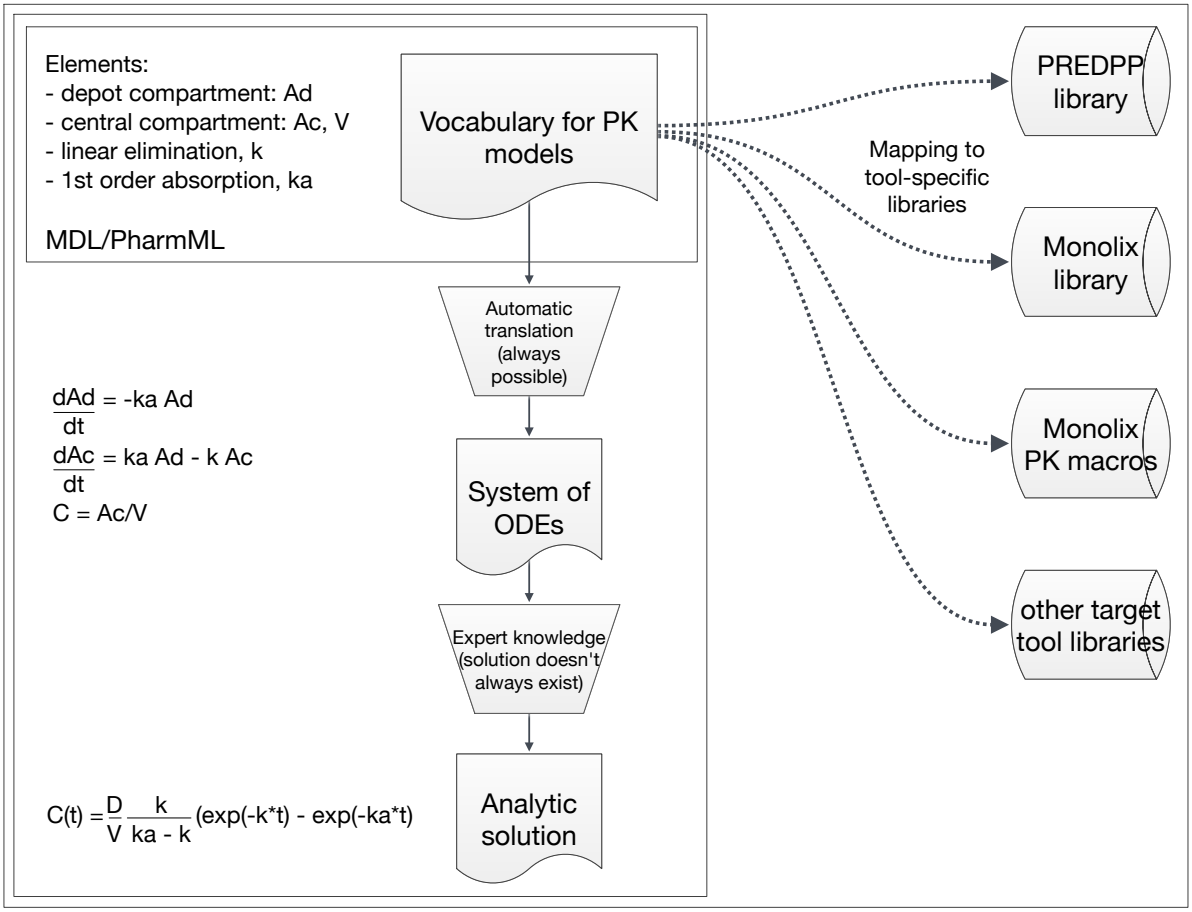


Figure 1.1: A diagram explaining how the PK macro system can be used in practise. The inner MDL/PharmML box contains the information to be stored in PharmML, i.e. the macros and arguments from the PK model vocabulary. It can then be uniquely translated into a set of ODEs and for some models there exists an analytic solution. Models encoded using the controlled vocabulary can be mapped to tool specific implementations (dotted lines).

```
compartment(cmt=1,concentration=Cc,volume=V)
oral(cmt=1, ka)
elimination(cmt=1, k)
```

Any information about input, i.e. dosing times and amounts will be stored in the dataset or explicitly within the `<TrialDesign>`.

The key realisation is that every set of such macro statements, if correctly defined, can be translated automatically into a unique set of ODE's and/or algebraic equations, in this case:

$$\frac{dAd}{dt} = -ka \times Ad$$
$$\frac{dAc}{dt} = ka \times Ad - k \times Ac$$
$$C = Ac/V$$

Moreover, for some models such as the above one, there exists an analytic solution. This however cannot be derived automatically for an arbitrary ODE system. It requires a powerful symbolic calculation software or expert knowledge. Once the PK model macros and their attribute set have been defined, a mapping to tool-specific libraries can be provided, see Fig.1.1. This again requires expert knowledge of each particular tool.

Chapter 2

PK macros in PharmML

2.1 MLXTRAN macros

PK macros offer an equation-free encoding solution for pharmacokinetic models. The novel system allows to implement in MLXTRAN virtually any compartmental model within well-defined constraints. See [3] for a detailed description. The following table lists the available PK macros with their arguments.

Component	Macro	Arguments
Compartment	*compartment*	cmt, amount, volume, concentration
Peripheral compartment	*peripheral*	**kij**, **k_i_j**, amount, volume, concentration
Effect compartment	*effect*	cmt, **ke0**, concentration
Absorption from a depot	*depot*	type/adm, **Tlag**, **p**, target, **Tk0**, **ka**, **Ktr/Mtt**
Absorption for IV dose	*iv*	cmt, type/adm, **Tlag**, **p**
0-order absorption	*absorption* or *oral*	cmt, type/adm, **Tlag**, **p**, **Tk0**
1st order absorption	*absorption* or *oral*	cmt, type/adm, **Tlag**, **p**, **ka**, **Ktr/Mtt**
Linear elimination	*elimination*	cmt, volume, **k**, **Cl**
MM elimination	*elimination*	cmt, **Km**, **Vm**
Transfer	*transfer*	from, to, **kt**

Table 2.1: PK macros and their arguments as used in MLXPLORE, based on [3]. Please, note that there are more components listed, 10, then macros, 9, which are actually defined in MLXTRAN system. The presumed redundant definitions are meant to support the user in the use of the macros by providing the relevant set of arguments for a given administration type.

One can distinguish two types of arguments, see Table 2.1. The arguments marked with **bold** font, can be simply references to identifiers defined somewhere else in the model file such as *ka* or *Tlag*. The remaining arguments (cannot be left unassigned) have to be explicitly specified as attributes of the `<Value>` element and subsequently assigned a numerical value or symbol reference, e.g. *cmt* or *amount*. See Section 2.2 for the detailed explanation.

2.2 Encoding rules

To provide the full support for MLXTRAN PK macros, all macros and their (unordered) named arguments have to be encodable, see Table 2.1. Every macro has its own PharmML element, e.g. the basic *concentration* macro with all its arguments, encoded as `<Value>` elements, will be encoded within the `<Compartment>` element. The optional attribute `argument` is assigned a value corresponding to the argument of the macro.

As mentioned above, there are two types of arguments, and the following rules are defined to achieve a consistent macro definition and its interpretation:

- **bold** arguments (can act alone) can be references to symbol identifiers defined in other places in the model, such as `k`, `ka` etc.

- cannot be assigned numerical values or expressions within the macro – all such assignments must be defined outside the `<PKmacros>` element, for example in the `<ParameterModel>` or in the `<ModellingSteps>` section
- if an argument, e.g. 'k', is specified in a macro only, as shown below

```
elimination(cmt=1, k)
```

then a simple `SymbIdRef` is sufficient

```
<Elimination>
    <!-- cmt argument omitted -->>
    <Value>
        <ct:SymbRef blkIdRef="pm1" symbIdRef="k"/>
    </Value>
</Elimination>
```

but it must be defined in a parameter model, here `pm1`.

- if a symbol identifier, here 'k1', is assigned to the key-word argument `k`, e.g.

```
elimination(cmt=1, k=k1)
```

then both the keyword as argument, `k`, and the `symbIdRef`, `k1` are required

```
<Elimination>
    <!-- cmt argument omitted -->>
    <Value argument="k">
        <ct:SymbRef blkIdRef="pm1" symbIdRef="k1"/>
    </Value>
</Elimination>
```

- all *other* arguments (which cannot act alone), e.g. 'cmt' or 'amount' as in the following macro

```
compartment(cmt=1, amount=Ac)
```

have to be specified as attributes and subsequently assigned a numerical value or a `SymbIdRef`, i.e.

```
<Compartment>
    <Value argument="cmt">
        <ct:Int>1</ct:Int>
    </Value>
    <Value argument="amount">
        <ct:SymbRef symbIdRef="Ac"/>
    </Value>
</Compartment>
```

2.3 Defaults, fixed argument values and exceptions

Optional arguments Some arguments are optional. For example `adm` is not required if there is a single route of administration or the `amount` argument in the `<peripheral>` macro. Another example is the macro `iv()` which doesn't require any arguments if the model is one-compartmental. See [3] for more details. However, in all following examples we don't use defaults for the sake of better understanding of the macro concept.

Fixed and model structure dependent arguments All but one macro have a fixed set of arguments. For example the `compartment` can take only these predefined four arguments: {`cmt, amount, volume, concentration`}. The exception is the `peripheral` macro with three fixed arguments {`amount, volume, concentration`} and a variable one, `k_ij`/`k_i_j` (and `k_ji`/`k_j_i`), where `i` and `j` stand for input and output compartment, respectively. These compartments numbers are set according to the model configuration.

For example, in the case of models ADVAN11 or 12, see sections 3.1.6 and 3.1.7 for complete description, we need to set the `kij` arguments names accordingly to the model specification, meaning that macro

```
peripheral(k12, k21, amount=Ap1)
```

will be implemented as shown in the following snippet

```
<Peripheral>
    <Value>
        <ct:SymbRef blkIdRef="pm1" symbIdRef="k12"/>
    </Value>
    <Value>
        <ct:SymbRef blkIdRef="pm1" symbIdRef="k21"/>
    </Value>
    <!-- omitted amount argument -->
</Peripheral>
```

In the special case of `k_ij or k_i_j` attributes, an assignment in the macro formulated as

peripheral(k12, k21 = k_test, amount=Ap1)

will be translated to PharmML as

```
<Peripheral>
    <Value>
        <ct:SymbRef blkIdRef="pm1" symbIdRef="k12"/>
    </Value>
    <Value argument="k21">
        <ct:SymbRef blkIdRef="pm1" symbIdRef="k_test"/>
    </Value>
    <!-- omitted amount argument -->
</Peripheral>
```

Note the difference between the implementation of the two first arguments. The first argument, `k12`, is implemented as before because no assignment is required. In the second case we use the `argument` attribute to inform the target tool about the meaning of the assigned parameter. Therefore, the symbol `k21` is not a parameter itself and `k_test` serves as the transfer parameter `kij` between compartment 2 and 1, which can be defined in the parameter model `pm1` and can be assigned any value in the `<ModellingSteps>` section.

Specifying the administration types A very useful feature of the macro system is that it allows other then numerical identifiers for administration types. This also means that the associate dataset can contain a meaningful string rather then numbers only, see Table 2.2 for the implantation and Table 2.3 how an acceding dataset would look like.

Using integers	Using arbitrary strings

`<IV>`
`  <!-- option 1 -->`
`  <Value argument="adm">`
`  <ct:Int>1</ct:Int>`
`  </Value>`
`  <Value argument="cmt">`
`  <ct:Int>1</ct:Int>`
`  </Value>`
`</IV>`
	`<IV>`
	`  <!-- option 2 -->`
	`  <Value argument="adm">`
	`  <ct:String>bolusIV</ct:String>`
	`  </Value>`
	`  <Value argument="cmt">`
	`  <ct:Int>1</ct:Int>`
	`  </Value>`
	`</IV>`

Table 2.2: Using numerical values or strings to identify the administrations.

MONOLIX				
ID	TIME	AMT	**ADM**	Y
1	6	10	**1**	.
1	9	20	**2**	.
1	18	10	**1**	.
1	33	20	**2**	.
...	...	...	...	...
1	0	.	.	0
1	12	.	.	1.18
...	...	...	...	...

MONOLIX				
ID	TIME	AMT	**ADM**	Y
1	6	10	**bolusIV**	.
1	9	20	**ORAL**	.
1	18	10	**bolusIV**	.
1	33	20	**ORAL**	.
...	...	...	...	...
1	0	.	.	0
1	12	.	.	1.18
...	...	...	...	...

Table 2.3: MONOLIX datasets for the administration type identified using numbers (left) and strings (right) corresponding to the previous Table 2.2.

2.4 Macros and target tools

One of the main reason to introduce the system of macros is that they offer a flexible and efficient way for encoding PK models. The proposed macros are fully compatible with Monolix and its data format, see Appendix B in [5].

Of course, from the perspective of the interoperability platform it is crucial that the models stored in PharmML using macros can be unambiguously translated to NONMEM and other target tools. Especially the compatibility with few predefined PREDPP routines (i.e. ADVAN 1-4 and 10-12) is essential because of their popularity among the NONMEM modellers community.

To understand the required translation rules, the knowledge of relevant NONMEM and MONOLIX features and the dataset formats these target tools use is helpful. For convenience, few most essential differences between these tools/formats from the perspective of the macros and their usage are listed

- In PREDPP library (ADVAN 1-4, 10-12 routines) the *DEPOT* compartment is always considered as the 1^{st} compartment, the central compartment as 2^{nd} and the peripheral compartments as 3^{rd}, 4^{th} etc.
- In contrast, MONOLIX doesn't assign a number to the *DEPOT* compartment at all and the numbering starts usually with the central compartment.
- The CMT column, standard in NONMEM, is not used in Monolix. It uses an ADM column instead.

2.4.1 An illustrative example

Figure 2.1 and datasets in Table 2.4 give us some clues about two alternative interpretations for a model with at least one oral administration, dependent on the target tool. Note, that apart from few predefined models, such as ADVAN1-4 and 10-12, one can assign an arbitrary number to the (first) depot compartment in the NONMEM dataset. This is exactly what happens in this example. While in MONOLIX we would have only two compartments with assigned number to them, all four compartments have numbers assigned in NONMEM, Figure 2.1 (right). This mean that while translating a model from PharmML a transformation of the associated dataset is required. Note, the contrary to models corresponding to one of the ADVAN 1-4 and 10-12 routines, the depot compartment can be assigned an arbitrary number in NONMEM.

It is possible to translate a macro and dataset, utilising the ADM column to identify administrations, used by Monolix into an PREDPP/ADVAN based NONMEM encoding system with data set using the CMT column.

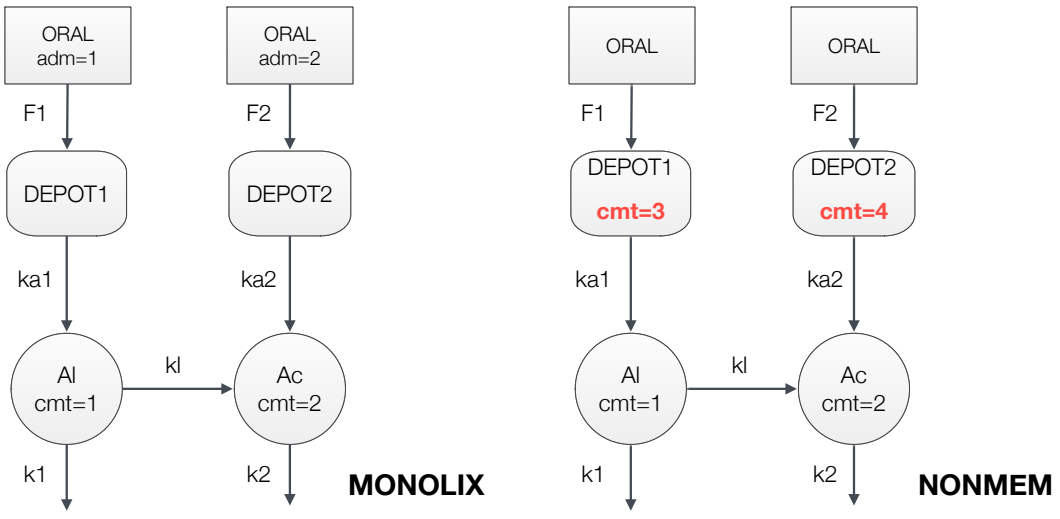


Figure 2.1: Two interpretations for a model with two oral administrations. While MONOLIX (left) doesn't assign a number to the depot compartments, NONMEM (right) does. The translation of macro coded models to target tools requires in this case changes in the dataset as well, see Table 2.4.

MONOLIX				
ID	TIME	AMT	**ADM**	Y
1	6	10	**1**	.
1	9	20	**2**	.
1	18	10	**1**	.
1	33	20	**2**	.
...	...	...	...	...
1	0	.	.	0
1	12	.	.	1.18
...	...	...	...	...

NONMEM				
ID	TIME	AMT	**CMT**	DV
1	6	10	**3**	.
1	9	20	**4**	.
1	18	10	**3**	.
1	33	20	**4**	.
...	...	...	...	...
1	0	.	1	0
1	12	.	1	1.18
...	...	...	...	...

Table 2.4: MONOLIX and NONMEM datasets for the model above, Figure 2.1. The translator has to make sure that the numbers in the ADM column of the former are properly converted to the CMT column in the latter dataset.

2.4.2 Basic translation rules

The following few basic translation rules should be treated merely as a guidance. Also, they are not covering the whole aspect of the model/data handling within the interoperability platform but are limited to the models encoded with macros. A more careful analysis is required to cover the structural model exchange between the key target tools and associated datasets.

Models corresponding to ADVAN 1-4 and 10-12 Fortunately, the translation of models encoded with macros and having their equivalents in routines ADVAN1-4 and 10-12 in PREDPP and their datasets is straightforward. Simply, one has to translate a corresponding macro into its equivalent ADVAN routine and then to rename the ADM column in the MONOLIX data set into CMT, see for more details and examples in Section 3.1.

All other models For every oral administration one new depot compartment needs tho be introduced and assigned a number. The Figure 2.1 and datasets in Table 2.4 can again help to understand what is required.

In this example, `adm=1` and `adm=2` denote two oral administrations. In the NONMEM system, Figure 2.1 (right), '1' and '2' are assigned to the central and peripheral compartment respectively, as in the MONOLIX, so that the next free numbers, '3' and '4', will be assigned to the depot compartments.

Next, the NONMEM dataset needs to be produced based on the information stored in the MONOLIX dataset in such a way that it will match the new compartment numbering scheme with the CMT column replacing the ADM column.

The oral administration, `adm=1`, aims now at the compartment 3, so that '1' in the ADM column will correspond to '3' in the CMT column. Similarly, `adm=2`, aims now at the compartment 4, so that '2' in the ADM column will be translated to '4' in the CMT column.

As additional rules, applicable for all models, one should keep in mind that (1) Y column in a MONOLIX dataset has to be renamed into DV of the NONMEM dataset and (2) the TINF column has to be renamed into RATE and its content recalculated accordingly.

2.4.3 Datasets mapping

The mapping structure as defined in PharmML version 0.6 [6] is used to encode the mapping between macros and the MONOLIX dataset, i.e. the mapping between `adm/type` attributes in the PK macros and the administration type as stored in the ADM column,

- `<TargetMapping>` element with the `blkIdRef` attribute, the latter one because in PharmML multiple structural models are allowed, so we have to specify which model is holding the target compartment definition
- `<Map>` element with
  - `dataSymbol` – attribute denoting the target symbol as encoded in a dataset and
  - new attribute `admNumber` – to identify the target symbol in the model.

The use of these new elements is explained in the Table 2.5.

MONOLIX datasets mapping

```
<ExternalDataSet toolName="Monolix" oid="MLXoid">
    <!-- omitted details -->
    <ColumnMapping>
        <ds:ColumnRef columnIdRef="ADM"/>
        <ds:TargetMapping blkIdRef="sm3">
            <ds:Map dataSymbol="1" admNumber="1"/>
            <ds:Map dataSymbol="2" admNumber="2"/>
            <ds:Map dataSymbol="3" admNumber="3"/>
        </ds:TargetMapping>
    </ColumnMapping>

    <!-- omitted Definition of dataset
        columns such as ID,TIME,AMT,ADM -->
```

Table 2.5: Mappings of MONOLIX type datasets and PK macros.

2.5 Linking macros and <TrialDesign>

<TrialDesign> model provides an alternative to store data records (observations, dosing and covariates) in a tool-independent manner. The <IndividualDosing> element is the place where relevant dosing data can be encoded inline or be referred to from an external datafile. The actual mapping is defined within the <Activity> tag.

A minor extension in the <TrialDesign> is necessary to link the administrations coded in <Activity> elements to the targets in the PK macros and identified using the adm attribute, i.e.

- new value *admType* is added to the inputTarget attribute.

Otherwise, elements defined in the last section will be reused. The Table 2.6 shows the implementations within the <TrialDesign> element in cases when (left) a PK model is expressed using algebraic equation with dose parameter, D, and (right) when using PK macros.

Using dose parameter, D, and algebraic equation for PK model, e.g. $C(t) = f(D, V, k, ...)$ (available in PharmML since version 0.2.1)

```
<Activity oid="actBolusD">
    <Bolus>
        <DoseAmount inputTarget="parameter">
            <ct:SymbRef blkIdRef="sm3" symbIdRef="D"/>
            <ct:Assign>
                <ct:Real>10</ct:Real>
            </ct:Assign>
        </DoseAmount>
        <DosingTimes>
            <!-- e.g. 10 -->
        </DosingTimes>
    </Bolus>
</Activity>
```

Using value *administrationType* for inputTarget and <TargetMapping>/<Map> elements

```
<Activity oid="actBolusMacro">
    <Bolus>
        <DoseAmount inputTarget="admType">
            <ds:TargetMapping blkIdRef="sm3">
                <ds:Map admNumber="2"/>
            </ds:TargetMapping>
            <ct:Assign>
                <ct:Real>10</ct:Real>
            </ct:Assign>
        </DoseAmount>
        <!-- omitted DosingTimes -->
    </Bolus>
</Activity>
```

Table 2.6: Comparison of the link between administration definitions and structural model, dependent on the formulation of the latter. (left) PK model expressed using algebraic equations with dose parameter, D, and when using the PK macros (right).

2.5.1 Connection between macros and the model

The possible outputs of PK macros are amounts, e.g. Ac, and concentrations, e.g. C. A PK model defined by a system of macros will often be connected to a subsequent PD model, which expects the concentration, C, as one of the inputs. Another option is that the output of the PK macros will be mapped directly to the data in the <ObservationModel>.

For the *compartment* macro in question, there are two possibilities, either

- the concentration is defined in the macro

compartment(cmt=1,concentration=C, volume=V)

in which case no output has to be defined explicitly, or

- the alternative form of this macro is used

  compartment(cmt=1,amount=Ac, volume=V)

and if then the concentration is required, a subsequent assignment for C should be provided in MLXTRAN in the *EQUATION* block, i.e.

C = Ac/V

In PharmML, these two cases have to be treated accordingly.

- In the first case only the concentration variable, C, needs to be defined

```
<StructuralModel blkId="sm1">
    <ct:Variable symbolType="real" symbId="C"/>

    <PKmacros>
           <!-- omitted details of the macro
           compartment(cmt=1,concentration=C, volume=V) -->
```

- otherwise the amount variable, Ac, and concentration variable, C, with the $C = Ac/V$ assignment needs to be defined

```
<StructuralModel blkId="sm1">
    <ct:Variable symbolType="real" symbId="Ac"/>
    <ct:Variable symbolType="real" symbId="Cc">
        <ct:Assign>
            <math:Equation>
                <math:Binop op="divide">
                    <ct:SymbRef symbIdRef="Ac"/>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
                </math:Binop>
            </math:Equation>
        </ct:Assign>
    </ct:Variable>

    <PKmacros>
        <Compartment>
            <!-- omitted details of the macro
            compartment(cmt=1,amount=Ac, volume=V) -->
```

If a model using the PK macros is defined properly, following the principles described above, both options make sure that the connection to a subsequent model or the `<ObservationModel>` will be established.

Chapter 3

Examples

We start with the most popular PREDPP, ADVAN1-4 and 10-12, examples and go over later to more complex administrations arrangements.

3.1 Predefined PREDPP models using PK macros and PharmML

This section describes PREDPP library models as encoded in routines ADVAN1-4 and 10-12 and their PK macros implementation.

As explained in Section 2.4 there are differences between NONMEM and MONOLIX and their datasets one has to consider when dealing with PK macros. They manifest themself in that way how the transfer rate constants are numbered/named. Models with more then two compartments and with 1st order input (using *DEPOT* compartment) will have different transfer rate symbols, e.g.:

- ADVAN 4

	PREDPP routines	PK macros
transfer rate constants	k23, k32	k12, k21

- ADVAN 12

	PREDPP routines	PK macros
transfer rate constants	k23, k32	k12, k21
	k24, k42	k13, k31

Models described in this section use the MLXTRAN numbering convention in order to comply with the PK macros notation.

To allow for a lossless translation between PREDPP coded models and PK macros one needs to established well defined translation rules. To achieve this goal and provide help to other translation/converter teams we need to analyse the predefined PREDPP routines ADVAN1-4 & 10-12 and more specifically to

1. understand what they mean
2. dissect the routines into elementary components and finally
3. find equivalent formulation for each such element in the PK macro speak.

The above steps are illustrated in Table 3.1 in an abbreviated form. In columns *Interpretation* and *Elements* first two steps are implemented, the third column provides the according PK macro formulation.

ADVAN routine & default TRANS1	Interpretation	MLXTRAN macro
	one compartment	
ADVAN1 compartment	– 1 compartment	`compartment(cmt=1, amount=Ac, volume=V)`
	– iv bolus administration	`iv(adm=1, cmt=1)`
	– linear elimination	`elimination(cmt=1, k)`
ADVAN2	– 1 compartment	`compartment(cmt=1, amount=Ac, volume=V)`
	– oral administration, 1st order absorption	`oral(adm=1, cmt=1, ka)`
	– linear elimination	`elimination(cmt=1, k)`
ADVAN10	– 1 compartment	`compartment(cmt=1, amount=Ac, volume=V)`
	– iv bolus administration	`iv(adm=1, cmt=1)`
	– saturable elimination	`elimination(cmt=1, Km, Vm)`
	two compartments	
ADVAN3	– 2 compartments one central (1) & one peripheral with linear transfer rates	`compartment(cmt=1, amount=Ac, volume=V)` `peripheral(k12, k21, amount=Ap)`
	– iv bolus administration (into 1)	`iv(adm=1, cmt=1)`
	– linear elimination (from 1)	`elimination(cmt=1, k)`
ADVAN4	– 2 compartments one central (1) & one peripheral with linear transfer rates	`compartment(cmt=1, amount=Ac, volume=V)` `peripheral(k12, k21, amount=Ap)`
	– oral administration, 1st order absorption (into 1)	`oral(adm=1, cmt=1, ka)`
	– linear elimination (from 1)	`elimination(cmt=1, k)`
	three compartments	
ADVAN11	– 3 compartments one central (1) & two peripheral with linear transfer rates	`compartment(cmt=1, amount=Ac, volume=V)` `peripheral(k12, k21, amount=Ap1)` `peripheral(k13, k31, amount=Ap2)`
	– iv bolus administration (into 1)	`iv(adm=1, cmt=1)`
	– linear elimination (from 1)	`elimination(cmt=1, k)`
ADVAN12	– 3 compartments one central (1) & two peripheral with linear transfer rates	`compartment(cmt=1, amount=Ac, volume=V)` `peripheral(k12, k21, amount=Ap1)` `peripheral(k13, k31, amount=Ap2)`
	– oral administration, 1st order absorption (into 1)	`oral(adm=1, cmt=1, ka)`
	– linear elimination (from 1)	`elimination(cmt=1, k)`

Table 3.1: Interpretation and translation of PREDPP models using ADVAN1-4 & 10-12 routines, with default TRANS1 parameterization, into PK macros. Please, note that some attributes are are optional if the context allows it. For example `adm` is not required if there is only one administration route. See [3] and Section 2.3 for more details.

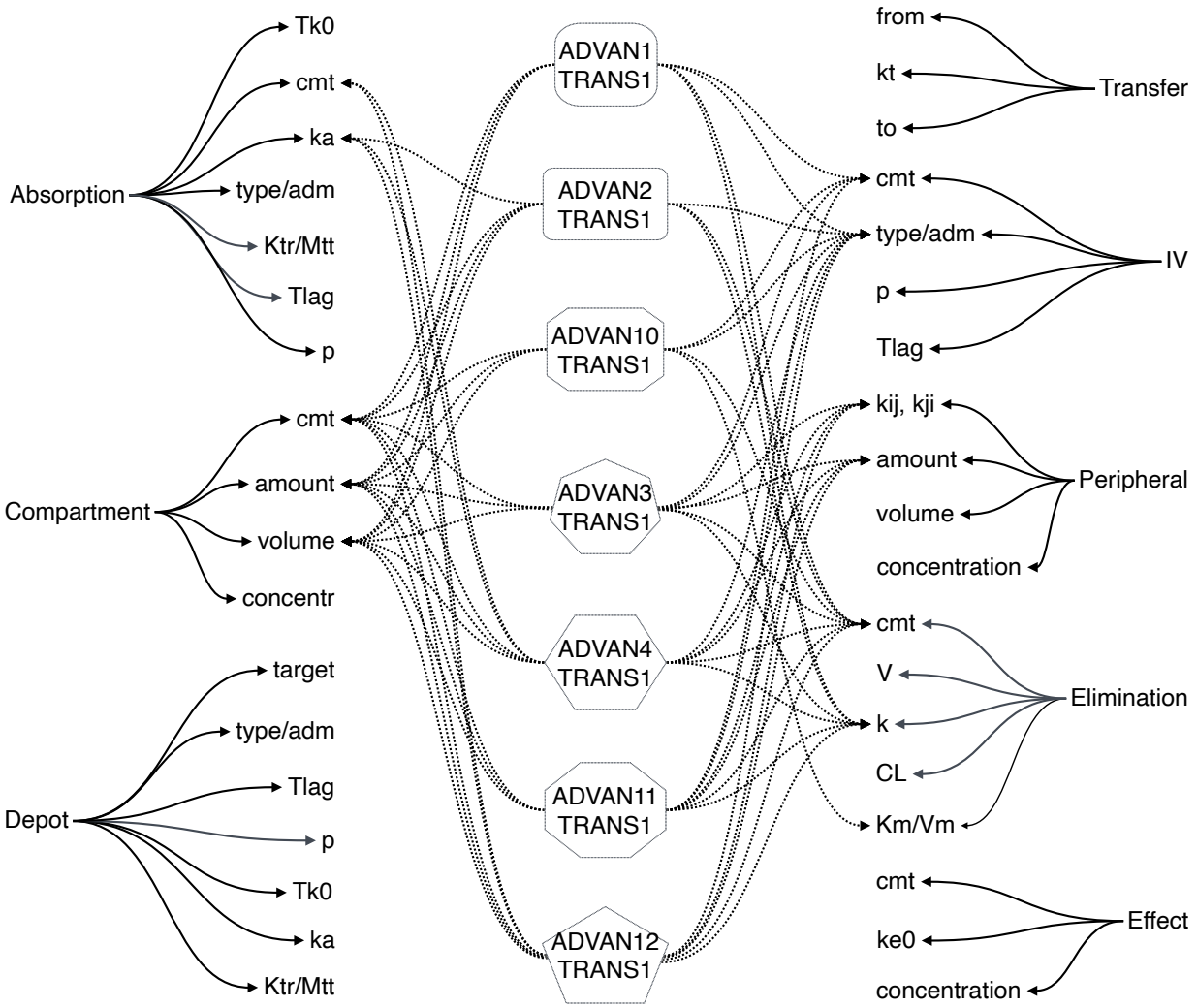


Figure 3.1: The overview of the ADVAN models, with their default parameterization, and their connection to PK macros based on the analysis in Table 3.1. Each ADVAN model can be uniquely mapped to macros and their attributes. For more details see Sections 3.1.1 to 3.1.7.

3.1.1 ADVAN1, TRANS1 – 1-comp IV input

ODE formulation:

$$\frac{dAc}{dt} = -k \times Ac$$

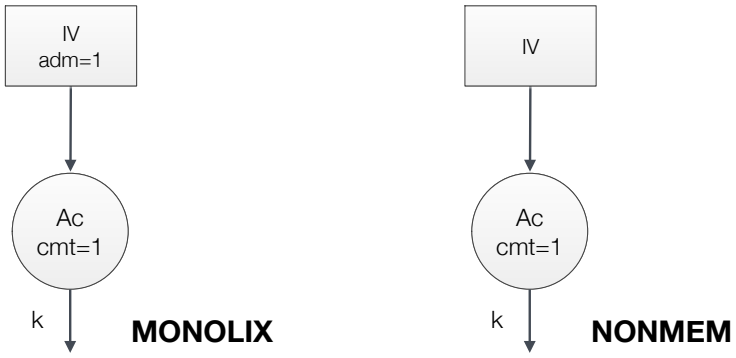


Figure 3.2: ADVAN1 model, in this case compartment numbering is identical.

MONOLIX				
ID	TIME	AMT	**ADM**	Y
1	0	10	**1**	.
1	2	.	.	5
...	...	...	...	...

NONMEM				
ID	TIME	AMT	**CMT**	DV
1	0	10	**1**	.
1	2	.	1	5
...	...	...	...	...

Table 3.2: MONOLIX and NONMEM datasets for the ADVAN1 model.

PK macro
input = {V, k}
PK:
compartment(cmt=1, amount=Ac, volume=V)
iv(adm=1, cmt=1)
elimination(cmt=1, k)

Table 3.3: PK macros for the ADVAN1 model, as shown in Figure 3.9 (left).

PharmML code[1]

```
<ParameterModel blkId="pm1">
    <!-- no IIV assumed for simplicity -->
    <SimpleParameter symbId="k"/>
    <SimpleParameter symbId="V"/>
</ParameterModel>

<StructuralModel blkId="sm1">
    <ct:Variable symbolType="real" symbId="Ac"/>

    <PKmacros>
        <Compartment>
            <Value argument="cmt">
                <ct:Int>1</ct:Int>
            </Value>
            <Value argument="amount">
                <ct:SymbRef symbIdRef="Ac"/>
            </Value>
            <Value argument="volume">
                <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
            </Value>
```

[1]Only the ADVAN1 code example contains the parameter model, `pm1`, in which all model parameters has to be defined. It is omitted in the remaining cases for simplicity.

```
            </Compartment>

            <IV>
                <Value argument="adm">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value argument="cmt">
                    <ct:Int>1</ct:Int>
                </Value>
            </IV>

            <Elimination>
                <Value argument="cmt">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k"/>
                </Value>
            </Elimination>
        </PKmacros>
    </StructuralModel>
```

3.1.2 ADVAN2, TRANS1 – 1-comp 1st order input

ODE formulation:

$$\frac{dAd}{dt} = -ka \times Ad$$
$$\frac{dAc}{dt} = ka \times Ad - k \times Ac$$

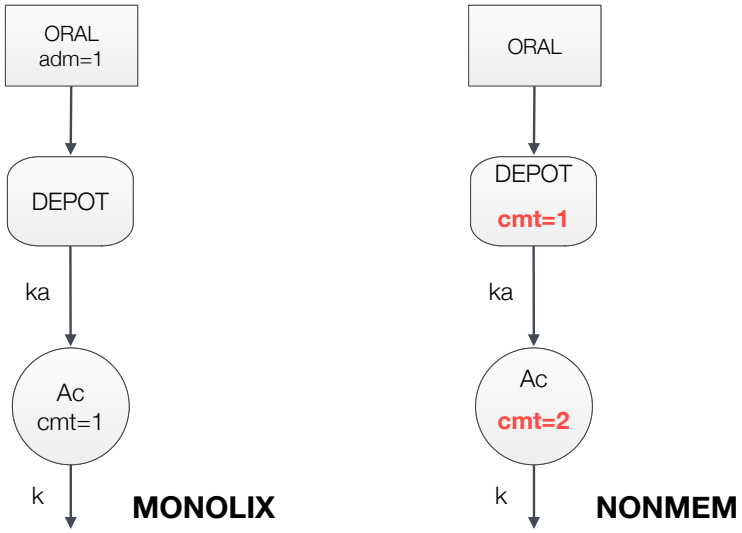


Figure 3.3: ADVAN2 model, with compartment numbering dependent on the target tool.

MONOLIX				
ID	TIME	AMT	**ADM**	Y
1	0	10	**1**	.
1	2	.	.	5
...	...	...	...	...

NONMEM				
ID	TIME	AMT	**CMT**	DV
1	0	10	**1**	.
1	2	.	2	5
...	...	...	...	...

Table 3.4: MONOLIX and NONMEM datasets for the ADVAN2 model.

PharmML code:

PK macro
input = {ka, V, k}
PK:
compartment(cmt=1, amount=Ac, volume=V)
oral(adm=1, cmt=1, ka)
elimination(cmt=1, k)

Table 3.5: PK macros for the ADVAN2 model, as shown in Figure 3.3 (left).

```
<StructuralModel blkId="sm2">
    <ct:Variable symbolType="real" symbId="Ac"/>

    <PKmacros>
        <Compartment>
            <Value argument="cmt">
                <ct:Int>1</ct:Int>
            </Value>
            <Value argument="amount">
                <ct:SymbRef symbIdRef="Ac"/>
            </Value>
            <Value argument="volume">
                <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
            </Value>
        </Compartment>

        <Oral>
            <Value argument="adm">
                <ct:Int>1</ct:Int>
            </Value>
            <Value argument="cmt">
                <ct:Int>1</ct:Int>
            </Value>
            <Value>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="ka"/>
            </Value>
        </Oral>

        <Elimination>
            <Value argument="cmt">
                <ct:Int>1</ct:Int>
            </Value>
            <Value>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="k"/>
            </Value>
        </Elimination>
    </PKmacros>
</StructuralModel>
```

3.1.3 ADVAN3, TRANS1 – 2-comp IV input

ODE formulation:

$$\frac{dAc}{dt} = -k_{12} \times Ac + k_{21} \times Ap - k \times Ac$$
$$\frac{dAp}{dt} = k_{12} \times Ac - k_{21} \times Ap$$

MONOLIX				
ID	TIME	AMT	**ADM**	Y
1	0	10	**1**	.
1	2	.	.	5
...	...	...	...	...

NONMEM				
ID	TIME	AMT	**CMT**	DV
1	0	10	**1**	.
1	2	.	2	5
...	...	...	...	...

Table 3.6: MONOLIX and NONMEM datasets for the ADVAN3 model.

PharmML code:

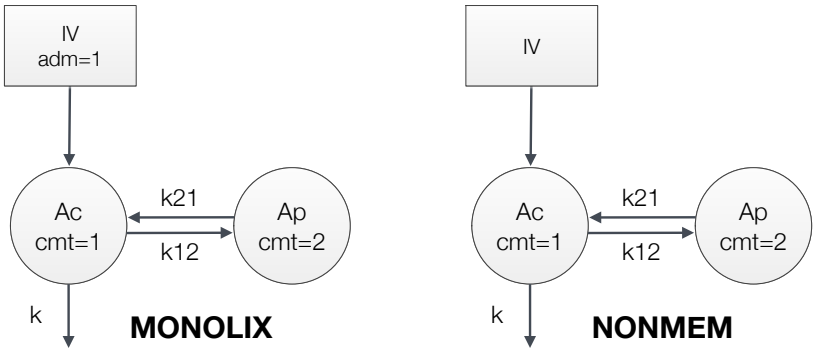


Figure 3.4: ADVAN3 model, in this case compartment numbering is identical.

PK macro
input = {V, k, k12, k21}
PK:
compartment(cmt=1, amount=Ac, volume=V)
peripheral(k12, k21, amount=Ap)
iv(adm=1, cmt=1)
elimination(cmt=1, k)

Table 3.7: PK macros for the ADVAN3 model, as shown in Figure 3.4 (left).

```
<StructuralModel blkId="sm3">

    <ct:Variable symbolType="real" symbId="Ac"/>
    <ct:Variable symbolType="real" symbId="Ap"/>

    <PKmacros>
        <Compartment>
            <Value argument="cmt">
                <ct:Int>1</ct:Int>
            </Value>
            <Value argument="amount">
                <ct:SymbRef symbIdRef="Ac"/>
            </Value>
            <Value argument="volume">
                <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
            </Value>
        </Compartment>

        <Peripheral>
            <Value>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="k12"/>
            </Value>
            <Value>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="k21"/>
            </Value>
            <Value argument="amount">
                <ct:SymbRef symbIdRef="Ap"/>
            </Value>
        </Peripheral>

        <IV>
            <Value argument="adm">
                <ct:Int>1</ct:Int>
            </Value>
            <Value argument="cmt">
                <ct:Int>1</ct:Int>
            </Value>
        </IV>

        <Elimination>
            <Value argument="cmt">
```

```
                <ct:Int>1</ct:Int>
            </Value>
            <Value>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="k"/>
            </Value>
        </Elimination>
    </PKmacros>
</StructuralModel>
```

3.1.4 ADVAN4, TRANS1 – 2-comp 1st order input

ODE formulation:

$$\frac{dAd}{dt} = -ka \times Ad$$
$$\frac{dAc}{dt} = ka \times Ad - k_{12} \times Ac + k_{21} \times Ap - k \times Ac$$
$$\frac{dAp}{dt} = k_{12} \times Ac - k_{21} \times Ap$$

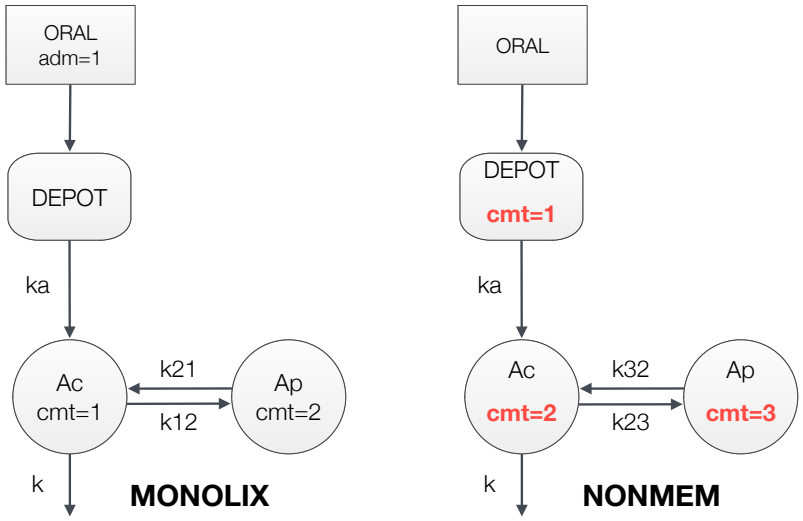


Figure 3.5: ADVAN4 model, with compartment numbering dependent on the target tool. Note, that not only the compartment numbers are different in NONMEM coded model, the rate constants names are different as well.

MONOLIX				
ID	TIME	AMT	**ADM**	Y
1	0	10	**1**	.
1	2	.	.	5
...	...	...	...	...

NONMEM				
ID	TIME	AMT	**CMT**	DV
1	0	10	**1**	.
1	2	.	2	5
...	...	...	...	...

Table 3.8: MONOLIX and NONMEM datasets for the ADVAN4 model.

PharmML code:

```
<StructuralModel blkId="sm4">
    <ct:Variable symbolType="real" symbId="Ac"/>
    <ct:Variable symbolType="real" symbId="Ap"/>
    <ct:Variable symbolType="real" symbId="Cc">
        <ct:Assign>
            <math:Equation>
                <math:Binop op="divide">
                    <ct:SymbRef symbIdRef="Ac"/>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
                </math:Binop>
            </math:Equation>
```

PK macro
input = {ka, V, k, k12, k21}
PK:
compartment(cmt=1, amount=Ac, volume=V)
peripheral(k12, k21, amount=Ap)
oral(adm=1, cmt=1, ka)
elimination(cmt=1, k)

Table 3.9: PK macros for the ADVAN4 model, as shown in Figure 3.5 (left).

```
            </ct:Assign>
        </ct:Variable>

        <PKmacros>
            <Compartment>
                <Value argument="cmt">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value argument="amount">
                    <ct:SymbRef symbIdRef="Ac"/>
                </Value>
                <Value argument="volume">
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
                </Value>
            </Compartment>

            <Peripheral>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k12"/>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k21"/>
                </Value>
                <Value argument="amount">
                    <ct:SymbRef symbIdRef="Ap"/>
                </Value>
            </Peripheral>

            <Oral>
                <Value argument="adm">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value argument="cmt">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="ka"/>
                </Value>
            </Oral>

            <Elimination>
                <Value argument="cmt">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k"/>
                </Value>
            </Elimination>
        </PKmacros>
    </StructuralModel>
```

3.1.5 ADVAN10, TRANS1 – 1-comp IV input with saturable elimination

ODE formulation:

$$\frac{dAc}{dt} = -\frac{Vm \times Ac}{Km + Ac}$$

PharmML code:

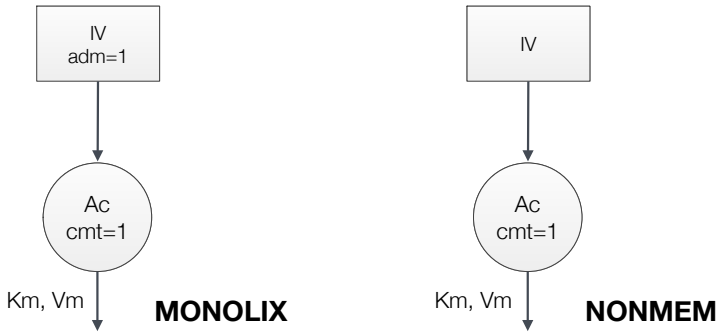


Figure 3.6: ADVAN10 model, in this case compartment numbering is identical.

MONOLIX				
ID	TIME	AMT	**ADM**	Y
1	0	10	**1**	.
1	2	.	.	5
...	...	...	...	...

NONMEM				
ID	TIME	AMT	**CMT**	DV
1	0	10	**1**	.
1	2	.	1	5
...	...	...	...	...

Table 3.10: MONOLIX and NONMEM datasets for the ADVAN10 model.

PK macro
input = {V, Km, Vm}
PK:
compartment(cmt=1, amount=Ac, volume=V)
iv(adm=1, cmt=1)
elimination(cmt=1, Km, Vm)

Table 3.11: PK macros for the ADVAN10 model, as shown in Figure 3.6 (left).

```
<StructuralModel blkId="sm10">
    <ct:Variable symbolType="real" symbId="Ac"/>
    <ct:Variable symbolType="real" symbId="Cc">
        <ct:Assign>
            <math:Equation>
                <math:Binop op="divide">
                    <ct:SymbRef symbIdRef="Ac"/>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
                </math:Binop>
            </math:Equation>
        </ct:Assign>
    </ct:Variable>

    <PKmacros>
        <Compartment>
            <Value argument="cmt">
                <ct:Int>1</ct:Int>
            </Value>
            <Value argument="amount">
                <ct:SymbRef symbIdRef="Ac"/>
            </Value>
            <Value argument="volume">
                <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
            </Value>
        </Compartment>

        <IV>
            <Value argument="adm">
                <ct:Int>1</ct:Int>
            </Value>
            <Value argument="cmt">
                <ct:Int>1</ct:Int>
            </Value>
        </IV>
```

```
            <Elimination>
                <Value argument="cmt">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="Km"/>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="Vm"/>
                </Value>
            </Elimination>
        </PKmacros>
    </StructuralModel>
```

3.1.6 ADVAN11, TRANS1 – 3-comp IV input

ODE formulation:

$$\begin{aligned}\frac{dAc}{dt} &= -k_{12} \times Ac + k_{21} \times Ap1 - k_{13} \times Ac \\ &\quad + k_{31} \times Ap2 - k \times Ac \\ \frac{dAp1}{dt} &= k_{12} \times Ac - k_{21} \times Ap1 \\ \frac{dAp2}{dt} &= k_{13} \times Ac - k_{31} \times Ap2\end{aligned}$$

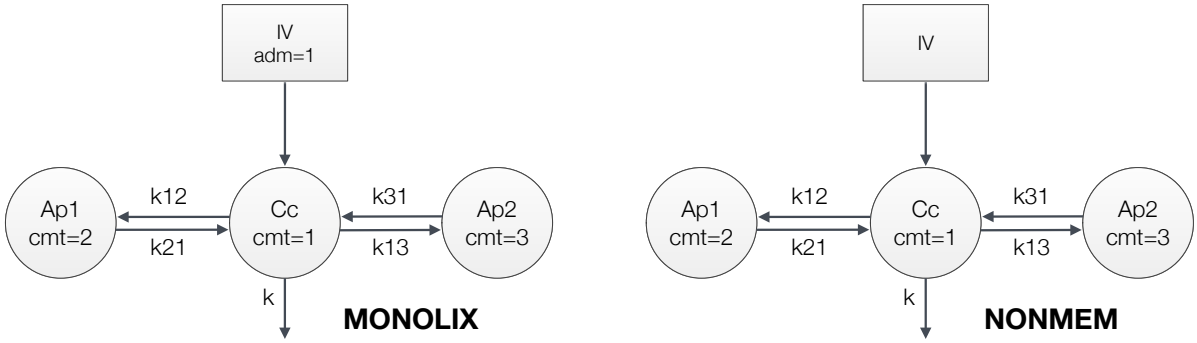


Figure 3.7: ADVAN11 model, in this case compartment numbering is identical.

MONOLIX				
ID	TIME	AMT	**ADM**	Y
1	0	10	**1**	.
1	2	.	.	5
...	...	...	...	...

NONMEM				
ID	TIME	AMT	**CMT**	DV
1	0	10	**1**	.
1	2	.	1	5
...	...	...	...	...

Table 3.12: MONOLIX and NONMEM datasets for the ADVAN11 model.

PK macro
input = {V, k, k12, k21, k13, k31}
PK:
compartment(cmt=1, amount=Ac, volume=V)
peripheral(k12, k21, amount=Ap1)
peripheral(k13, k31, amount=Ap2)
iv(adm=1, cmt=1)
elimination(cmt=1, k)

Table 3.13: PK macros for the ADVAN11 model, as shown in Figure 3.7 (left).

PharmML code:

```
<StructuralModel blkId="sm11">
    <ct:Variable symbolType="real" symbId="Ac"/>
    <ct:Variable symbolType="real" symbId="Ap1"/>
    <ct:Variable symbolType="real" symbId="Ap2"/>
    <ct:Variable symbolType="real" symbId="Cc">
        <ct:Assign>
            <math:Equation>
                <math:Binop op="divide">
                    <ct:SymbRef symbIdRef="Ac"/>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
                </math:Binop>
            </math:Equation>
        </ct:Assign>
    </ct:Variable>

    <PKmacros>
        <Compartment>
            <Value argument="cmt">
                <ct:Int>1</ct:Int>
            </Value>
            <Value argument="amount">
                <ct:SymbRef symbIdRef="Ac"/>
            </Value>
            <Value argument="volume">
                <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
            </Value>
        </Compartment>

        <Peripheral>
            <Value>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="k12"/>
            </Value>
            <Value>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="k21"/>
            </Value>
            <Value argument="amount">
                <ct:SymbRef symbIdRef="Ap1"/>
            </Value>
        </Peripheral>

        <Peripheral>
            <Value>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="k13"/>
            </Value>
            <Value>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="k31"/>
            </Value>
            <Value argument="amount">
                <ct:SymbRef symbIdRef="Ap2"/>
            </Value>
        </Peripheral>

        <IV>
            <Value argument="adm">
                <ct:Int>1</ct:Int>
            </Value>
            <Value argument="cmt">
                <ct:Int>1</ct:Int>
            </Value>
        </IV>

        <Elimination>
            <Value argument="cmt">
                <ct:Int>1</ct:Int>
            </Value>
            <Value>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="k"/>
            </Value>
        </Elimination>
    </PKmacros>
</StructuralModel>
```

3.1.7 ADVAN12, TRANS1 – 3-comp 1st order input

ODE formulation:

$$\begin{aligned}
\frac{dAd}{dt} &= -ka \times Ad \\
\frac{dAc}{dt} &= ka \times Ad - k_{12} \times Ac + k_{21} \times Ap1 - k_{13} \times Ac \\
&\quad + k_{31} \times Ap2 - k \times Ac \\
\frac{dAp1}{dt} &= k_{12} \times Ac - k_{21} \times Ap1 \\
\frac{dAp2}{dt} &= k_{13} \times Ac - k_{31} \times Ap2
\end{aligned}$$

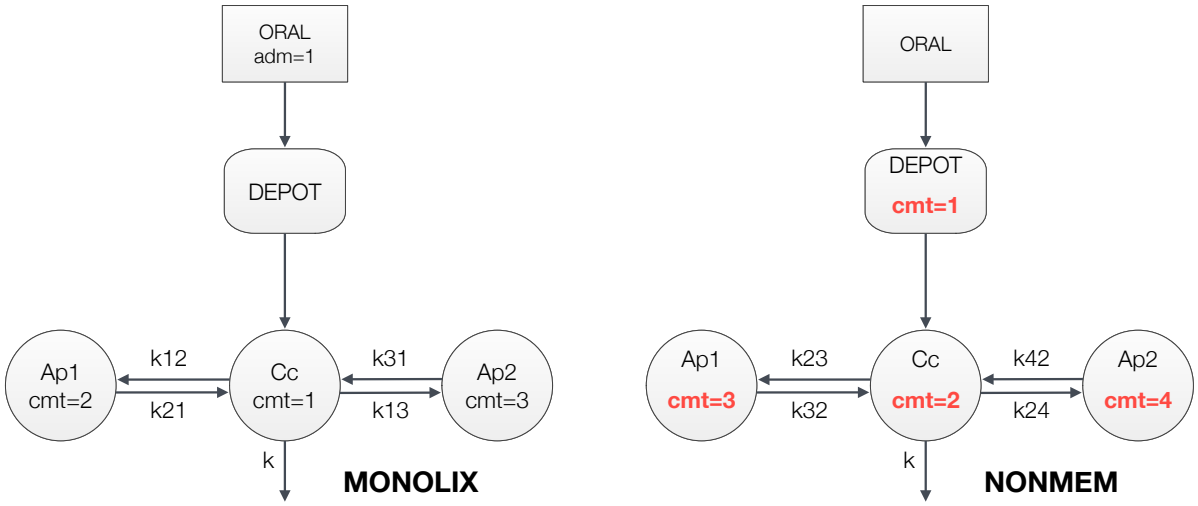


Figure 3.8: ADVAN12 model, with compartment numbering dependent on the target tool. Note, that not only the compartment numbers are different in NONMEM coded model, the rate constants names are different as well.

MONOLIX				
ID	TIME	AMT	**ADM**	Y
1	0	10	**1**	.
1	2	.	.	5
...	...	...	...	...

NONMEM				
ID	TIME	AMT	**CMT**	DV
1	0	10	**1**	.
1	2	.	2	5
...	...	...	...	...

Table 3.14: MONOLIX and NONMEM datasets for the ADVAN12 model.

PK macro
input = {ka, V, k, k12, k21, k13, k31}
PK:
compartment(cmt=1, amount=Ac, volume=V)
peripheral(k12, k21, amount=Ap1)
peripheral(k13, k31, amount=Ap2)
oral(adm=1, cmt=1, ka)
elimination(cmt=1, k)

Table 3.15: PK macros for the ADVAN12 model, as shown in Figure 3.8 (left).

PharmML code:

```
<StructuralModel blkId="sm12">
    <ct:Variable symbolType="real" symbId="Ac"/>
    <ct:Variable symbolType="real" symbId="Ap1"/>
```

```
        <ct:Variable symbolType="real" symbId="Ap2"/>
        <ct:Variable symbolType="real" symbId="Cc">
            <ct:Assign>
                <math:Equation>
                    <math:Binop op="divide">
                        <ct:SymbRef symbIdRef="Ac"/>
                        <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
                    </math:Binop>
                </math:Equation>
            </ct:Assign>
        </ct:Variable>

        <PKmacros>
            <Compartment>
                <Value argument="cmt">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value argument="amount">
                    <ct:SymbRef symbIdRef="Ac"/>
                </Value>
                <Value argument="volume">
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
                </Value>
            </Compartment>

            <Peripheral>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k12"/>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k21"/>
                </Value>
                <Value argument="amount">
                    <ct:SymbRef symbIdRef="Ap1"/>
                </Value>
            </Peripheral>

            <Peripheral>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k13"/>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k31"/>
                </Value>
                <Value argument="amount">
                    <ct:SymbRef symbIdRef="Ap2"/>
                </Value>
            </Peripheral>

            <Oral>
                <Value argument="adm">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value argument="cmt">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="ka"/>
                </Value>
            </Oral>

            <Elimination>
                <Value argument="cmt">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k"/>
                </Value>
            </Elimination>
        </PKmacros>
    </StructuralModel>
```

3.2 Alternative model parameterizations

In case of routine others then the default TRANS1 the procedure is quite straightforward. The reparametrization, called *Reparameterization Lines* in the NONMEM manual [1], can be defined in the `<ParameterModel>` leaving the PK macros in the `<StructuralModel>` unchanged. This is presented using the basic ADVAN1 and more complex ADVAN11 routines.

ADVAN1 with TRANS2

Following table shows the two allowed parameterizations for ADVAN1 and the according formula.

TRANS1	TRANS2	Formula
K Rate constant of elimination	CL Clearance V Volume of distribution	K=CL/V

Table 3.16: ADVAN1 parameters with TRANS1 and TRANS2.

Table 3.17 shows the PK macros for the two options

ADVAN1, TRANS1	ADVAN1, TRANS2
compartment(cmt=1, amount=Ac, volume=V) iv(cmt=1) elimination(cmt=1, k)	compartment(cmt=1, amount=Ac, volume=V) iv(cmt=1) elimination(cmt=1, Cl)

Table 3.17: PK macros for ADVAN1 using two different parameterizations.

The reparameterization is done in the `<ParameterModel>`, `pm1`:

```
<ParameterModel blkId="pm1">

    <!-- Reparameterization Lines for k, Cl, V -->
    <SimpleParameter symbId="k">
        <ct:Assign>
            <math:Equation>
                <math:Binop op="divide">
                    <ct:SymbRef symbIdRef="Cl"/>
                    <ct:SymbRef symbIdRef="V"/>
                </math:Binop>
            </math:Equation>
        </ct:Assign>
    </SimpleParameter>

</ParameterModel>

<StructuralModel blkId="sm1">
    <ct:Variable symbolType="real" symbId="Ac"/>
    <PKmacros>
           <!-- omitted, identical to ADVAN1, TRANS1 -->
    </PKmacros>
</StructuralModel>
```

ADVAN11 with TRANS4

Following table shows the two allowed parameterizations for ADVAN11 and the according re-parameterisation formulas. and the following table shows the PK macros for the two routines

PharmML code – modified from section 3.1.6. In this case the reparameterization is done in the parameter model `pm1`, here example for one parameter:

```
<ParameterModel blkId="pm1">

    <!-- Reparameterization Lines for k12, Q, V1 -->
    <SimpleParameter symbId="k12">
        <ct:Assign>
            <math:Equation>
```

TRANS1	TRANS4	Formula
K Rate constant of elimination	CL Clearance	K=CL/V1
K12 Rate constant from central to periph. 1	V1 Central volume	K12=Q2/V1
K21 Rate constant from periph. 1 to central	Q2 Intercompartmental clearance 1	K21=Q2/V2
K13 Rate constant from central to periph. 2	V2 Peripheral volume 1	K13=Q3/V1
K31 Rate constant from periph. 2 to central	Q3 Intercompartmental clearance 2	K31=Q3/V3
	V3 Peripheral volume 2	V3=V3

Table 3.18: ADVAN11 parameters with TRANS1 and TRANS4.

ADVAN11, TRANS1	ADVAN11, TRANS4
compartment(cmt=1, amount=Ac, volume=V)	compartment(cmt=1, amount=Ac, volume=V)
peripheral(k12, k21, amount=Ap1)	peripheral(k12 = Q2/V1, k21 = Q2/V2, amount=Ap1)
peripheral(k13, k31, amount=Ap2)	peripheral(k13 = Q3/V1, k31 = Q3/V3, amount=Ap2)
iv(cmt=1)	iv(cmt=1)
elimination(cmt=1, k)	elimination(cmt=1, k = Cl/V1)

Table 3.19: PK macros for ADVAN11 using two different parameterizations.

```
                    <math:Binop op="divide">
                        <ct:SymbRef symbIdRef="Q2"/>
                        <ct:SymbRef symbIdRef="V1"/>
                    </math:Binop>
                </math:Equation>
            </ct:Assign>
        </SimpleParameter>
        <!-- omitted parameters k21, k13, k31, V3 and k, Cl, V1 -->

    </ParameterModel>

    <StructuralModel blkId="sm1">
        <ct:Variable symbolType="real" symbId="Ac"/>
        <ct:Variable symbolType="real" symbId="Ap1"/>
        <ct:Variable symbolType="real" symbId="Ap2"/>
        <PKmacros>
               <!-- omitted, identical to ADVAN11, TRANS1 -->
        </PKmacros>
    </StructuralModel>
```

3.3 Complex PK macro examples

Additional examples have been encoded in PharmML in order to further validate the schema and to test the implementation of more complex PK macros.

Note that the first two of the following models, C1 (versions C1a and C1b) and C2, can be treated as equivalent to that discussed in Sections 3.1.2, 3.1.7, and modelled by the ADVAN2 ands ADVAN12 routines, respectively, with an additional IV administration.

Importantly, the according dataset transformation reduces to the renaming of the ADM and CMT columns **but** only if the oral administration is treated as the first one, i.e. with attribute `adm=1`. Otherwise a renumbering is required as well. Examples C1a and C1b illustrate this.

3.3.1 Example C1a

The following example is from [4], pages 70-74. This model can be modelled by the ADVAN2 routine, with an additional IV administration. Because the oral administration is labeled as the first one, attribute `adm=1`, no renumbering is required when transforming the data and renaming the ADM to CMT column is sufficient to get the suitable NONMEM data set.

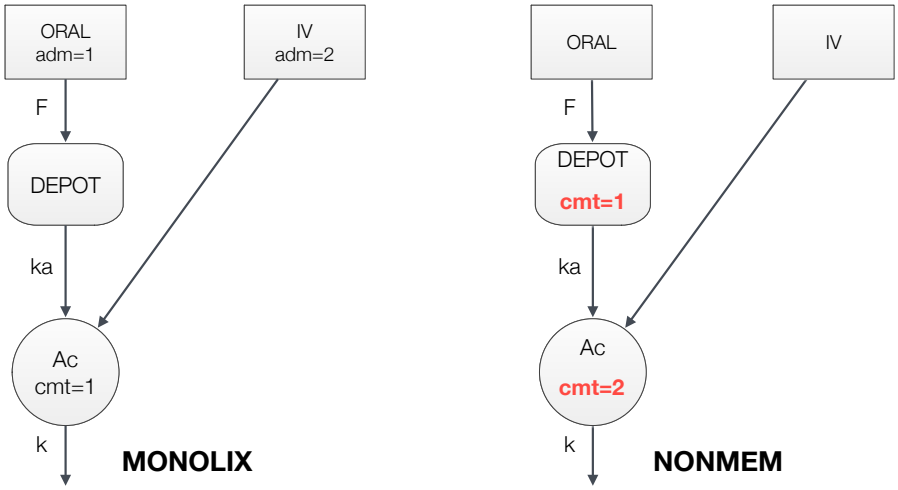


Figure 3.9: C1a model, with compartment numbering dependent on the target tool.

MONOLIX				
ID	TIME	AMT	**ADM**	Y
1	0	2.24	**2**	.
1	1	.	.	142
1	2	.	.	54.9
...	...	...	...	...
1	6	7	**1**	.
1	7	.	.	192
1	8	.	.	141
...	...	...	...	...
2	0	2.73	**2**	.
2	1	.	.	176
...	...	...	...	...

NONMEM				
ID	TIME	AMT	**CMT**	DV
1	0	2.24	**2**	.
1	1	.	2	142
1	2	.	2	54.9
...	...	...	...	...
1	6	7	**1**	.
1	7	.	2	192
1	8	.	2	141
...	...	...	...	...
2	0	2.73	**2**	.
2	1	.	2	176
...	...	...	...	...

Table 3.20: NONMEM and MONOLIX datasets for the C1a model. Because of the equivalence to ADVAN2 no renumbering is required when transforming the data and renaming the ADM to CMT column.

PharmML code:

```
<StructuralModel blkId="smC1a">
    <ct:Variable symbolType="real" symbId="Ap"/>
    <ct:Variable symbolType="real" symbId="Cc">
        <ct:Assign>
            <math:Equation>
                <math:Binop op="divide">
```

PK macro
input = {F, ka, V, k}
PK:
compartment(cmt=1, amount=Ac)
oral(adm=1, cmt=1, ka, p=F)
iv(adm=2, cmt=1)
elimination(cmt=1, k)
Cc = Ac/V

Table 3.21: PK macros for the C1a model, as shown in Figure 3.9 (left).

```
                    <ct:SymbRef symbIdRef="Ac"/>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
                </math:Binop>
            </math:Equation>
        </ct:Assign>
    </ct:Variable>

    <PKmacros>
        <!-- compartment(cmt=1, amount=Ac) -->
        <Compartment>
            <Value argument="cmt">
                <ct:Int>1</ct:Int>
            </Value>
            <Value argument="amount">
                <ct:SymbRef symbIdRef="Ac"/>
            </Value>
        </Compartment>

        <!-- oral(adm=1, cmt=1, ka, p=F) -->
        <Oral>
            <Value argument="adm">
                <ct:Int>1</ct:Int>
            </Value>
            <Value argument="cmt">
                <ct:Int>1</ct:Int>
            </Value>
            <Value>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="ka"/>
            </Value>
            <Value argument="p">
                <ct:SymbRef blkIdRef="pm1" symbIdRef="F"/>
            </Value>
        </Oral>

        <!-- iv(adm=2, cmt=1) -->
        <IV>
            <Value argument="adm">
                <ct:Int>2</ct:Int>
            </Value>
            <Value argument="cmt">
                <ct:Int>1</ct:Int>
            </Value>
        </IV>

        <!-- elimination(cmt=1, k) -->
        <Elimination>
            <Value argument="cmt">
                <ct:Int>1</ct:Int>
            </Value>
            <Value>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="k"/>
            </Value>
        </Elimination>
    </PKmacros>
</StructuralModel>
```

3.3.2 Example C1b

This example is very similar to the previous one with the difference in the administration order. IV is labeled as the first one, with attribute `adm=1`, the oral as the second, `adm=2`, Figure 3.10. The model can still be modelled by the ADVAN2 routine, with an additional IV administration, but the dataset translation will not be limited to the renaming of column names only as in C1a case. Now the numbers in columns ADM and CMT will not be identical, see Table 3.22.

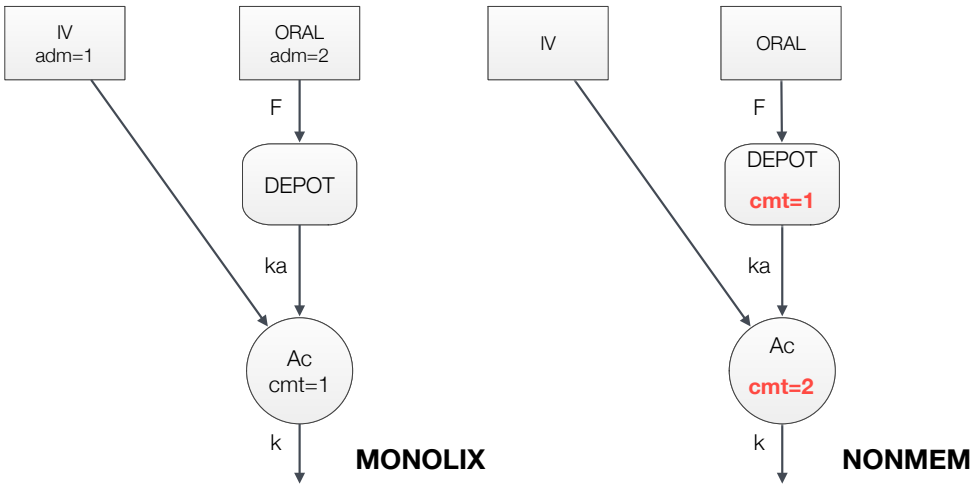


Figure 3.10: C1b model, with compartment numbering dependent on the target tool.

MONOLIX				
ID	TIME	AMT	**ADM**	Y
1	0	2.24	**1**	.
1	1	.	.	142
1	2	.	.	54.9
...	...	...	...	...
1	6	7	**2**	.
1	7	.	.	192
1	8	.	.	141
...	...	...	...	...
2	0	2.73	**1**	.
2	1	.	.	176
...	...	...	...	...

NONMEM				
ID	TIME	AMT	**CMT**	DV
1	0	2.24	**2**	.
1	1	.	2	142
1	2	.	2	54.9
...	...	...	...	...
1	6	7	**1**	.
1	7	.	2	192
1	8	.	2	141
...	...	...	...	...
2	0	2.73	**2**	.
2	1	.	2	176
...	...	...	...	...

Table 3.22: NONMEM and MONOLIX datasets for the C1b model. In contrast to the previous case, a renumbering is required when transforming the data and renaming the ADM to CMT column.

PK macro
input = {F, ka, V, k}
PK:
compartment(cmt=1, amount=Ac)
oral(adm=2, cmt=1, ka, p=F)
iv(adm=1, cmt=1)
elimination(cmt=1, k)
Cc = Ac/V

Table 3.23: PK macros for the C1b model, as shown in Figure 3.10 (left).

PharmML code is identical to the previous example, and will not be shown, except the amended `adm` attribute assignments, `adm=2` for oral and `adm=1` for IV administration, see Table 3.23.

3.3.3 Example C2

Modified from [3]. This model can be modelled by the ADVAN12 routine, with an additional IV administration.

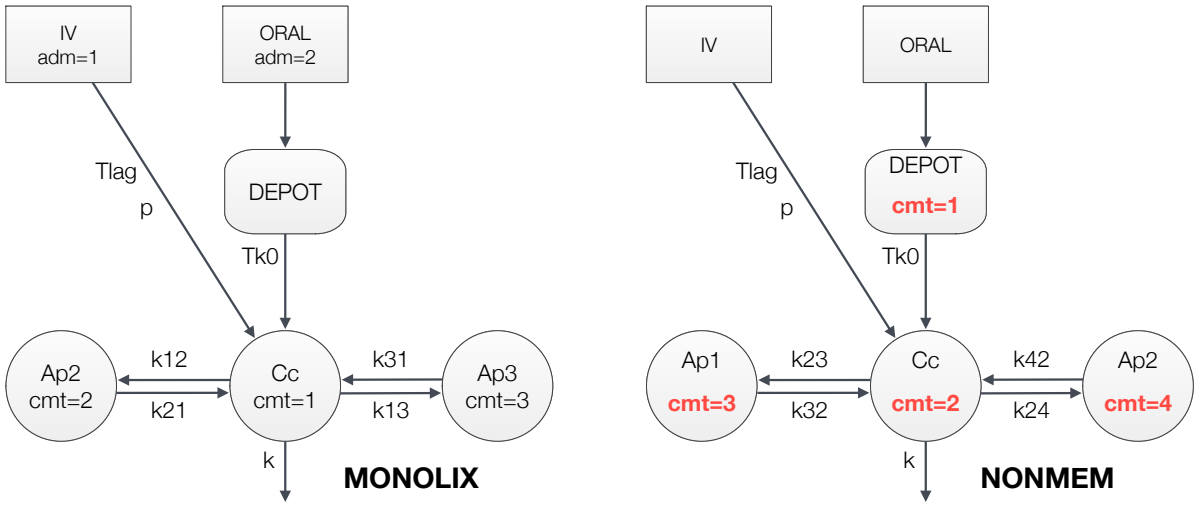


Figure 3.11: C2 model, with compartment numbering dependent on the target tool.

MONOLIX					
ID	TIME	AMT	**ADM**	TINF	Y
1	6	10	**1**	2	.
1	9	20	**2**	.	.
1	18	10	**1**	2	.
1	30	10	**1**	2	.
1	33	20	**2**	.	.
...	...	...	...	...	...
1	0	.	.	.	0
1	6	.	.	.	0
1	12	.	.	.	1.18
...	...	...	...	...	...

NONMEM					
ID	TIME	AMT	**CMT**	RATE	DV
1	6	10	**1**	5	.
1	9	20	**2**	.	.
1	18	10	**1**	5	.
1	30	10	**1**	5	.
1	33	20	**2**	.	.
...	...	...	...	...	...
1	0	.	1	.	0
1	6	.	1	.	0
1	12	.	1	.	1.18
...	...	...	...	...	...

Table 3.24: NONMEM and MONOLIX datasets for the C2 model.

PK macro
input = {V, alpha, beta}
PK:
compartment(cmt=1, volume=V, concentration=Cc)
iv(adm=1, cmt=1, p=0.1, Tlag=t/(t + 10))
oral(adm=2, cmt=1, Tk0=0.1)
elimination(cmt=1, k=0.2)
peripheral(k12=0.6, k21=0.8, amount=Ap2)
peripheral(k13=0.6+alpha, k31=0.8+beta, amount=Ap3)

Table 3.25: PK macros for the C2 model, as shown in Figure 3.11 (left).

The following code is split in three parts to improve the readability. The first contains the `<ParameterModel>`, the second the `<StructuralModel>` with `<PKmacros>` and the third the `<ModellingSteps>`.

PharmML code part 1:

```
<ParameterModel blkId="pm1">
   <SimpleParameter symbId="V"/>
   <SimpleParameter symbId="k"/>
   <SimpleParameter symbId="p"/>
   <SimpleParameter symbId="alpha"/>
   <SimpleParameter symbId="k13">
```

```
            <ct:Assign>
                <math:Equation>
                    <math:Binop op="plus">
                        <ct:Real>0.6</ct:Real>
                        <ct:SymbRef symbIdRef="alpha"/>
                    </math:Binop>
                </math:Equation>
            </ct:Assign>
        </SimpleParameter>

        <SimpleParameter symbId="Tk0"/>

        <SimpleParameter symbId="k12"/>
        <SimpleParameter symbId="k21"/>

        <SimpleParameter symbId="beta"/>
        <SimpleParameter symbId="k31">
            <ct:Assign>
                <math:Equation>
                    <math:Binop op="plus">
                        <ct:Real>0.8</ct:Real>
                        <ct:SymbRef symbIdRef="beta"/>
                    </math:Binop>
                </math:Equation>
            </ct:Assign>
        </SimpleParameter>
    </ParameterModel>
```

PharmML code part 2:

```
    <StructuralModel blkId="smC2">
        <ct:Variable symbolType="real" symbId="Cc"/>
        <ct:Variable symbolType="real" symbId="Ap2"/>
        <ct:Variable symbolType="real" symbId="Ap3"/>

        <ct:Variable symbolType="real" symbId="Tlag">
            <ct:Assign>
                <math:Equation>
                    <math:Binop op="divide">
                        <ct:SymbRef symbIdRef="t"/>
                        <math:Binop op="plus">
                            <ct:SymbRef symbIdRef="t"/>
                            <ct:Real>10</ct:Real>
                        </math:Binop>
                    </math:Binop>
                </math:Equation>
            </ct:Assign>
        </ct:Variable>

        <PKmacros>
            <Compartment>
                <Value argument="cmt">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value argument="volume">
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
                </Value>
                <Value argument="concentration">
                    <ct:SymbRef symbIdRef="Cc"/>
                </Value>
            </Compartment>

            <IV>
                <Value argument="adm">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value argument="cmt">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="p"/>
                </Value>
                <Value>
```

```
                    <ct:SymbRef symbIdRef="Tlag"/>
                </Value>
            </IV>

            <Oral>
                <Value argument="adm">
                    <ct:Int>2</ct:Int>
                </Value>
                <Value argument="cmt">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="Tk0"/>
                </Value>
            </Oral>

            <Elimination>
                <Value argument="cmt">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k"/>
                </Value>
            </Elimination>

            <Peripheral>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k12"/>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k21"/>
                </Value>
                <Value argument="amount">
                    <ct:SymbRef symbIdRef="Ap2"/>
                </Value>
            </Peripheral>

            <Peripheral>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k13"/>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k31"/>
                </Value>
                <Value argument="amount">
                    <ct:SymbRef symbIdRef="Ap3"/>
                </Value>
            </Peripheral>

        </PKmacros>
    </StructuralModel>
```

PharmML code part 3:

```
<ModellingSteps xmlns="http://www.pharmml.org/2013/03/ModellingSteps">

    <NONMEMdataSet oid="NMoid">
        <!-- omitted details -->
    </NONMEMdataSet>

    <EstimationStep oid="estTask1">
        <TargetToolReference>
            <ct:OidRef oidRef="NMoid"/>
        </TargetToolReference>

        <ParametersToEstimate>
            <ParameterEstimation>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="k12"/> <!-- fixed -->
                <InitialEstimate fixed="true">
                    <ct:Real>0.6</ct:Real>
                </InitialEstimate>
            </ParameterEstimation>
            <ParameterEstimation>
```

```
                <ct:SymbRef blkIdRef="pm1" symbIdRef="k21"/> <!-- fixed -->
                <InitialEstimate fixed="true">
                    <ct:Real>0.8</ct:Real>
                </InitialEstimate>
            </ParameterEstimation>
            <ParameterEstimation>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="Tk0"/> <!-- fixed -->
                <InitialEstimate fixed="true">
                    <ct:Real>0.1</ct:Real>
                </InitialEstimate>
            </ParameterEstimation>
            <ParameterEstimation>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="p"/> <!-- fixed -->
                <InitialEstimate fixed="true">
                    <ct:Real>0.1</ct:Real>
                </InitialEstimate>
            </ParameterEstimation>

            <ParameterEstimation>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="a"/>
                <InitialEstimate>
                    <ct:Real>1</ct:Real>
                </InitialEstimate>
            </ParameterEstimation>

            <ParameterEstimation>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
                <InitialEstimate>
                    <ct:Real>10</ct:Real>
                </InitialEstimate>
            </ParameterEstimation>

            <!-- omitted other parameters -->
        </ParametersToEstimate>

        <!-- omitted other elements -->
    </EstimationStep>
```

3.3.4 Example C3

From [3]. Figure 3.12 is illustrating this model in two versions.

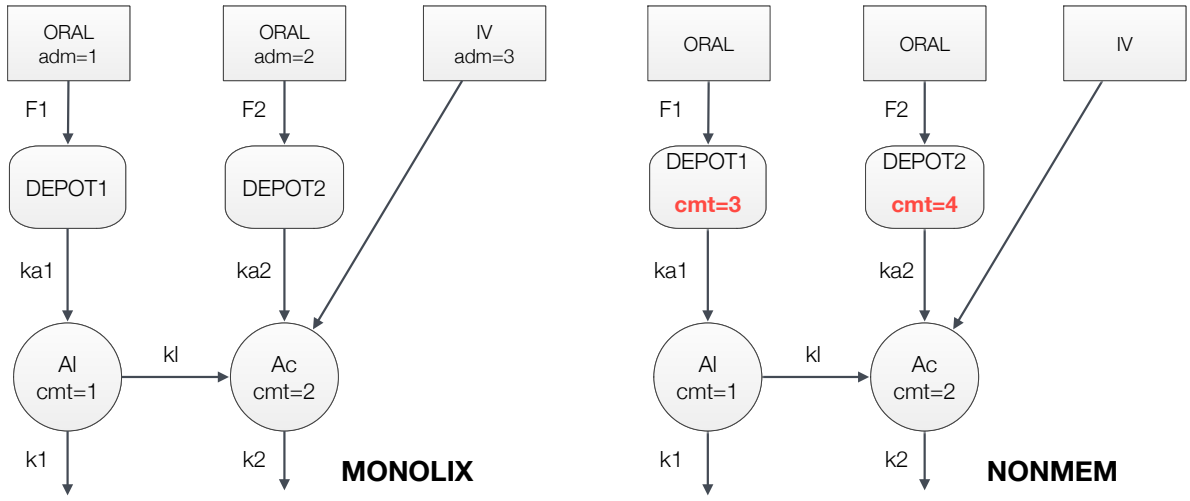


Figure 3.12: C3 model, with compartment numbering dependent on the target tool.

PharmML code:

```
    <StructuralModel blkId="smC3">
        <ct:Variable symbolType="real" symbId="Al"/>
        <ct:Variable symbolType="real" symbId="Ac"/>
```

MONOLIX					
ID	TIME	AMT	**ADM**	TINF	Y
1	6	10	**1**	.	.
1	9	20	**2**	.	.
1	12	30	**3**	2	.
1	18	10	**1**	.	.
1	30	10	**1**	.	.
1	33	20	**2**	.	.
1	36	30	**3**	2	.
...	...	...	...	...	...
1	0	.	.	.	0
1	6	.	.	.	0
1	12	.	.	.	1.18
...	...	...	...	...	...

NONMEM					
ID	TIME	AMT	**CMT**	RATE	DV
1	6	10	**3**	.	.
1	9	20	**4**	.	.
1	12	30	**2**	**15**	.
1	18	10	**3**	.	.
1	30	10	**3**	.	.
1	33	20	**4**	.	.
1	36	30	**2**	**15**	.
...	...	...	...	...	...
1	0	.	1	.	0
1	6	.	1	.	0
1	12	.	1	.	1.18
...	...	...	...	...	...

Table 3.26: NONMEM and MONOLIX datasets for the C3 model.

PK macro
input = {F1, F2, ka1, ka2, k1, k2, kl, V}
PK:
compartment(cmt=1, amount=Al)
compartment(cmt=2, amount=Ac, volume=V)
oral(adm=1, cmt=1, ka1, p=F1)
oral(adm=2, cmt=2, ka2, p=F2)
iv(adm=3, cmt=2)
transfer(from=1, to=2, kt=kl)
elimination(cmt=1, k1)
elimination(cmt=2, k2)
Cc = Ac/V

Table 3.27: PK macros for the C3 model, as shown in Figure 3.12 (left).

```
<ct:Variable symbolType="real" symbId="Cc">
    <ct:Assign>
        <math:Equation>
            <math:Binop op="divide">
                <ct:SymbRef symbIdRef="Ac"/>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
            </math:Binop>
        </math:Equation>
    </ct:Assign>
</ct:Variable>

<PKmacros>
    <Compartment>
        <Value argument="cmt">
            <ct:Int>1</ct:Int>
        </Value>
        <Value argument="amount">
            <ct:SymbRef symbIdRef="Al"/>
        </Value>
    </Compartment>

    <Compartment>
        <Value argument="cmt">
            <ct:Int>2</ct:Int>
        </Value>
        <Value argument="amount">
            <ct:SymbRef symbIdRef="Ac"/>
        </Value>
        <Value argument="volume">
            <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
        </Value>
    </Compartment>

    <Oral>
        <Value argument="adm">
            <ct:Int>1</ct:Int>
```

```
                </Value>
                <Value argument="cmt">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="ka1"/>
                </Value>
                <Value argument="p">
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="F1"/>
                </Value>
            </Oral>

            <Oral>
                <Value argument="adm">
                    <ct:Int>2</ct:Int>
                </Value>
                <Value argument="cmt">
                    <ct:Int>2</ct:Int>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="ka2"/>
                </Value>
                <Value argument="p">
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="F2"/>
                </Value>
            </Oral>

            <IV>
                <Value argument="adm">
                    <ct:Int>3</ct:Int>
                </Value>
                <Value argument="cmt">
                    <ct:Int>2</ct:Int>
                </Value>
            </IV>

            <Transfer>
                <Value argument="from">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value argument="to">
                    <ct:Int>2</ct:Int>
                </Value>
                <Value argument="kt">
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="kl"/>
                </Value>
            </Transfer>

            <Elimination>
                <Value argument="cmt">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k1"/>
                </Value>
            </Elimination>

            <Elimination>
                <Value argument="cmt">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k2"/>
                </Value>
            </Elimination>

        </PKmacros>
    </StructuralModel>
```

3.3.5 Example C4

The following example is from 'Four models' document [2]. Figure 3.13 is illustrating this model in two versions.

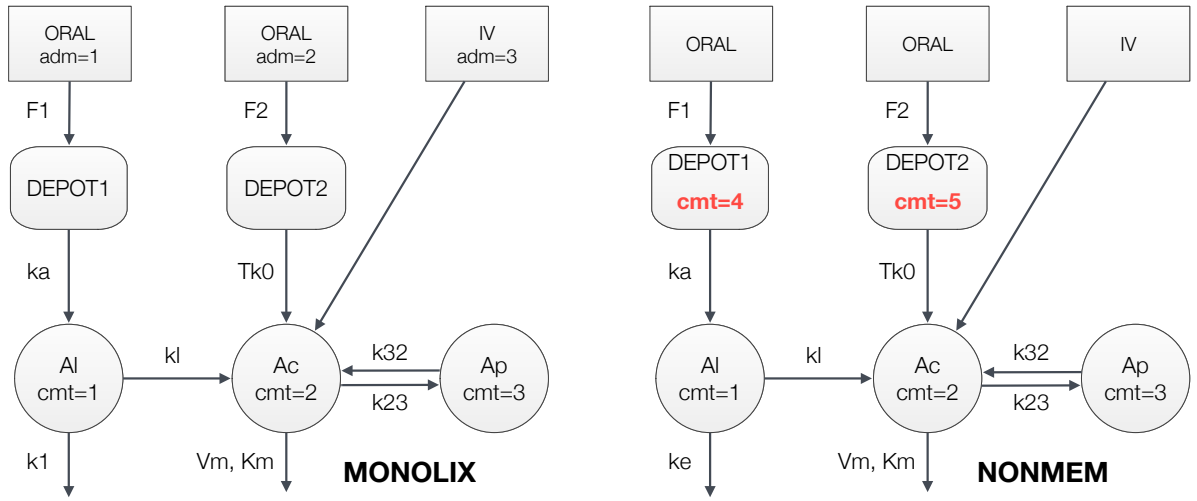


Figure 3.13: C4 model, with compartment numbering dependent on the target tool.

MONOLIX					
ID	TIME	AMT	**ADM**	TINF	Y
1	6	10	**1**	.	.
1	9	20	**2**	.	.
1	12	30	**3**	2	.
1	18	10	**1**	.	.
1	30	10	**1**	.	.
1	33	20	**2**	.	.
1	36	30	**3**	2	.
...	...	...	...	...	...
1	0	.	.	.	0
1	6	.	.	.	0
1	12	.	.	.	1.18
...	...	...	...	...	...

NONMEM					
ID	TIME	AMT	**CMT**	RATE	DV
1	6	10	**4**	.	.
1	9	20	**5**	.	.
1	12	30	**2**	**15**	.
1	18	10	**4**	.	.
1	30	10	**4**	.	.
1	33	20	**5**	.	.
1	36	30	**2**	**15**	.
...	...	...	...	...	...
1	0	.	1	.	0
1	6	.	1	.	0
1	12	.	1	.	1.18
...	...	...	...	...	...

Table 3.28: NONMEM and MONOLIX datasets for the C4 model.

PK macro
input = {F1, F2, ka, Tk0, kl, V, k, Vm, Km}
PK:
compartment(cmt=1, amount=Al)
compartment(cmt=2, amount=Ac)
oral(adm=1, cmt=1, ka, p=F1)
oral(adm=2, cmt=2, Tk0, p=F2)
iv(adm=3, cmt=2)
transfer(from=1, to=2, kt=kl)
peripheral(k23, k32, amount=Ap)
elimination(cmt=1, k)
elimination(cmt=2, Km, Vm)
Cc = Ac/V

Table 3.29: PK macros for the C4 model, as shown in Figure 3.13 (left).

PharmML code:

```
<StructuralModel blkId="smC4">
   <ct:Variable symbolType="real" symbId="Al"/>
   <ct:Variable symbolType="real" symbId="Ac"/>
```

```
<ct:Variable symbolType="real" symbId="Cc">
    <ct:Assign>
        <math:Equation>
            <math:Binop op="divide">
                <ct:SymbRef symbIdRef="Ac"/>
                <ct:SymbRef blkIdRef="pm1" symbIdRef="V"/>
            </math:Binop>
        </math:Equation>
    </ct:Assign>
</ct:Variable>

<PKmacros>
    <Compartment>
        <Value argument="cmt">
            <ct:Int>1</ct:Int>
        </Value>
        <Value argument="amount">
            <ct:SymbRef symbIdRef="Al"/>
        </Value>
    </Compartment>

    <Compartment>
        <Value argument="cmt">
            <ct:Int>2</ct:Int>
        </Value>
        <Value argument="amount">
            <ct:SymbRef symbIdRef="Ac"/>
        </Value>
    </Compartment>

    <Oral>
        <Value argument="adm">
            <ct:Int>1</ct:Int>
        </Value>
        <Value argument="cmt">
            <ct:Int>1</ct:Int>
        </Value>
        <Value>
            <ct:SymbRef blkIdRef="pm1" symbIdRef="ka"/>
        </Value>
        <Value argument="p">
            <ct:SymbRef blkIdRef="pm1" symbIdRef="F1"/>
        </Value>
    </Oral>

    <Oral>
        <Value argument="adm">
            <ct:Int>2</ct:Int>
        </Value>
        <Value argument="cmt">
            <ct:Int>2</ct:Int>
        </Value>
        <Value>
            <ct:SymbRef blkIdRef="pm1" symbIdRef="Tk0"/>
        </Value>
        <Value argument="p">
            <ct:SymbRef blkIdRef="pm1" symbIdRef="F2"/>
        </Value>
    </Oral>

    <IV>
        <Value argument="type">
            <ct:Int>3</ct:Int>
        </Value>
        <Value argument="cmt">
            <ct:Int>2</ct:Int>
        </Value>
    </IV>

    <Transfer>
        <Value argument="from">
            <ct:Int>1</ct:Int>
```

```
                </Value>
                <Value argument="to">
                    <ct:Int>2</ct:Int>
                </Value>
                <Value argument="kt">
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="kl"/>
                </Value>
            </Transfer>

            <Peripheral>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k23"/>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k32"/>
                </Value>
                <Value argument="amount">
                    <ct:SymbRef symbIdRef="Ap"/>
                </Value>
            </Peripheral>

            <Elimination>
                <Value argument="cmt">
                    <ct:Int>1</ct:Int>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="k"/>
                </Value>
            </Elimination>

            <Elimination>
                <Value argument="cmt">
                    <ct:Int>2</ct:Int>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="Vm"/>
                </Value>
                <Value>
                    <ct:SymbRef blkIdRef="pm1" symbIdRef="Km"/>
                </Value>
            </Elimination>
        </PKmacros>
    </StructuralModel>
```

Bibliography

[1] Stuart L. Beal, Lewis B. Sheiner, Alison J. Boeckmann, and Robert J. Bauer. NONMEM User's Guides. (1989-2009). Technical report, Icon Development Solutions, Ellicott City, MD, USA, June 2009.

[2] Marc Lavielle. Four models. Technical report, INRIA Saclay, April 3, 2014.

[3] Marc Lavielle and Lixoft Team. MLXTRAN, The model coding language for Monolix. Technical report, INRIA Saclay & Lixoft, May 2014.

[4] Lixoft. Monolix 4.3 - Model MLXTRAN Tutorial. Technical report, Lixoft, 2014.

[5] Lixoft. Monolix 4.3.2 - User Guide. Technical report, Lixoft, May 2014.

[6] Maciej J Swat, Sarala M. Wimalaratne, Niels R Kristensen, Florent Yvon, Stuart Moodie, and N. Le Novère. Pharmacometrics Markup Language (PharmML), Language Specification for Version 0.6. January 2015.