Tucuxi Drug Model File Specification

Release 1.0

Tucuxi dev team

Feb 18, 2022

Contents

1

Drug description file

Contents

This chapter presents the way medical drugs have to be described. Tucuxi is flexible in terms of handling various medical drugs (or substances). The drugs are defined within XML files, and can be edited thanks to the special Tucuxi drug editor.

The suffix of a Tucuxi Drug Description file is *.tdd*, and by convention, the name of the file shall be similar to the drug model Id.

1.1 General elements

This section lists some useful elements used everywhere in the drug model file.

1.1.1 Units

The units in Tucuxi consist of a base and a multiplier. For example in *ug/l*, *g/l* is the base, and *u* the multiplier, which express a unit of *micrograms per liter*. The convertion factors are used to convert the data produced by the models of Tucuxi. The molar mass is used to give the user the choice to use moles instead of the default units.

1.1.2 stdAprioriValue

In various parts of the drug model, elements can have a default value, and an apriori value calculated with the help of the patient covariates. Every time such a pattern is used, the element is of type *stdAprioriValue*.

Table 1: stdAprioriValue content

Tag name	Format	Occ.	Description
<standardValue>	double	1:1	The default value
<aprioriComputation>	*Operation*	0:1	The operations to calculate the a priori value

When such an element is used, the software will use default values for calculation involving the typical patient. In case of a priori and a posteriori calculations, it will try to apply the operations on each of such value, in order to better fit the patient reality.

1.1.3 Operation

The operations are used to modify the values of the parameters in accordance with the patient's covariates, as well as other values (targets, domain validity, . . .). They are used to compute the *a priori* parameters and can be of three types:

- softFormula
- hardFormula
- multiFormula

These three options are mutually exclusive, so the possible styles of operations are:

- SoftFormula

```
<operation>
  <softFormula>
    <inputs>
      <input>
        <id>bodyweight</id>
        <type>double</type>
      </input>
    </inputs>
    <code><![CDATA[
        return bodyweight < 100.0 and bodyweight > 0.0;
            ]]>
    </code>
```

(continues on next page)

(continued from previous page)

```
  </softFormula>
  <comments/>
</operation>
```

- hardFormula

```
<operation>
  <hardFormula>formulaId</hardFormula>
  <comments/>
</operation>
```

Where the formulaId can be:

- IdealBodyWeight
- BodySurfaceArea
- eGFR_CockcroftGaultGeneral
- OperationEGFRCockcroftGaultGeneral
- eGFR_CockcroftGaultIBW
- eGFR_CockcroftGaultAdjIBW
- GFR_MDRD
- GFR_CKD_EPI
- eGFR_Schwartz
- GFR_Jelliffe
- eGFR_SalazarCorcoran
- direct
- sum2
- multiFormula (not yet supported)

```
<operation>
  <multiFormula>
    <softFormula>
      <inputs>
        <input>
          <id>bodyweight</id>
          <type>double</type>
        </input>
      </inputs>
      <code><![CDATA[
          return bodyweight < 100.0 and bodyweight > 0.0;
              ]]>
      </code>
    </softFormula>
    <hardFormula>formulaId</hardFormula>
  </multiFormula>
  <comments/>
</operation>
```

Tag name	Format	Occ.	Description
<operation>			Description of an operation
____<softFormula>		0:1	An Javascript operation
________<inputs>		1:1	The list of required inputs
____________<input>		1:∞	An input for the formula
________________<id>		1:1	The Id of the required input for the formula
________________<type>		1:1	The type of data : double, int or bool
________<code>	*Code*	1:∞	The operation formula
____<hardFormula>	string	0:1	A hardcoded operation
____<multiFormula>		0:1	A multi-operation formula
________<…>		1:∞	Any of softFormula and hardFormula
____<comments>		1:1	Comments about the operation

An operation can be used in many elements. For instance it is used in parameters, targets, covariates, in order to calculate a priori values.

A formula can use the value of any global or drug-specific covariate. To do so, you must use the covariate's ID , as shown above with *bodyweight*. You can also use any of the drug's parameters, using its ID followed by the *_population* keyword, as in *V_population*. The formula should simply return a value, nothing else is mandatory.

The formula must always be surrounded by the *<![CDATA[* and *]]* markers. The language used to express the formula is based on Javascript and supports a subset of it. It is to be noted that the online editor takes care of the addition of the markers.

A formula must always return a value.

When an operation can contain a list of formula the computing engine shall try to apply the first formula. If there are missing covariates for such formula, then the second formula is tried, and so on, until a valid formula is found.

The list of inputs is important and should contain all the inputs required by the formula (for a soft formula). The type should obviously be correct. In case the input is a covariate, the type should be the same type as the covariate, and in case of a parameter the type is *double*. The editor does not check if the input list is correct or not, so be careful.

Code

A formula is an operation returning a value, based on some inputs. The content of the element is then a source code in the correct format.

The source code must always be surrounded by the *<![CDATA[* and *]]* markers. The language used to express the formula is based on Javascript and supports a subset of it. A formula must always return a value.

The following mathematical functions are available within scripts:

```
Math.E()
Math.log(a)
Math.log10(a)
Math.exp(a)
Math.pow(a,b)

Math.sqr(a)
Math.sqrt(a)

Math.abs(a)
Math.round(a)
Math.min(a,b)
Math.max(a,b)
Math.range(x,a,b)
Math.sign(a)

Math.PI()
Math.toDegrees(a)
Math.toRadians(a)
Math.sin(a)
Math.asin(a)
Math.cos(a)
Math.acos(a)
Math.tan(a)
Math.atan(a)
Math.sinh(a)
Math.asinh(a)
Math.cosh(a)
Math.acosh(a)
Math.tanh(a)
Math.atanh(a)
```

Such function must always be written with the *Math.* as prefix. Example:

```
newValue = Math.pow(aValue, anotherValue) + Math.exp(yetAnotherValue);
```

Structures such as *if/then/else* are supported, as in the following example:

```
if (sex > 0.5) {
  aValue = 12;
}
else {
  aValue = 10;
}
```

Examples

Computation of the PK parameter V based on sex, for Imatinib:

```
theta2 = V_population;
theta8 = 46.2;
tvv = theta2 + theta8 * sex - theta8 * (1 - sex);
return tvv;
```

Computation of the PK parameter CL based on sex, bodyweight, age and Gist, for Imatinib:

```
theta1 = CL_population;
theta4 = 5.42;
theta5 = 1.49;
theta6 = -5.81;
theta7 = -0.806;

MEANBW = 70;
FBW = (bodyweight - MEANBW) / MEANBW;

MEANAG = 50;
FAGE = (age - MEANAG) / MEANAG;

if (gist)
  PATH = 1;
else
  PATH = 0;

MALE = sex;

TVCL = theta1 + theta4 * FBW + theta5 * MALE-theta5 * (1-MALE) + theta6 * FAGE + theta7␣
↪* PATH - theta7 * (1 - PATH);

return TVCL;
```

And the list of inputs for that specific computation:

```
<inputs>
    <input>
        <id>CL_population</id>
        <type>double</type>
    </input>
    <input>
        <id>bodyweight</id>
        <type>double</type>
    </input>
    <input>
        <id>age</id>
        <type>double</type>
    </input>
    <input>
        <id>gist</id>
        <type>bool</type>
    </input>
```

(continues on next page)

(continued from previous page)

```
    <input>
        <id>sex</id>
        <type>double</type>
    </input>
</inputs>
```

1.1.4 Comments

Before explaining all the specific fields, a word on comments is required, as the <comments> tag can be found at different places of the file. A comment has the following structure:

```
<comments>
  <comment lang='en'>This work is based on the paper nice_paper, that can be found here.
↪</comment>
  <comment lang='fr'>La description de ce médicament est basée sur ce nice_paper, qui␣
↪peut être trouvé ici.</comment>
</comments>
```

Tag name	Format	Occ.	Description
<comments>		1:1	List of translated comments
___<comment lang=’xx’>	string	0:∞	Comment for a specific language

It contains as many <comment> tags as required. Each <comment> tag has an attribute *lang* defining the language of the comment, enabling multi-language support for the description of the medical drugs.

A validation element allows to specify a validity function to check another element value. It is used in covariates and parameters.

It is based on an operation, and embeds an error message that can serve to display relevant information to the user.

```
<validation> <!-- pourrait être une contrainte -->
  <errorMessage><text lang="fr"></text></errorMessage>
  <operation>
    <softFormula>
      <inputs>
        <input>
          <id>bodyweight</id>
          <type>double</type>
        </input>
      </inputs>
      <formula><![CDATA[
```

(continues on next page)

(continued from previous page)

```
          return bodyweight < 300 and bodyweight > 0;
          ]]>
      </formula>
    </softFormula>
    <comments/>
  </operation>
  <comments/>
</validation>
```

Table 2: head content

Tag name	Format	Occ.	Description
<validation>		1:1	Description of a validation
___<errorMessage>		1:1	List of error messages
______<text lang=’xx’>	string	0:∞	Message for a specific language
___<operation>	*Operation*	1:1	The checking operation
___<comments>	*Comments*	1:1	Comments about validation

1.2 Model

The global structure of the XML file is the following:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<?xml-stylesheet href="drugsmodel2.xsl" type="text/xsl" ?>
<model version='1.0'
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="drug2.xsd">
  <history></history>
  <head></head>
  <drugModel></drugModel>
</model>
```

Tag name	Format	Occ.	Description
<history>	*History*	1:1	History of the file
<head>	*Head*	1:1	General information
<drugModel>	*DrugModel*	1:1	Everything needed for computation

The XML format is described in the file `drug2.xsd` which is used by Tucuxi in order to check the structure correctness before loading the drug description file.

1.3 History

Used by *Model*.

The history of the file is described inside the <history> tag. A standard history looks like this:

```xml
<history>
  <revisions>
    <revision>
      <revisionAction>creation</revisionAction>
      <revisionAuthorName>John Doe</revisionAuthorName>
      <institution>Name of his institution</institution>
      <email>john@doe.com</email>
      <date>2014-07-17</date>
      <comments/>
    </revision>
    <revision>
      <revisionAction>modification</revisionAction>
      <revisionAuthorName>Jane Doe</revisionAuthorName>
      <institution>Name of her institution</institution>
      <email>jane@doe.com</email>
      <date>2014-07-30</date>
      <comments/>
    </revision>
  </revisions>
</history>
```

Tag name	Format	Occ.	Description
<revisions>		1:1	List of revisions
___<revision>		1:∞	Revision
________<revisionAction>	string	1:1	Type of revision
________<revisionAuthorName>	string	1:1	Name of the person who wrote the revision
________<institution>	string	1:1	Name of institution from which the revision was written
________<email>	string	1:1	Email of the person who wrote the revision
________<date>	date	1:1	Date of the revision
________<comments>	*Comments*	1:1	Comments about the modifications done in this revision

The type of revision is one of the following possibilities:

Value	Description
creation	Creation of the file. Only one revision of this type is allowed.
modification	Modification of the file.
review	Review by a person. Comments in the revision tag allows to comment the review.
validation	Validated by a person. No modification done on the document.

The history section keeps track of the creator of the drug XML file as well as the people that modified it. The <revisions> contains as many <revision> tag as necessary. Each <revision> tag contains the same structure, as shown in the figure above, and the correct revisionAction should be used according to the action done on the file.

1.4 Head

Used by *Model*.

The head of the file contains general information about the drug, such as its ID, the model it uses, its name, description and so on. It does not embed information required by the computing engines. The head structure is the following:

```
<head>
  <drug>
    <atcs>
      <atc>J01XA01</atc>
    </atcs>
    <activeSubstances>
      <activeSubstance>vancomycin</activeSubstance>
    </activeSubstances>
    <drugName>
      <name lang="en">Vancomycin</name>
      <name lang="fr">Vancomycine</name>
    </drugName>
    <drugDescription>
      <desc lang="fr">Vanc est un un antibio super sympa</desc>
    </drugDescription>
    <tdmStrategy>
      <text lang="fr">Pourquoi et comment le TDm devrait être appliqué</text>
    </tdmStrategy>
  </drug>
  <study>
    <studyName>
      <name lang="en">Population pharmacokinetic parameters of vancomycin in critically␣
↪ill patients</name>
      <name lang="fr">Paramètres pharmacocinétiques de la vancomycine de patients en␣
↪soins intensifs</name>
    </studyName>
    <studyAuthors>Llopis-Salvia, P. and Jiménez-Torres, N. V.</studyAuthors>
    <description>
```

(continues on next page)

(continued from previous page)

```
      <desc lang="en">Study description on the parameters concerning the administration␣
↪of vancomycin</desc>
      <desc lang="fr">Description de l'étude sur les paramètres ayant attrait à l
↪'administration de la vancomycine</desc>
    </description>
    <references>
      <reference type="bibtex">@article{llopis-salvia_population_2006,
        title = {Population pharmacokinetic parameters of vancomycin in critically ill␣
↪patients},
        volume = {31},
        url = {http://onlinelibrary.wiley.com/doi/10.1111/j.1365-2710.2006.00762.x/full},
        number = {5},
        urldate = {2014-10-23},
        journal = {Journal of clinical pharmacy and therapeutics},
        author = {Llopis-Salvia, P. and Jiménez-Torres, N. V.},
        year = {2006},
        pages = {447--454},
        file = {Llopis-Salvia_Vancomycin.pdf:/home/rob/.zotero/zotero/iv5zqg2p.default/
↪zotero/storage/B5WXR3BE/Llopis-Salvia_Vancomycin.pdf:application/pdf}
        }
      </reference>
    </references>
  </study>
  <comments/>
</head>
```

Table 3: head content

Tag name	Format	Occ.	Description
<drug>		1:1	Description of the drug itself
___<atcs>		1:1	List of ATCs
______<atc>	string	1:∞	The drug code, or Anatomical Therapeutic Chemical (ATC)
___<activeSubstances>		1:1	List of active substances
______<activeSubstance>	string	1:∞	An active substance present in the drug
___<drugName>		1:1	The translated drug's names
______<name>	string	1:∞	Name of the drug for the specified language
___<drugDescriptions>		1:1	The translated drug's descriptions
______<desc>	string	1:∞	Description of the drug for the specified language
___<tdmStrategy>		1:1	The translated drug's TDM strategy
______<text>	string	1:∞	The description of the TDM strategy
<study>		1:1	Description of the study used to fill this file
___<studyName>		1:1	The translated drug's study names
______<name>	string	1:∞	Study name of the drug for the specified language
___<studyAuthors>	string	1:1	The name of the study authors
___<description>		1:1	The translated descriptions of the study
______<desc>	string	1:∞	Description of the study for the specified language
___<references>		1:1	References to related publications
______<reference>	string	0:∞	Reference to a specific article or publication
<comments>	*Comments*	1:1	Comments about the drug's header

The active substances are identifiers, and shall come from a dictionary.

The corresponding ATC can be found online using the Anatomical Therapeutic Chemical Classification System. The drug ID must be unique amongst all the drugs. By convention, it starts with *ch.heig-vd.ezechiel.*, followed by the drug name and eventually the domain and/or study names, for all the drugs that are shipped with the default package of Tucuxi.

The drug name is the general name of the drug, eg. *Gentamicin*. The study name finally specifies the study on which this drug is based. Any of those names can be translated by using the *lang* attribute to specify the target language. It is to be noted that the *lang* attribute is mandatory for such translatable strings.

1.5 DrugModel

Used by *Model*.

This third part of a model file embed every information required for any computation.

```
<drugModel>
  <drugId>vancomycin</drugId>
  <drugModelId>ch.heig-vd.vancomycin</drugModelId>
  <domain></domain>
  <covariates></covariates>
  <activeMoieties></activeMoieties>
  <analyteGroups></analyteGroups>
  <formulationAndRoutes></formulationAndRoutes>
  <timeConsiderations></timeConsiderations>
  <comments />
</drugModel>
```

Table 4: drugModel content

Tag name	Format	Occ.	Description
<drugModel>			Everything needed for any calculation
____<drugId>	string	1:1	Unique identifier of the drug
____<drugModelId>	string	1:1	Identifier of the model described in the file
____<domain>	*Domain*	1:1	Usage domain for this model. TB
____<covariates>	*Covariates*	1:1	List of covariates used by the model
____<activeMoieties>	*ActiveMoieties*	1:1	List of active moieties. TB
____<analyteGroups>	*Analyte groups*	1:1	List of groups of analytes
____<formulationAndRoutes>	*Formulation and route*	1:1	List of formulation and routes of administration
____<timeConsiderations>	*Time considerations*	1:1	Some information about time, such as halflife
____<comments>	*Comments*	1:1	General comments on the model

The *drugId* shall uniquely identify the drug. It shall come out of a dictionary.

The *drugModelId* is the identifier of the model proposed in the file. It shall be unique. Some conventions will help maintain such models. It shall consist of alphanumeric characters separated by dots, such as **ch.heig-vd.vancomycin**. The first part is the country identifier, the second the instution identifier, and the third the drug identifier. A fourth one can be added in case various models for the same drug are proposed by the same institution: **ch.heig-vd.vancomycin.model2**.

Except the *drugId*, the *drugModelId*, and the *comments* fields, the other ones are described in specific sections.

The *Domain* represents the validity of the drug model for a specific patient. So, depending on his covariates, the model should not be used.

The *Covariates* list all the covariates used in various parts of the drugModel.

The *ActiveMoieties* list the active moieties of the drug. For a majority of drugs there should be only a single active moiety, and so filling this part should be straightforward.

The *Analyte groups* list the groups of analytes. This concept of groups allow to have independent or dependent analytes in terms of Pk models.

The *Formulation and route* list the formulations and routes of administrations. As such it mainly embeds absorption parameters descriptions.

The *Time considerations* allows to indicate the half life of the drug and the validity time of measures.

1.6 Domain

The domain of application gives information about the population on which the model can be applied.

```
<domain>
    <description>
        <desc lang="en">This is the domain, bla bla bla</desc>
    </description>
    <constraints>
        <constraint>
            <constraintType>hard</constraintType>
            <errorMessage>
                <text lang="en">The age shall be positive</text>
            </errorMessage>
            <requiredCovariates>
                <covariateId>age</covariateId>
            </requiredCovariates>
            <checkOperation>
                <softFormula>
                    <inputs>
                        <input>
                            <id>age</id>
                            <type>int</type>
                        </input>
                    </inputs>
                    <formula><![CDATA[return (age > 0);
                        ]]>
                    </formula>
                </softFormula>
                <comments/>
            </checkOperation>
            <comments/>
        </constraint>
        <constraint>
            <constraintType>soft</constraintType>
            <errorMessage>
                <text lang="en">The weight should not be too much</text>
            </errorMessage>
            <requiredCovariates>
                <covariateId>bodyweight</covariateId>
            </requiredCovariates>
            <checkOperation>
```

(continues on next page)

(continued from previous page)

```
                <softFormula>
                    <inputs>
                        <input>
                            <id>bodyweight</id>
                            <type>double</type>
                        </input>
                    </inputs>
                    <formula><![CDATA[return (return (bodyweight < 100);
                        ]]>
                    </formula>
                </softFormula>
                <comments/>
            </checkOperation>
            <comments/>
        </constraint>
    </constraints>
</domain>
```

Table 5: domain content

Tag name	Format	Occ.	Description
<description>		1:1	The translated descriptions of the domain
___<desc>	string	1:∞	Description of the domain for the specified language
<constraints>	string	1:1	List of constraints allowing the model to be used
___<constraint>	*Constraint*	0:∞	A constraint for using the model
______<constraintType>	*ConstraintType*	1:1	Importance of the constraint: soft or hard
______<errorMessage>		1:1	The translated error message for the constraint
_________<text>	string	1:∞	Error message in the specified language
______<requiredCovariates>		1:1	List of required covariates for checking this constraint
_________<covariateId>	string	1:∞	Id of a covariate required by this constraint
______<checkOperation>	*Operation*	1:1	formula used to check the validity of covariates
______<comments>	*Comments*	1:1	Comments conserning the constraint

The description shall allow to understand the domain of application of the model. It shall be as complete as possible.

The constraints then allow the software to check wether the model can be used for a specific patient or not.

For instance, if the model is valid for adults from 20 to 60 years old, the covariate *ageInYears* of the patient shall be available and checked in order to avoir the use of the model for children.

In case a specific constraint is not met, then the errorMessage is used to notify the user.

1.6.1 Constraint

A constraint allows to check for the possibility to use the model with a specific patient, depending on his covariates validity.

Table 6: constraint content

Tag name	Format	Occ.	Description
<constraint>			A constraint for using the model
____<constraintType>	*ConstraintType*	1:1	Importance of the constraint: soft or hard
____<errorMessage>		1:1	The translated error message for the constraint
________<text>	string	1:∞	Error message in the specified language
____<requiredCovariates>		1:1	List of required covariates for checking this constraint
________<covariateId>	string	1:∞	Id of a covariate required by this constraint
____<checkOperation>	*Operation*	1:1	formula used to check the validity of covariates

A constraint can be *soft* or *hard*. In case of a *soft* one, only a warning is issued, while a *hard* constraint shall impose a rejection of a priori and a posteriori calculations.

The list of required covariates allows to check wether all covariates are available, and the formula is used to check the constraint validity. This formula shall return a boolean returning *true* in case the constraint is met, and *false* otherwise.

1.6.2 ConstraintType

Used by *Constraint*.

Type of constraint

Table 7: constraintType

Value	Description
soft	If the patient covariate can not be retrieved, a warning is issued, but the model can be used
hard	If the patient covariate can not be retrieved, a warning is issued, and the model can not be used

1.7 Covariates

A covariate is a medical information about the patient. A drug can contain from 0 to an unlimited number of covariates that can then be used by Tucuxi to compute and adapt the parameters for the given patient. They are defined in this manner:

```
<covariate>
    <covariateId>bodyweight</covariateId>
    <covariateName>
        <name lang='en'>Total Body Weight</name>
        <name lang='fr'>Poids total</name>
    </covariateName>
    <description>
        <desc lang='en'>Total body weight of patient, in kilogramms</desc>
```

(continues on next page)

(continued from previous page)

```
    <desc lang='fr'>Poids total du patient, en kilogramme</desc>
  </description>
  <unit>kg</unit>
  <covariateType>standard</covariateType>
  <dataType>double</dataType>
  <interpolationType>linear</interpolationType>
  <refreshPeriod>
   <unit>d</unit>
   <value>30</value>
  </refreshPeriod>
  <covariateValue>
    <standardValue>70</standardValue>
  </covariateValue>
  <validation>
    <errorMessage>
      <text lang='fr'>The body weight shall be in the interval [44,100].</text>
    </errorMessage>
    <operation>
      <softFormula>
        <inputs>
          <input>
            <id>bodyweight</id>
            <type>double</type>
          </input>
        </inputs>
        <code>
          <![CDATA[return ((bodyweight >= 44) && (bodyweight <= 110));
          ]]>
        </code>
      </softFormula>
      <comments/>
    </operation>
    <comments/>
  </validation>
  <comments/>
</covariate>
```

Table 8: covariates content

Tag name	Format	Occ.	Description
<covariate>		0:∞	Description of a covariate
____<covariateId>	string	1:1	The covariate's unique identifier
____<covariateName>		1:1	The translated covariate's names
________<name>	string	1:∞	Name of the covariate for the specified language
____<description>		1:1	The translated covariate's descriptions
________<desc>	string	1:∞	Description of the covariate for the specified language
____<unit>	string	1:1	The covariate's unit
____<covariateType>	*covariateType*	1:1	The covariate's type
____<dataType>	*covariateDataType*	1:1	The covariate's type
____<interpolationType>	*interpolationType*	1:1	The covariate's type
____<refreshPeriod>		0:1	The refresh period for the covariate value
________<unit>		1:1	The unit of the refresh period
________<value>		1:1	The duration of the refresh period
____<covariateValue>	*stdAprioriValue*	1:1	The covariate's value, that can be modified by other covariates
____<validation>	ref:*validation*	1:1	Potential validation function for the covariate value
____<comments>	*Comments*	1:1	Comments about the covariate

The interpolation type allow to decide how the calculation interprets covariates changing over time.

The covariate type allows to indicate if the covariate is standard, related to birth, or related to sex.

The covariate data type allows to indicate the type in terms of int, double, bool or date.

The refresh period is used when the interpolation type is not direct. The period defines the time between two modifications of the real covariate value to be used for adjusting other parameters.

A covariate ID must identify the covariate. Some conventions allow to handle generic covariates and are described below. The name and desciption of the covariate can be translated using the *lang* attribute of the <name> and <desc> tags.

The <unit> of the covariate is the unit presented to the user - for example *kg* for the covariate *weight*. The <type>, on the other hand, is the internal type used to store the covariate's value. It can be either an int, a double or a boolean. In case of boolean, then the term is *bool*, and the value shall be 1 for true and 0 for false.

For genetic factors, it is suggested to use *gene_XXX* for the covariate ID, where *XXX* is the gene identification.

Covariate ID	Description
birthbodyweight	weight of the patient at birth
bodyweight	current body weight of the patient
age	current age of the patient, in years
pna	for neonates, post natal age, in days
ga	gestational age, in weeks
clcr	clearance of creatinine
gist	boolean indicating the presence of a gastrointestinal stromal tumor
sex	sex of the patient (0: female, 1: male)

When it is not possible to know the value of a covariate for a specific patient, the default value is used instead. It corresponds to the value of the average individual, also called the typical patient.

1.7.1 covariateType

Used by *Covariates*.

Type of covariate

Table 9: covariateType

Value	Description
standard	A normal covariate
sex	The covariate represents the sex of the patient. Can be automatically retrieved from administrative data, specifically in a patient covariate called *sex*
ageInYears	The age of the patient, in years. Can be automatically retrieved from administrative data, specifically in a patient covariate called *birthdate*
ageInMonths	The age of the patient, in months. Can be automatically retrieved from administrative data, specifically in a patient covariate called *birthdate*
ageInWeeks	The age of the patient, in weeks. Can be automatically retrieved from administrative data, specifically in a patient covariate called *birthdate*
ageInDays	The age of the patient, in days. Can be automatically retrieved from administrative data, specifically in a patient covariate called *birthdate*
dose	The current dose is automatically from the intake list

1.7.2 covariateDataType

Used by *Covariates*.

Table 10: covariateDataType

Value	Description
int	An integer
double	A 64-bit double
bool	A boolean value
date	A date in format YYYY-MM-DDTHH:MM:SS

1.7.3 interpolationType

Used by *Covariates*.

Table 11: interpolationType

Value	Description
direct	As soon as a new covariate value exists, its value is applied
linear	If the covariate has two values at two different times, a linear interpolation is applied between both time points
sigmoid	If the covariate has two values at two different times, a sigmoid interpolation is applied between both time points
tanh	If the covariate has two values at two different times, an hyperbolic tangent interpolation is applied between both time points

1.8 ActiveMoieties

```
<activeMoieties>
  <activeMoiety>
    <activeMoietyId>vancomycin</activeMoietyId>
    <activeMoietyName>
      <name lang="fr">        </name>
    </activeMoietyName>
    <unit>mg/l</unit>
    <analyteIdList>
      <analyteId>vancomycin</analyteId>
    </analyteIdList>
    <formula>...</formula>
    <!-- Drug targets -->
    <targets>
    </targets>


  </activeMoiety> <!--fraction active -->
</activeMoieties>
```

Table 12: activeMoiety content

Tag name	Format	Occ.	Description
<activeMoiety>			An active moiety
____<activeMoietyid>	string	1:1	active moiety unique identifier
____<activeMoietyName>		1:1	The translated active moiety's names
_______<name>	string	1:∞	Name of the active moiety for the specified language
____<unit>	string	1:1	The active moiety's unit
____<analyteIdList>		1:1	The list of analytes influencing this active moiety
_______<analyteId>	string	1:∞	The Id of an analyte required to compute this active moiety
____<analytesToMoietyFormula>	*Operation*	1:1	The formula for calculating the active moiety concentration based on the analytes
____<targets>	*Targets*	1:1	A list of targets
________<target>	*Targets*	0:∞	A target to be reached

An active moiety corresponds to an active substance of the medical drug. A drug model can be composed of more than one active moiety, but this case is not very common. The *activeMoietyId* uniquely identifies the active moiety. In case of a single analyte drug model it is usually convenient to share the same Id as the analyte.

The *activeMoietyName* is the name that will be used when information display is required.

The *unit* is the unit in which the computation results shall be displayed.

In case of a single analyte drug model, the active moiety concentration is the concentration of the analyte. However, in case of multi-analytes drug models, the computation of an active moiety concentration requires a formula exploiting the concentrations of the analytes. Therefore, a list of required analyte Ids, and a formula, are required to convert correctly the analyte. It is to be noted that these are mandatory in any case, but the formula can be an hardcoded *direct* in case of a single analyte. The software will then be able to avoid computation.

Finally, therapeutic targets are related to an active moiety. The list is mandatory, but it could be empty. It is important to notice that if no target is given, the dosage adaptation won't be possible, except if individual targets are supplied for computation.

1.8.1 Targets

The targets of an active moiety are described in the <target> tag. It contains all the data about the targeted residual concentration, peak concentration and mean concentration. The structure is as follows:

```
<target>
  <targetType>residual</targetType>
  <targetValues>
    <unit>mg/l</unit>
    <min>
      <standardValue>10</standardValue>
    </min>
    <max>
      <standardValue>20</standardValue>
      <aprioriComputation>
        <formula type="softcoded"><![CDATA[
```

(continues on next page)

(continued from previous page)

```
          if gist then
          return 20;
          else
          return 30;
          ]]>
        </formula>
        <comments/>
      </aprioriComputation>
    </max>
    <best>
      <standardValue>15</standardValue>
    </best>
    <toxicityAlarm><standardValue>15</standardValue></toxicityAlarm>
    <inefficacyAlarm><standardValue>3</standardValue></inefficacyAlarm>
  </targetValues>
  <comments>
    <comment lang="en">As seen with Aline for the residual concentration</comment>
    <comment lang="fr">Vu avec Aline pour la concentration résiduelle</comment>
  </comments>
</target>
```

Table 13: target content

Tag name	Format	Occ.	Description
<target>			A target
____<targetType>	*TargetType*	1:1	Type of target, from an enumeration
____<targetValues>		1:1	The target values
________<unit>	string	1:1	The target unit
________<min>	*stdAprioriValue*	1:1	Minimum targeted value
________<max>	*stdAprioriValue*	1:1	Maximum targeted value
________<best>	*stdAprioriValue*	1:1	Best targeted value
________<toxicityAlarm>	*stdAprioriValue*	1:1	Threshold over which an alarm shall be triggered
________<inefficacyAlarm>	*stdAprioriValue*	1:1	Threshold under which an alarm shall be triggered
________<mic>		0:1	The MIC value, optional
____________<unit>	string	1:1	The MIC unit
____________<micValue>	*stdAprioriValue*	1:1	The MIC value
____<times>		0:1	Time targets when required by the target type
________<unit>	string	1:1	The time unit
________<min>	*stdAprioriValue*	1:1	Minimum targeted time
________<max>	*stdAprioriValue*	1:1	Maximum targeted time
________<best>	*stdAprioriValue*	1:1	Best targeted time
____<comments>	*Comments*	1:1	Comments about the target

1.8.2 TargetType

A target can be of any of these types:

Table 14: targetType

Value	Description
peak	The target is the peak concentration. Times are to be added in the target to define when the peak should be found
residual	The target is the residual concentration
mean	The target is the mean concentration
auc	The target is the area under curve, for a single intake cycle
auc24	The target is the area under curve on 24h
cumulativeAuc	The target is the cumulative area under curve since the beginning of the treatment
aucOverMic	The target is the area under the concentration curve, but only the portion over the MIC, for a single intake cycle
auc24OverMic	The target is the area under the concentration curve, but only the portion over the MIC, for 24h
timeOverMic	The target is the time spent over the MIC, for a single intake cycle
aucDividedByMic	The target is the area under the concentration curve divided by the MIC, for a single intake cycle
auc24DividedByMic	The target is the area under the concentration curve for 24h divided by the MIC
peakDividedByMic	The target is the value of the peak concentration divided by the MIC
residualDividedByMic	The target is the value of the residual concentration divided by the MIC

1.9 Analyte groups

```xml
<analyteGroups>
  <analyteGroup>
    <groupId>vanco1</groupId>
    <pkModelId>std.linear.2comp</pkModelId>
    <analytes>
      <analyte>
        <analyteId>vanco</analyteId>
        <unit>mg/l</unit>
        <molarMass>
          <value>0</value>
          <unit>g/mol</unit>
        </molarMass>
        <description>
          <desc lang="fr"></desc>
        </description> <!-- peut être vide -->

        <errorModel> <!-- optional -->
          <errorModelType>formula</errorModelType>
```

(continues on next page)

(continued from previous page)

```
          <formula>
            <formula type="softcoded"><![CDATA[
              return eps * sigma0 + eps * exp(sigma1);
              ]]>
            </formula> <!-- probablement une autre formule pour Bayes -->
            <comments/>
          </formula>
          <sigmas>
            <sigma>
              <standardValue>0.239</standardValue>
            </sigma>
            <sigma>
              <standardValue>0.185</standardValue>
            </sigma>
          </sigmas>
          <comments>
            <comment lang="fr">Une fonction d'erreur model définie par la fiche médicament</comment>
          </comments>
        </errorModel>
        <comments/>
      </analyte>
    </analytes>

    <!-- Drug parameters -->
    <dispositionParameters>
      <parameters>
      </parameters>

      <!-- elimination parameters correlations -->
      <correlations />
    </dispositionParameters>



  </analyteGroup>
</analyteGroups>
```

Table 15: analyteGroups content

Tag name	Format	Occ.	Description
<analyteGroups>		0:∞	List of groups of analytes
____<analyteGroup>	string	1:∞	A group a analytes
________<groupId>	string	1:1	A unique Id for the group of analytes
________<pkModelId>	string	1:1	The Id of the Pk Model to be used for computation related to this group
________<analytes>		1:1	The list of analytes of the group
____________<analyte>	*Analyte*	1:∞	An analyte
________<dispositionParameters>		1:1	A set of disposition parameters
____________<parameters>		1:1	List of parameters
________________<parameter>	*Parameter*	1:∞	A disposition parameter
____________<correlations>	*Correlations*	1:1	correlation between disposition parameters

The *groupId* uniquely identifies the group within the drugModel. It is required by the *Formulation and route* in order to indicate on which group the various parameters are connected to.

The drug's Pk model ID must reference an existing pharmacokinetic model. The list of models and their corresponding parameters are presented below.

Todo: Update this list

Pk Model ID	Description
std.linear.1comp.macro	linear elimination, 1 compartment
std.linear.1comp.micro	linear elimination, 1 compartment
std.linear.2comp.macro	linear elimination, 2 compartment
std.linear.2comp.micro	linear elimination, 2 compartment
std.linear.3comp.macro	linear elimination, 3 compartment
std.linear.3comp.micro	linear elimination, 3 compartment

The difference between *macro* and *micro* model consists in the parameters supplied. In case of *macro* the parameters are typically clearance and volume, while in case of *micro* the parameters are the micro constants k.

1.9.1 Analyte

```
<analyte>
  <analyteId>vanco</analyteId>
  <unit>mg/l</unit>
  <molarMass>
    <value>0</value>
    <unit>g/mol</unit>
  </molarMass>
  <description>
    <desc lang="fr"></desc>
  </description> <!-- peut être vide -->

  <errorModel> <!-- optional -->
    <errorModelType>formula</errorModelType>
    <formula>
      <formula type="softcoded"><![CDATA[
        return eps * sigma0 + eps * exp(sigma1);
        ]]>
      </formula> <!-- probablement une autre formule pour Bayes -->
      <comments/>
    </formula>
    <sigmas>
      <sigma>
        <standardValue>0.239</standardValue>
      </sigma>
      <sigma>
        <standardValue>0.185</standardValue>
      </sigma>
    </sigmas>
    <comments>
      <comment lang="fr">Une fonction d'erreur model définie par la fiche médicament</
↪comment>
    </comments>
  </errorModel>
  <comments/>
</analyte>
```

Table 16: analyte content

Tag name	Format	Occ.	Description
<analyte>			An analyte
____<analyteId>	string	1:1	The Id of the analyte
____<unit>	string	1:1	The unit used to do calculation with the analyte
____<molarMass>	string	1:1	The molar mass of the analyte
________<value>		1:1	Value of the molar mass
________<unit>	*Analyte*	1:∞	Unit of the molar mass
____<description>		1:1	The translated descriptions of the analyte
________<desc>	string	1:∞	Description of the analyte
____<errorModel>	*Error model*	1:1	The error model corresponding to this analyte
____<comments>	*Comments*	1:1	Comments about this analyte

1.9.2 Error model

The error model is the representation of the intra-individual error. It is used by the post engines and the percentiles engines to regulate the patient's measures and intra-individual variability.

The structure is the following:

```
<errorModel>
  <errorModelType>formula</errorModelType>
  <formula>
    <formula type="softcoded"><![CDATA[
      return eps * sigma0 + eps * exp(sigma1);
      ]]>
    </formula> <!-- probablement une autre formule pour Bayes -->
    <comments/>
  </formula>
  <sigmas>
    <sigma>
      <standardValue>0.239</standardValue>
    </sigma>
    <sigma>
      <standardValue>0.185</standardValue>
    </sigma>
  </sigmas>
  <comments>
    <comment lang="fr">Une fonction d'erreur model définie par la fiche médicament</
↪comment>
  </comments>
</errorModel>
```

Table 17: errorModel content

Tag name	Format	Occ.	Description
<errorModel>			Some time considerations
____<errorModelType>	*errorModelType*	1:1	Type of error model, an enum
____<errorModelFormula>	*Operation*	0:1	A formula if required by the errorModelType
____<sigmas>		1:1	A list of sigmas
________<sigma>	*stdAprioriValue*	1:∞	A sigma used by the error model
____<comments>	*Comments*	1:1	Comments about the error model

1.9.3 errorModelType

Used by *Error model*.

Table 18: errorModelType

Value	Description
additive	An additive error model. Requires a single sigma
proportional	A proportional error model. Requires a single sigma
exponential	An exponential error model. Requires a single sigma
mixed	A mixed error model. Requires two sigmas
propexp	A proportional error model for a posteriori computation, but exponential for percentiles. Requires a single sigma
softcoded	The error model is defined by the formula following the declaration of the errorModelType

For **additive**, **proportional**, **exponential** or **propexp**, a single sigma is required. For **mixed**, two sigmas are mandatory: The first corresponds to the additive error, and the second to the proportional error.

The **propexp** model is provided because some papers are exploiting an exponential which is in fact interpreted by NONMEM as a proportional variability model. Therefore, the NONMEM cross validation does not work as expected. If the user really wants a pure exponential model, then they should use **exponential**. However if the paper has been published based on NONMEM computations, it is safer to choose **propexp**.

For each error model except the **softcoded** one, the model is implemented in the software. For a **softcoded**, the formula supplied in the file is used instead.

1.10 Formulation and route

```
<formulationAndRoutes default="id0">
  <formulationAndRoute>
    <formulationAndRouteId>id0</formulationAndRouteId>
    <formulation>parenteralSolution</formulation>
    <administrationName>champ libre</administrationName>
    <administrationRoute>i.v.</administrationRoute>
    <routeModelId>bolus</routeModelId>

    <dosages>
    </dosages>

    <absorptionParameters>
      <analyteGroupId>vanco1</analyteGroupId>
      <parameters>
      </parameters>
      <correlations />
    </absorptionParameters>
  </formulationAndRoute>
</formulationAndRoutes>
```

Table 19: formulationAndRoute content

Tag name	Format	Occ.	Description
<formulationAndRoute>			Formulation and route
____<formulationAndRouteId>	string	1:1	Id of the formulation and route
____<formulation>	string	1:1	the formulation. Taken from a dictionary
____<administrationName>	string	1:1	A free field to discriminate vendors
____<administrationRoute>	string	1:1	The route of administration, taken from a dictionary
____<absorptionModel>	*absorptionModel*	1:1	Id of the absorption model
____<dosages>	*Dosages*	1:1	Possible dosages
____<absorptionParameters>		1:1	Sets of absorption parameters
________<parameterSetAnalyteGroup>		0:∞	A set of absorption parameters for an analyte group
____________<analyteGroupId>	string	1:1	Id of the analyte group
____________<absorptionModel>	string	1:1	Id of the absorption model
____________<parameterSet>		1:1	The absorption parameters
________________<parameters>	*Parameter*	1:1	List of parameters
____________________<parameter>	*Parameter*	0:∞	A parameter
________________<correlations>	*Correlations*	1:1	Correlations between absorption parameters or between absorption and disposition parameters
____________________<correlation>	*Correlations*	0:∞	Correlations between absorption parameters or between absorption and disposition parameters

The *formulationAndRouteId* is an Id identifying this formulation and route.

The *formulation* is taken from a dictionary that still has to be defined. Example: *parenteralSolution*.

For now it supports the following values:

- "undefined"
- "parenteralSolution"
- "oralSolution"

The *administrationName* is a free string, and can allow to differentiate between identical formulations offered by different vendors.

The *administrationRoute* is taken from a dictionary that still has to be defined. Example : *i.v.*.

For now it supports the following values:

- "undefined"

- “intramuscular”
- “intravenousBolus”
- “intravenousDrip”
- “nasal”
- “oral”
- “rectal”
- “subcutaneous”
- “sublingual”
- “transdermal”
- “vaginal”

The *absorptionModel* can be either *extra*, *infu* or *bolus*, respectiveley for extravascular, infusion or bolus.

1.10.1 absorptionModel

Table 20: absorptionModel

Value	Description
extra	Extravascular route. Can be used for various administration routes, like in a muscle, oral, anal, …
infusion	Infusion in the central compartment
bolus	Immediate availability of the drug in the blood, like an intravenous bolus.

The *dosages* are the possible dosages, used for proposing dosage adaptation.

Finally, the *absorptionParameters* are the absorption parameters corresponding to the *absorptionModel* selected.

For these absorption parameters, the *analyteGroupId* allows to identify the analytes group related to the parameter set.

1.10.2 Dosages

This section contains all the information about the dosages, such as the default units and values of the doses, intervals and infusions. It also contains the lists of doses, intervals and infusions used by the dosage adaptation module[1]. The dosages structure looks like this:

```
<dosages>
  <analyteConversions>
      <analyteConversion>
          <analyteId>imatinib</analyteId>
          <factor>1</factor>
      </analyteConversion>
  </analyteConversions>
  <availableDoses>
      <unit>mg</unit>
      <default>
          <standardValue>400</standardValue>
```

(continues on next page)

[1] A dosage adaptation module is used by Tucuxi to propose and adapt the dosage of a drug, given its targets and the patient’s data.

(continued from previous page)

```
      </default>
      <rangeValues>
          <from>
              <standardValue>100</standardValue>
          </from>
          <to>
              <standardValue>400</standardValue>
          </to>
          <step>
              <standardValue>100</standardValue>
          </step>
      </rangeValues>
      <fixedValues>
          <value>600</value>
          <value>800</value>
      </fixedValues>
  </availableDoses>
  <intervals>
      <unit>h</unit>
      <default>
          <standardValue>24</standardValue>
      </default>
      <fixedValues>
          <value>12</value>
          <value>24</value>
      </fixedValues>
  </intervals>
  <comments/>
</dosages>
```

Table 21: dosages content

Tag name	Format	Occ.	Description
<dosages>			Available dosages associated with a formulation and route
____<standardTreatment>		0:1	A potential standard treatment
________<isFixedDuration>	bool	1:1	Has the treatment a fixed duration?
________<timeValue>		0:1	The duration of the fixed duration
____________<unit>		1:1	The unit of the duration
____________<value>		1:1	The duration value
____<analyteConversions>		1:1	List of analyte conversions
________<analyteConversion>		1:∞	Conversion from the quantity of drug to the quantity of analyte
____________<analyteId>	string	1:1	The Id of the analyte
____________<factor>	double	1:1	The factor to be multiplied to the drug quantity to obtain the analyte quantity
____<availableDoses>	*AvailableDoses*	1:1	Available doses
____<availableIntervals>	*AvailableIntervals*	1:1	Available intervals
____<availableInfusions>	*AvailableInfusions*	0:1	Available infusion times
____<comments>	*Comments*	1:1	Comments about the dosages

The analyte conversion is important if not all the drug is part of a single analyte. In that case the factor allows to link the quantity of analyte corresponding to a certain quantity of drug.

The available doses, intervals and infusions are used by the dosage adaptation engine to propose a suitable dosage.

1.10.3 AvailableDoses

Table 22: dosages content

Tag name	Format	Occ.	Description
<availableDoses>			Available doses
____<unit>		1:1	Unit of the doses
____<default>	*stdAprioriValue*	1:1	Default dose
____<rangeValues>		0:∞	Available doses represented as a range
________<from>	*stdAprioriValue*	1:1	Starting value of the range
________<to>	*stdAprioriValue*	1:1	Ending value of the range
________<step>	*stdAprioriValue*	1:1	Step to be applied between from and to
____<fixedValues>		0:1	A list of fixed doses
________<value>	double	1:1	A dose value

The idea here is to use *rangeValues* OR *fixedValues*, but not both at the same time. However the software support both at the same time. Using *stdAprioriValue* fields allow the dose range to be very flexible, depending on the patient covariates.

If *rangeValues* is used, then the dosage adaptation engine will try every dose between the boundaries *from* and *to*, using *step*. For instance, if *from=5*, *to=25*, and *step=10*, then the values will be 5, 15, and 25.

1.10.4 AvailableIntervals

Table 23: dosages content

Tag name	Format	Occ.	Description
<availableIntervals>			Available intervals
____<unit>		1:1	Unit of the intervals
____<default>	*stdAprioriValue*	1:1	Default interval
____<rangeValues>		0:∞	Available intervals represented as a range
________<from>	*stdAprioriValue*	1:1	Starting value of the range
________<to>	*stdAprioriValue*	1:1	Ending value of the range
________<step>	*stdAprioriValue*	1:1	Step to be applied between from and to
____<fixedValues>		0:1	A list of fixed intervals
________<value>	double	1:1	An interval value

The idea here is to use *rangeValues* OR *fixedValues*, or both at the same time. Using *stdAprioriValue* fields allow the dose range to be very flexible, depending on the patient covariates.

If *rangeValues* is used, then the dosage adaptation engine will try every interval between the boundaries *from* and *to*, using *step*. For instance, if *from=5*, *to=25*, and *step=10*, then the values will be 5, 15, and 25.

1.10.5 AvailableInfusions

Table 24: dosages content

Tag name	Format	Occ.	Description
<availableInfusions>			Available infusion times
____<unit>		1:1	Unit of the infusion times
____<default>	*stdAprioriValue*	1:1	Default infusion time
____<rangeValues>		0:∞	Available infusion times represented as a range
________<from>	*stdAprioriValue*	1:1	Starting value of the range
________<to>	*stdAprioriValue*	1:1	Ending value of the range
________<step>	*stdAprioriValue*	1:1	Step to be applied between from and to
____<fixedValues>		0:1	A list of fixed infusion times
________<value>	double	1:1	An infusion time value

The idea here is to use *rangeValues* OR *fixedValues*, or both at the same time. Using *stdAprioriValue* fields allow the infusion time range to be very flexible, depending on the patient covariates.

If *rangeValues* is used, then the dosage adaptation engine will try every infusion time between the boundaries *from* and *to*, using *step*. For instance, if *from=5*, *to=25*, and *step=10*, then the values will be 5, 15, and 25.

Notes

1.11 Time considerations

Used by *DrugModel*.

Time considerations can help the software to optimize some computations, and also to get information about the relevance of a measure.

Here is an example of time consideration:

```
<timeConsiderations>
  <!-- Drug half-life -->
  <halfLife>
    <unit>h</unit>
    <value>
      <standardValue>12</standardValue>
    </value>
    <multiplier>10</multiplier>
    <comments/>
  </halfLife>


  <outdatedMeasure>
    <unit>d</unit>
    <value>
      <standardValue>100</standardValue>
    </value>
    <comments/>
```

(continues on next page)

(continued from previous page)

```
  </outdatedMeasure>
</timeConsiderations>
```

The second part of the time considerations consists in the time after which a measure is considered irrelevant, and shall not be used for a posteriori computations.

Table 25: timeConsiderations content

Tag name	Format	Occ.	Description
<timeConsideration>			Some time considerations
____<halfLife>	halfLife	1:1	Half life of the drug
________<unit>		1:1	Time unit of the half life
________<duration>	*stdAprioriValue*	1:1	value of the half life
________<multiplier>		1:1	Number of half lifes to reach steady state
________<comments>	*Comments*	1:1	Comments about the half life
____<outdatedMeasure>	outdatedMeasure	1:1	Indication about the relevance of a measure
________<unit>	string	1:1	Time unit
________<duration>	*stdAprioriValue*	1:1	Time after which a measure shall be considered as irrelevant
________<comments>	*Comments*	1:1	Comments about the outdate measure fields

1.11.1 Half life

The half-life describes the time it takes for the plasma concentration, or the amount of drug in the body, to be reduced by 50%. Therefore, in each succeeding half-life, less drug is eliminated. After one half-life the amount of drug remaining in the body is 50%, after two half-lives 25%, etc. After 4 half-lives the amount of drug (6.25%) is considered to be negligible regarding its therapeutic effects.

The half-life is used to determine the residual concentration of a drug at steady-state. The half-life duration given above is multiplied by the cycle multiplier in order to find out how many cycles need to be completed before reaching the steady-state. It is then possible to compute the residual concentration of the drug at steady-state. In most cases, a multiplier of 10 is sufficient, but it is suggested to have a a bigger multiplier. At the end, the automated tests allow to detect if a multiplier was suffenciently big.

1.12 Parameter

The parameters are used by the models to compute the curves and their value depend on the type of the prediction. If the prediction is made for the typical patient, the model will use the population parameters. If the prediction is made *a priori* for a specific patient, the population parameters will be adapted using patient's covariates. Finally, if the prediction is made *a posteriori*, the *a priori* parameters will be adapted using the patient's measures and a post engine. The parametrs are declared this way:

```
<parameter>
  <id>CL</id>
  <unit>l/h</unit>
  <value>
```

(continues on next page)

(continued from previous page)

```
    <standardValue>3.505</standardValue>
    <aprioriComputation>
      <formula type="softcoded"><![CDATA[
        CCR = covariate_CLcr
        BW = covariate_bodyweight
        theta_1 = 0.034;
        theta_2 = 0.015;

        TVCL = theta_1*CCR+theta_2*BW;
        return TVCL;
        ]]>
      </formula>
      <comments/>
    </aprioriComputation>
  </value>
  <bsv>
    <bsvType>normal</bsvType> <!-- même chose que le modèle d'erreur -->
    <stdDevs>
      <stdDev>0.292</stdDev>
    </stdDevs>
  </bsv>
  <validation>
    <errorMessage><text lang="fr"></text></errorMessage>
    <formula type="softcoded"><![CDATA[
      return bodyweight < 300 and bodyweight > 20;
      ]]>
    </formula>
    <comments/>
  </validation>
  <comments>
    <comment lang="en">Typical clearance calculated for a patients with weight = 75 kg et CCR = 70 ml/min</comment>
  </comments>
</parameter>
```

Table 26: parameter content

Tag name	Format	Occ.	Description
<parameter>			A Pk parameter
____<parameterId>	string	1:1	Id of the parameter
____<unit>	string	1:1	the unit of the parameter value
____<parameterValue>	*stdAprioriValue*	1:1	The parameter value and its optional apriori computation
____<bsv>	string	1:1	Between Subject Variability (BSV)
________<bsvType>	*BsvType*	1:1	Type of BSV
________<stdDevs>		0:1	A list of standard deviations
____________<stdDev>	double	1:∞	A standard deviation
____<validation>		1:1	A potential validation of the parameter value
________<errorMessage>		1:1	A translated list of error messages
____________<text>	string	1:∞	An error message, translated in a specific language
________<formula>	*Operation*	1:1	A formula to check the validity of the parameter
________<comments>	*Comments*	1:1	Comments about the validation
____<comments>	*Comments*	1:1	Comments about the parameter

The default value of the parameter corresponds to the typical patient's value, and is used by the model as the population parameter. The parameters identifier must match one of the parameter IDs used by the model.

Warning: Please check carefully the ID of the parameter with respect to the selected model.

1.12.1 BsvType

Table 27: bsvType

Value	Description
none	The parameter does not have variability. It is fixed
normal	The parameter variability follows a Normal distribution
lognormal	The parameter variability follows a LogNormal distribution

For bsv type, the model is implemented in the software.

1.13 Correlations

With the help of the BSV (Between Subject Variability) of each parameter, the correlations are used to build the correlation and covariance matrices. It is possible to express correlations between two parameters in the following structure:

```
<correlations>
  <correlation>
    <param1>CL</param1>
    <param2>V</param2>
    <value>0.798</value>
    <comments></comments>
  </correlation>
</correlations>
```

Tag name	Format	Occ.	Description
<correlation>		0:∞	Description of the correlation
____<param1>	string	1:1	The first parameter's ID
____<param2>	string	1:1	The second parameter's ID
____<value>	double	1:1	The correlation's value
____<comments>		1:1	Comments about the correlation

The first and second parameters IDs must match those in the parameters section.

2 Drug description file specification

This section describes formal specifications of the Tucuxi Drug Description file (.tdd). Specifically it lists formal specifications of the various fields. Each specification is a unique number, prefixed by DFS (for Drug File Specification).

The drug model importer shall implement these verifications, and shall reference the DFS numbers in the comments.

The drug file editor shall implement these verifications, and shall reference the DFS numbers in the comments.

2.1 Model

The global structure of the XML file is the following:

Tag name	Format	Occ.	Description
<history>	*History*	1:1	History of the file
<head>	*Head*	1:1	General information
<drugModel>	*DrugModel*	1:1	Everything needed for computation

---

`DFS-0001`

A *model* shall have exactly one *history*, one *head* and one *drugModel*.

2.2 History

The history of the file is described inside the <history> tag. A standard history looks like this:

```
<history>
  <revisions>
    <revision>
      <revisionAction>creation</revisionAction>
      <revisionAuthorName>John Doe</revisionAuthorName>
```

(continues on next page)

(continued from previous page)

```
      <institution>Name of his institution</institution>
      <email>john@doe.com</email>
      <date>2014-07-17</date>
      <comments/>
    </revision>
    <revision>
      <revisionAction>modification</revisionAction>
      <revisionAuthorName>Jane Doe</revisionAuthorName>
      <institution>Name of her institution</institution>
      <email>jane@doe.com</email>
      <date>2014-07-30</date>
      <comments/>
    </revision>
  </revisions>
</history>
```

Tag name	Format	Occ.	Description
<revisions>		1:1	List of revisions
___<revision>		1:∞	Revision
_________<revisionAction>	string	1:1	Type of revision
_________<revisionAuthorName>	string	1:1	Name of the person who wrote the revision
_________<institution>	string	1:1	Name of institution from which the revision was written
_________<email>	string	1:1	Email of the person who wrote the revision
_________<date>	date	1:1	Date of the revision
_________<comments>	*Comments*	1:1	Comments about the modifications done in this revision

---

DFS-0002

The type of revision is one of the following possibilities:

- creation
- modification
- review
- validation

DFS-0003

In a *revisions* list, there shall be at least one *revision* with *revisionAction* equal to *creation*.

DFS-0004

A *revision* shall contain exactly one *revisionAction*, one *revisionAuthorName*, one *institution*, one *email*, one *date*, and one *comments*.

DFS-0005

An *email* shall contain a valid email address, with an @ in it.

2.3 Head

The head of the file contains general information about the drug, such as its ID, the model it uses, its name, description and so on. It does not embed information required by the computing engines.

Table 1: head content

Tag name	Format	Occ.	Description
<drug>		1:1	Description of the drug itself
___<atcs>		1:1	List of ATCs
______<atc>	string	1:∞	The drug code, or Anatomical Therapeutic Chemical (ATC)
___<activeSubstances>		1:1	List of active substances
______<activeSubstance>	string	1:∞	An active substance present in the drug
___<drugName>		1:1	The translated drug's names
______<name>	string	1:∞	Name of the drug for the specified language
___<drugDescriptions>		1:1	The translated drug's descriptions
______<desc>	string	1:∞	Description of the drug for the specified language
___<tdmStrategy>		1:1	The translated drug's TDM strategy
______<text>	string	1:∞	TB : The description of the TDM strategy
<study>		1:1	Description of the study used to fill this file
___<studyName>		1:1	The translated drug's study names
______<name>	string	1:∞	Study name of the drug for the specified language
___<studyAuthors>	string	1:1	The name of the study authors
___<description>		1:1	The translated descriptions of the study
______<desc>	string	1:∞	Description of the study for the specified language
___<references>		1:1	References to related publications
______<reference>	string	0:∞	Reference to a specific article or publication
<comments>	*Comments*	1:1	Comments about the drug's header

DFS-0006

A *head* shall contain exactly one *drug*, one *study* and one *comments*.

`DFS-0007`

A *drug* shall contain exactly one *atcs*, one *activeSubstances*, one *drugName*, one *drugDescriptions* and one *tdmStrategy*.

`DFS-0008`

In *drug*, the list *atcs* shall contain at least one *atc*.

`DFS-0009`

In *drug*, an *atc* shall be a non empty string corresponding to a valid ATC format.

`DFS-0010`

In *drug*, the list *activeSubstances* shall contain at least one *activeSubstance*.

`DFS-0011`

In *drug*, an *activeSubstance* shall be a non empty string.

`DFS-0012`

In *drug*, the list *drugName* shall contain at least one *name*.

`DFS-0013`

In *drug*, the list *drugDescriptions* shall contain at least one *desc*.

`DFS-0014`

In *drug*, the list *tdmStrategy* shall contain at least one *text*.

`DFS-0015`

A *study* shall contain exactly one *studyName*, one *studyAuthors*, one *description* and one *references*.

`DFS-0016`

In *study*, an *studyAuthors* shall be a non empty string.

`DFS-0017`

In *study*, an *studyName* shall contain at least one *name*.

`DFS-0018`

A *description* follows the rules of a *translatedDescs*. It shall contain at least one *desc*, and only *desc* tags.

`DFS-0019`

In *study*, a *reference* shall have an attribute *type* that can be any of the following values:

- plain
- bibtex

2.4 Drug model

The drug model embeds all information required for the various computations.

Table 2: drugModel content

Tag name	Format	Occ.	Description
<drugModel>			Everything needed for any calculation
____<drugId>	string	1:1	Unique identifier of the drug
____<drugModelId>	string	1:1	Identifier of the model described in the file
____<domain>	*Domain*	1:1	Usage domain for this model. TB
____<covariates>	*Covariates*	1:1	List of covariates used by the model
____<activeMoieties>	*ActiveMoieties*	1:1	List of active moieties. TB
____<analyteGroups>	*Analyte groups*	1:1	List of groups of analytes
____<formulationAndRoutes>	*Formulation and route*	1:1	List of formulation and routes of administration
____<timeConsiderations>	*Time considerations*	1:1	Some information about time, such as halflife
____<comments>	*Comments*	1:1	General comments on the model

`DFS-0020`

A drug model shall contain exactly one *drugId*, one *drugModelId*, one *domain*, one *covariates*, one *activeMoieties*, one *analyteGroups*, one *formulationAndRoutes*, one *timeConsiderations*, and one *comments*.

`DFS-0021`

A *drugId* shall be a non empty string.

`DFS-0022`

A *drugModelId* shall be a non empty string.

2.5 Domain

Table 3: domain content

Tag name	Format	Occ.	Description
<description>		1:1	The translated descriptions of the domain
___<desc>	string	1:∞	Description of the domain for the specified language
<constraints>	string	1:1	List of constraints allowing the model to be used
___<constraint>	*Constraint*	0:∞	A constraint for using the model
______<constraintType>	*ConstraintType*	1:1	Importance of the constraint: soft or hard
______<errorMessage>		1:1	The translated error message for the constraint
_________<text>	string	1:∞	Error message in the specified language
______<requiredCovariates>		1:1	List of required covariates for checking this constraint
_________<covariateId>	string	1:∞	Id of a covariate required by this constraint
______<checkOperation>	*Operation*	1:1	formula used to check the validity of covariates
______<comments>	*Comments*	1:1	Comments conserning the constraint

`DFS-0023`

A *domain* shall contain exactly one *description* and one *constraints*.

`DFS-0024`

A *constraint* shall contain exactly one *constraintType*, one *errorMessage*, one *requiredCovariates*, one *checkOperation*, and one *comments*.

`DFS-0025`

A *constraintType* is any of the following:

- soft
- hard

`DFS-0026`

An *errorMessage* shall contain at least one *text*.

`DFS-0027`

A *requiredCovariates* shall contain at least one *covariateId*.

`DFS-0028`

A *covariateId* found in a *requiredCovariates* list shall exist as a *covariateId* in a *covariate*.

`DFS-0029`

A *checkOperation* is an *operation*, and shall be validated as an *operation*.

2.6 Operation

An operation is used at many places within the file, with different tags refering to it.

`DFS-0030`

An operation shall contain one of the following:

- *softFormula*
- *hardFormula*
- *multiFormula*

and a *comments*.

`DFS-0031`

A *softFormula* shall contain one *inputs* and one *code*.

`DFS-0032`

An *inputs* shall contain any number of *input*.

`DFS-0033`

An *input* shall contain one *id* and one *type*.

`DFS-0034`

In an *input*, an *id* shall reference an existing *covariateId* present in a *covariate*.

`DFS-0035`

In an *input*, a *type* is any of the following:

- int
- double
- bool

`DFS-0036`

A *hardFormula* contains a single identifier, from a predefined list:

- To be defined

`DFS-0037`

A *multiFormula* contains at least one *softFormula* or *hardFormula*. It can then contain any number of these items, in any order.

2.7 Covariate

A covariate represents an information about the patient that can be useful for computations.

Table 4: covariates content

Tag name	Format	Occ.	Description
<covariate>		0:∞	Description of a covariate
____<covariateId>	string	1:1	The covariate's unique identifier
____<covariateName>		1:1	The translated covariate's names
_______<name>	string	1:∞	Name of the covariate for the specified language
____<description>		1:1	The translated covariate's descriptions
_______<desc>	string	1:∞	Description of the covariate for the specified language
____<unit>	string	1:1	The covariate's unit
____<covariateType>	*covariateType*	1:1	The covariate's type
____<dataType>	*covariateDataType*	1:1	The covariate's type
____<interpolationType>	*interpolationType*	1:1	The covariate's type
____<refreshPeriod>		0:1	The refresh period for the covariate value
________<unit>		1:1	The unit of the refresh period
________<value>		1:1	The duration of the refresh period
____<covariateValue>	*stdAprioriValue*	1:1	The covariate's value, that can be modified by other covariates
____<validation>	ref:*validation*	1:1	Potential validation function for the covariate value
____<comments>	*Comments*	1:1	Comments about the covariate

DFS-0038

A *covariates* shall contain any number of *covariate*, but only *covariate* tags.

DFS-0039

A *covariate* shall contain exactly one *covariateId*, one *covariateName*, one *description*, one *unit*, one *covariateType*, one *dataType*, one *interpolationType*, one *covariateValue*, one *validation* and one *comments*.

DFS-0040

A *covariateId*, in a *covariate*, shall be an non-empty string.

DFS-0041

A *covariateName* follows the rules of a *translatedNames*.

`DFS-0043`

In a *covariate*, a unit is any valid unit, as described in *Unit*.

`DFS-0044`

In a *covariate*, if the *covariateType* is a *sex*, then the unit shall be empty or -.

If the *covariateType* is *ageInYears*, then the unit shall be *y*.

If the *covariateType* is *ageInMonths*, then the unit shall be *mon*.

If the *covariateType* is *ageInWeeks*, then the unit shall be *w*.

If the *covariateType* is *ageInDays*, then the unit shall be *d*.

If the *covariateType* is *dose*, then the unit shall be a valid *Weight unit*.

`DFS-0045`

In a *covariate*, if the *covariateId* is *bodyweight*, then the unit shall be a *Weight unit*.

If the *covariateId* is *temperature*, then the unit shall be a *Temperature unit*.

TODO : Add validations for known covariateIds.

`DFS-0046`

A *dataType* is any of the following:

- int
- double
- bool

`DFS-0047`

A *covariateType* is any of the following:

- *standard*
- *sex*
- *ageInYears*
- *ageInMonths*
- *ageInWeeks*
- *ageInDays*
- *dose*

`DFS-0048`

A *interpolationType* is any of the following:

- *direct*
- *linear*

Two other values are valid, but not yet supported by the software.

DFS-0049

In a *covariate*, if *interpolationType* is not *direct*, then there should be a *refreshPeriod* in *covariate*.

DFS-0050

A *refreshPeriod* shall contain one *unit* and one *value*.

DFS-0051

In a *refreshPeriod*, the unit shall be a valid *Time unit*.

DFS-0052

In a *refreshPeriod*, the *value* shall be a valid strictly positive floating point number.

DFS-0053

A *validation* tag shall contain an *errorMessage*, an *operation*, and a *comments*.

2.8 Active moiety

An active moiety corresponds to an active substance of the medical drug. A drug model can be composed of more than one active moiety, but this case is not very common. The *activeMoietyId* uniquely identifies the active moiety. In case of a single analyte drug model it is usually convenient to share the same Id as the analyte.

Table 5: activeMoiety content

Tag name	Format	Occ.	Description
<activeMoiety>			An active moiety
____<activeMoietyid>	string	1:1	active moiety unique identifier
____<activeMoietyName>		1:1	The translated active moiety's names
________<name>	string	1:∞	Name of the active moiety for the specified language
____<unit>	string	1:1	The active moiety's unit
____<analyteIdList>		1:1	The list of analytes influencing this active moiety
________<analyteId>	string	1:∞	The Id of an analyte required to compute this active moiety
____<analytesToMoietyFormula>	*Operation*	1:1	The formula for calculating the active moiety concentration based on the analytes
____<targets>	*Targets*	1:1	A list of targets
________<target>	*Targets*	0:∞	A target to be reached

DFS-0055

An *activeMoieties* shall contain at least one *activeMoiety*, and only *activeMoiety* tags.

DFS-0056

An *activeMoiety* shall contain one *activeMoietyId*, one *activeMoietyName*, one *unit*, one *analyteIdList*, one *analytesToMoietyFormula*, and one *targets*.

DFS-0057

An *activeMoietyId* shall be a non-empty string.

DFS-0058

An *activeMoietyName* is a *translatedNames*.

DFS-0059

In an *activeMoiety* the *unit* shall be a valid *Concentration unit*.

DFS-0060

An *analyteIdList* shall contain at least one *analyteId*, and only *analyteId* tags.

DFS-0061

In an *analyteIdList*, an *analyteId* shall be a non-empty string.

DFS-0062

An *analytesToMoietyFormula* is an operation.

2.9 Target

The targets of an active moiety are described in the <target> tag. It contains all the data about the targeted residual concentration, peak concentration and mean concentration.

Table 6: target content

Tag name	Format	Occ.	Description
<target>			A target
____<targetType>	*TargetType*	1:1	Type of target, from an enumeration
____<targetValues>		1:1	The target values
________<unit>	string	1:1	The target unit
________<min>	*stdAprioriValue*	1:1	Minimum targeted value
________<max>	*stdAprioriValue*	1:1	Maximum targeted value
________<best>	*stdAprioriValue*	1:1	Best targeted value
________<toxicityAlarm>	*stdAprioriValue*	1:1	Threshold over which an alarm shall be triggered
________<inefficacyAlarm>	*stdAprioriValue*	1:1	Threshold under which an alarm shall be triggered
________<mic>		0:1	The MIC value, optional
____________<unit>	string	1:1	The MIC unit
____________<micValue>	*stdAprioriValue*	1:1	The MIC value
____<times>		0:1	Time targets when required by the target type
________<unit>	string	1:1	The time unit
________<min>	*stdAprioriValue*	1:1	Minimum targeted time
________<max>	*stdAprioriValue*	1:1	Maximum targeted time
________<best>	*stdAprioriValue*	1:1	Best targeted time
____<comments>	*Comments*	1:1	Comments about the target

DFS-0063

A *targets* shall only contain *target* tags.

DFS-0064

A *target* shall contain one *targetType*, one *targetValues*, 0 or 1 *times* (depending on the target type), and one *comments*.

If *targetType* is *peak* or *peakDividedByMic*, *times* is mandatory. Else it should not be present.

DFS-0065

A *targetType* shall be any of the following:

- peak
- residual
- mean
- auc
- auc24

- cumulativeAuc
- aucOverMic
- auc24OverMic
- timeOverMic
- aucDividedByMic
- auc24DividedByMic
- peakDividedByMic
- residualDividedByMic

`DFS-0066`

A *targetValues* shall contain one *unit*, one *max*, one *min*, one *best*, one *toxicityAlarm*, one *inefficacyAlarm*, and 0 or 1 *mic*, depending on the target type.

If *targetType* is *aucOverMic*, *auc24OverMic*, *timeOverMic*, *aucDividedByMic*, *auc24DividedByMic*, *peakDividedByMic*, or *residualDividedByMic* *mic* is mandatory. Else it should not be present.

`DFS-0067`

In a *target*, if the *targetType* is *peak*, *residual*, *mean*, then the *unit* shall be a valid *Concentration unit*.

In a *target*, if the *targetType* is *auc*, *auc24*, *cumulativeAuc*, *aucOverMic*, *auc24OverMic*, then the *unit* shall be a valid *Concentration Time unit*.

In a *target*, if the *targetType* is *timeOverMic*, *aucDividedByMic* or *auc24OverMic*, then the *unit* shall be a valid *Time unit*.

In a *target*, if the *targetType* is *peakDividedByMic* or *residualDividedByMic*, then the *unit* shall be empty or “-“.

`DFS-0068`

A *min* is a *stdAprioriValue*.

`DFS-0069`

A *max* is a *stdAprioriValue*.

`DFS-0070`

A *best* is a *stdAprioriValue*.

`DFS-0071`

A *toxicityAlarm* is a *stdAprioriValue*.

`DFS-0072`

A *inefficacyAlarm* is a *stdAprioriValue*.

`DFS-0153`

In a *targetValues*, the default value of *inefficacyAlarm* shall be smaller than the default value of *toxicityAlarm*.

`DFS-0154`

In a *targetValues*, the default value of *min* shall be less or equal to the default value of *best*, the default value of *best* shall be less or equal to the default value of *max*.

`DFS-0155`

In a *times*, the default value of *min* shall be less or equal to the default value of *best*, the default value of *best* shall be less or equal to the default value of *max*.

`DFS-0073`

A *times* shall contain one *unit*, one *min*, one *max*, and one *best*.

`DFS-0074`

In a *times*, the *unit* shall be a valid *Time unit*.

2.10 Analyte group

Table 7: analyteGroups content

Tag name	Format	Occ.	Description
<analyteGroups>		0:∞	List of groups of analytes
____<analyteGroup>	string	1:∞	A group a analytes
________<groupId>	string	1:1	A unique Id for the group of analytes
________<pkModelId>	string	1:1	The Id of the Pk Model to be used for computation related to this group
________<analytes>		1:1	The list of analytes of the group
____________<analyte>	*Analyte*	1:∞	An analyte
________<dispositionParameters>		1:1	A set of disposition parameters
____________<parameters>		1:1	List of parameters
________________<parameter>	*Parameter*	1:∞	A disposition parameter
____________<correlations>	*Correlations*	1:1	correlation between disposition parameters

DFS-0075

An *analytGroups* shall contain at least one *analyteGroup*, and only *analyteGroup* tags.

DFS-0076

An *analytGroup* shall contain one *groupId*, one *pkModelId*, on *analytes*, and one *dispositionParameters*.

DFS-0077

An *groupId* shall be a non-empty string.

DFS-0078

A *pkModelId* identifies the PK model, and shall be one of the following (this list has to be updated when new models are added to Tucuxi):

- "linear.1comp.macro"
- "linear.1comp.micro"
- "linear.2comp.macro"
- "linear.2comp.micro"
- "linear.3comp.macro"
- "linear.3comp.micro"
- "linear.2comp.erlang1.macro"
- "linear.2comp.erlang1.micro"
- "linear.2comp.erlang2.macro"
- "linear.2comp.erlang2.micro"
- "linear.2comp.erlang3.macro"
- "linear.2comp.erlang3.micro"
- "linear.2comp.erlang4.macro"
- "linear.2comp.erlang4.micro"
- "linear.2comp.erlang5.macro"
- "linear.2comp.erlang5.micro"
- "linear.2comp.erlang6.macro"
- "linear.2comp.erlang6.micro"
- "michaelismenten.enzyme.1comp"

DFS-0079

A *dispositionParameters* is a *parameterSet*.

2.11 Analyte

Table 8: analyte content

Tag name	Format	Occ.	Description
<analyte>			An analyte
____<analyteId>	string	1:1	The Id of the analyte
____<unit>	string	1:1	The unit used to do calculation with the analyte
____<molarMass>	string	1:1	The molar mass of the analyte
________<value>		1:1	Value of the molar mass
________<unit>	*Analyte*	1:∞	Unit of the molar mass
____<description>		1:1	The translated descriptions of the analyte
________<desc>	string	1:∞	Description of the analyte
____<errorModel>	*Error model*	1:1	The error model corresponding to this analyte
____<comments>	*Comments*	1:1	Comments about this analyte

DFS-0080

An *analytes* shall contain at least one *analyte*, and only *analyte* tags.

DFS-0081

An *analyte* shall contain one *analyteId*, one *unit*, one *molarMass*, one *description*, one *errorModel*, and one *comments*.

DFS-0082

In an *analyte*, the *unit* shall be a valid *Concentration unit*.

DFS-0083

A *molarMass* shall contain one *value* and one *unit*.

DFS-0084

In a *molarMass*, the *value* shall be a valid positive floating point number.

DFS-0085

In a *molarMass*, the *unit* shall be a *Molar mass*.

DFS-0086

In an *analyte*, an *analyteId* shall contain a non-empty string. This string shall be a valid *analyteId* defined in an *activeMoiety*.

DFS-0158

All analytes and active moieties have to share the same concentration unit. This constraint may be removed in a later version, but currently it has to be met.

2.12 Error model

Table 9: errorModel content

Tag name	Format	Occ.	Description
<errorModel>			Some time considerations
____<errorModelType>	*errorModelType*	1:1	Type of error model, an enum
____<sigmas>		1:1	A list of sigmas
________<sigma>	*stdAprioriValue*	1:∞	A sigma used by the error model
____<comments>	*Comments*	1:1	Comments about the error model

DFS-0087

An *errorModel* shall contain one *errorModelType*, one *sigmas* and one *comments*.

DFS-0088

An *errorModelType* shall be any of the following:

- additive
- proportional
- exponential
- mixed
- propexp
- none

DFS-0089

In an *errorModel*, if the *errorModelType* is *none*, then *sigmas* shall be empty. If the *errorModelType* is *mixed*, then *sigmas* shall contain two *sigma*. If the *errorModelType* is *additive*, *proportional*, *exponential* or *propexp*, then *sigmas* shall contain one *sigma*.

DFS-0090

A *sigma* shall be a a *stdAprioriValue* with a default positive value.

2.13 Formulation and routes

Table 10: formulationAndRoute content

Tag name	Format	Occ.	Description
<formulationAndRoute>			Formulation and route
____<formulationAndRouteId>	string	1:1	Id of the formulation and route
____<formulation>	string	1:1	the formulation. Taken from a dictionary
____<administrationName>	string	1:1	A free field to discriminate vendors
____<administrationRoute>	string	1:1	The route of administration, taken from a dictionary
____<absorptionModel>	*absorptionModel*	1:1	Id of the absorption model
____<dosages>	*Dosages*	1:1	Possible dosages
____<absorptionParameters>		1:1	Sets of absorption parameters
________<parameterSetAnalyteGroup>		0:∞	A set of absorption parameters for an analyte group
____________<analyteGroupId>	string	1:1	Id of the analyte group
____________<absorptionModel>	string	1:1	Id of the absorption model
____________<parameterSet>	parameterSet	1:1	The absorption parameters

DFS-0091

A *formulationAndRoutes* shall contain at least one *formulationAndRoute*, and only *formulationAndRoute* tags.

DFS-0092

A *formulationAndRoute* shall contain one *formulationAndRouteId*, one *formulation*, one *administrationName*, one *administrationRoute*, one *absorptionModel*, one *dosages*, one *absorptionParameters*.

DFS-0093

A *formulationAndRouteId* shall be a non-empty string.

DFS-0094

A *formulation* shall be any of the following:

- undefined
- parenteralSolution
- oralSolution
- test

DFS-0095

An *administrationName* is any string.

DFS-0096

An *administrationRoute* shall be any of the following:

- undefined
- intramuscular
- intravenousBolus
- intravenousDrip
- nasal
- oral
- rectal
- subcutaneous
- sublingual
- transdermal
- vaginal

DFS-0097

An *absorptionModel* shall be any of the following:

- undefined
- bolus
- extra
- extra.lag
- infusion

DFS-0098

An *absorptionParameters* shall only contain *parameterSetAnalyteGroup* tags.

DFS-0099

A *parameterSetAnalyteGroup* shall contain one *analyteGroupId*, one *absorptionModel*, and one *parameterSet*.

```
DFS-0100
```

In a *parameterSetAnalyteGroup*, the *analyteGroupId* shall contain an Id that corresponds to an existing *groupId* in an *ananyteGroup* of an *activeMoiety*.

```
DFS-0101
```

In a *parameterSetAnalyteGroup*, the *absorptionModel* shall be a valid absorption model Id.

2.14 Dosages

Table 11: dosages content

Tag name	Format	Occ.	Description
<dosages>			Available dosages associated with a formulation and route
____<standardTreatment>		0:1	A potential standard treatment
________<isFixedDuration>	bool	1:1	Has the treatment a fixed duration?
________<timeValue>		0:1	The duration of the fixed duration
____________<unit>		1:1	The unit of the duration
____________<value>		1:1	The duration value
____<analyteConversions>		1:1	List of analyte conversions
________<analyteConversion>		1:∞	Conversion from the quantity of drug to the quantity of analyte
____________<analyteId>	string	1:1	The Id of the analyte
____________<factor>	double	1:1	The factor to be multiplied to the drug quantity to obtain the analyte quantity
____<availableDoses>	*AvailableDoses*	1:1	Available doses
____<availableIntervals>	*AvailableIntervals*	1:1	Available intervals
____<availableInfusions>	*AvailableInfusions*	0:1	Available infusion times
____<comments>	*Comments*	1:1	Comments about the dosages

```
DFS-0102
```

A *dosages* shall have 0 or 1 *standardTreatment*, one *analyteConversions*, one *availableDoses*, one *availableIntervals*, 0 or 1 *availableInfusions* and one *comments*.

If *absorptionModel* is *infusion*, there shall be one *availableInfusions*, else it shall not be present.

```
DFS-0103
```

A *standardTreatment* shall have one *isFixedDuration*.

`DFS-0104`

In a *standardTreatment*, if *isFixedDuration* is *true*, then there shall be a *timeValue*.

`DFS-0105`

A *timeValue* shall contain one *unit* and one *value*.

`DFS-0106`

In a *timeValue*, the *unit* shall be a valid *Time unit*.

`DFS-0107`

In a *timeValue*, the *value* shall be a valid positive floating point number.

`DFS-0108`

An *analyteConversions* shall contain at least one *analyteConversion*, and only *analyteConversion* tags.

`DFS-0109`

An *analyteConversion* shall contain one *analyteId* and one *factor*.

`DFS-0110`

In an *analyteConversion*, the *analyteId* shall be an identifier existing in an *activeMoiety analyteId*.

`DFS-0111`

A *factor* shall be a valid floating point number in the interval [0, 1].

`DFS-0112`

An *availableDoses* is an *availableValues*.

`DFS-0113`

In an *availableDoses*, the *unit* shall be a *Weight unit*.

`DFS-0114`

An *availableIntervals* is an *availableValues*.

`DFS-0115`

In an *availableIntervals*, the *unit* shall be a *Time unit*.

DFS-0116

An *availableInfusions* is an *availableValues*.

DFS-0117

In an *availableInfusions*, the *unit* shall be a *Time unit*.

2.15 AvailableValues

The available values are used for the doses, intervals and infusions. In the following example *availableValues* would be replaced by *availableDoses*, *availableInfusions* or *availableIntervals*.

Table 12: availableValues content

Tag name	Format	Occ.	Description
<availableValues>			Available values
____<unit>		1:1	Unit of the values
____<default>	*stdAprioriValue*	1:1	Default value
____<rangeValues>		0:∞	Available values represented as a range
________<from>	*stdAprioriValue*	1:1	Starting value of the range
________<to>	*stdAprioriValue*	1:1	Ending value of the range
________<step>	*stdAprioriValue*	1:1	Step to be applied between from and to
____<fixedValues>		0:1	A list of fixed values
________<value>	double	1:1	A value

DFS-0118

An availableValues shall contain one *unit*, one *default*, any number of *rangeValues*, and 0 or 1 *fixedValues*.

DFS-0119

In an availableValues, *default* is an *StdAprioriValue*.

DFS-0120

A *rangeValues* shall contain one *from*, one *to*, and one *step*.

DFS-0121

In a *rangeValues*, *from*, *to*, and *step* are *StdAprioriValue*.

`DFS-0122`

In a *rangeValues*, *from* shall have a default value smaller than *to*.

`DFS-0123`

In a *rangeValues*, *step* shall have a strictly positive floating point number as default value.

`DFS-0156`

In a *rangeValues*, for the default values, *step* shall be strictly smaller than *to* - *from*.

`DFS-0124`

A *fixedValues* shall only contain *value* tags.

`DFS-0125`

In a *fixedValues*, a *value* shall contain a valid floating point number.

2.16 Time considerations

Table 13: timeConsiderations content

Tag name	Format	Occ.	Description
<timeConsideration>			Some time considerations
____<halfLife>	halfLife	1:1	Half life of the drug
_______<unit>		1:1	Time unit of the half life
_______<duration>	*stdAprioriValue*	1:1	value of the half life
_______<multiplier>		1:1	Number of half lifes to reach steady state
_______<comments>	*Comments*	1:1	Comments about the half life
____<outdatedMeasure>	outdatedMeasure	1:1	Indication about the relevance of a measure
_______<unit>	string	1:1	Time unit
_______<duration>	*stdAprioriValue*	1:1	Time after which a measure shall be considered as irrelevant
_______<comments>	*Comments*	1:1	Comments about the outdate measure fields

`DFS-0126`

A *timeConsiderations* shall contain one *halfLife* and one *outdatedMeasure*.

`DFS-0127`

A *halfLife* shall contain one *unit*, one *duration*, one *multiplier* and one *comments*.

DFS-0128

In a *halfLife*, the *unit* shall be a valid *Time unit*.

DFS-0129

A *duration* is a *StdAprioriValue*.

DFS-0130

A *duration* default value shall be positive.

DFS-0131

A *multiplier* shall be a positive floating point number.

DFS-0132

An *outdatedMeasure* shall contain one *unit*, one *duration* and one *comments*.

DFS-0133

In an *outdatedMeasure*, the *unit* shall be a *Time unit*.

2.17 Parameters

Table 14: parameterSet content

Tag name	Format	Occ.	Description
<parameterSet>		1:1	The absorption parameters
____<parameters>	*Parameter*	1:1	List of parameters
________<parameter>	*Parameter*	0:∞	A parameter
____<correlations>	*Correlations*	1:1	Correlations between absorption parameters or between absorption and disposition parameters
________<correlation>	*Correlations*	0:∞	Correlations between absorption parameters or between absorption and disposition parameters

DFS-0134

A *parameterSet* shall contain one *parameters* and one *correlations*.

DFS-0135

A *parameters* shall contain only *parameter* tags.

DFS-0136

A *correlations* shall contain only *correlation* tags.

Parameter structure:

Table 15: parameter content

Tag name	Format	Occ.	Description
<parameter>			A Pk parameter
____<parameterId>	string	1:1	Id of the parameter
____<unit>	string	1:1	the unit of the parameter value
____<parameterValue>	*stdAprioriValue*	1:1	The parameter value and its optional apriori computation
____<bsv>	string	1:1	Between Subject Variability (BSV)
________<bsvType>	*BsvType*	1:1	Type of BSV
________<stdDevs>		0:1	A list of standard deviations
____________<stdDev>	double	1:∞	A standard deviation
____<validation>		1:1	A potential validation of the parameter value
________<errorMessage>		1:1	A translated list of error messages
____________<text>	string	1:∞	An error message, translated in a specific language
________<formula>	*Operation*	1:1	A formula to check the validity of the parameter
________<comments>	*Comments*	1:1	Comments about the validation
____<comments>	*Comments*	1:1	Comments about the parameter

DFS-0137

A *parameter* shall contain one *parameterId*, one *unit*, one *parameterValue*, one *bsv*, one *validation*, and one *comments*.

DFS-0138

A *parameterId* shall be a non-empty string.

DFS-0157

A *parameterId* shall be a parameter Id present in the selected Pk model.

DFS-0139

A *parameterValue* is a *StdAprioriValue*.

DFS-0140

A *bsv* shall contain one *bsvType*. If *bsvType* is not *none*, then it shall also contain one *stdDevs*.

DFS-0141

A *bsvType* is any of the following:

- none
- normal
- lognormal
- proportional
- exponential
- additive

DFS-0142

An *stdDevs* shall contain at least one *stdDev* and only contain *stdDev* tags.

DFS-0143

An *stdDev* shall be a valid positive floating point number.

Correlation structure:

Tag name	Format	Occ.	Description
<correlation>		0:∞	Description of the correlation
____<param1>	string	1:1	The first parameter's ID
____<param2>	string	1:1	The second parameter's ID
____<value>	double	1:1	The correlation's value
____<comments>		1:1	Comments about the correlation

DFS-0145

A *correlation* shall contain one *param1*, one *param2*, one *value*, and one *comments*.

DFS-0146

A *param1* shall reference an existing parameter Id of the model used.

DFS-0147

A *param2* shall reference an existing parameter Id of the model used.

DFS-0148

In a *correlation*, *param1* and *param2* shall not be equal.

DFS-0149

In a *correlation*, *value* shall be a valid floating point number in the interval [-1,1].

2.18 StdAprioriValue

The StdAprioriValue is used at various places in the drug file.

`DFS-0150`

An *stdAprioriValue* shall contain one *standardValue*, and 0 or 1 *aprioriComputation*.

`DFS-0151`

A *standardValue* shall be a floating point number.

`DFS-0152`

An *aprioriComputation* is an *Operation*.

2.19 Unit

2.19.1 Weight unit

A valid weight unit is any of the following:

- kg
- g
- mg
- ug

2.19.2 Concentration unit

A valid concentration unit is any of the following:

- g/l
- mg/l
- ug/l
- g/ml
- mg/ml
- ug/ml

2.19.3 Temperature unit

A valid temperature unit is any of the following:

- celsius

2.19.4 Flow rate unit

A valid flow rate unit is any of the following:

- ml/min
- l/min
- ml/h
- l/h

2.19.5 Mole Concentration unit

A valid mole concentration unit is any of the following:

- mol/l
- mmol/l
- umol/l
- mol/ml
- mmol/ml
- umol/ml

2.19.6 Time unit

A valid time unit is any of the following:

- y
- month
- d
- h
- min
- s

2.19.7 Length unit

A valid length unit is any of the following:

- m
- cm
- dm
- mm
- ft
- in

2.19.8 Molar mass

A valid molar mass unit is any of the following:

- g/mol
- ug/mol
- kg/mol
- kg/umol

2.19.9 Concentration Time unit

A valid concentration times time unit is any of the following:

- h*g/l
- h*mg/l
- h*ug/l
- h*g/ml
- h*mg/ml
- h*ug/ml
- g*h/l
- mg*h/l
- ug*h/l
- g*h/ml
- mg*h/ml
- ug*h/ml
- min*g/l
- min*mg/l
- min*ug/l
- min*g/ml
- min*mg/ml

- min*ug/ml
- g*min/l
- mg*min/l
- ug*min/l
- g*min/ml
- mg*min/ml
- ug*min/ml

2.19.10 Empty unit

A valid absence of unit is any of the following:

- empty string
- “-“

3 Covariates

The covariates allow to adjust the prediction thanks to some information on the patient.

Some specific covariates are:

- Sex
- Age
- Dose

3.1 Sex

Sex is represented as a floating point value in the range [0,1]. The rationale is that it allows to have continuous formula taking advantage of the mean value. If we do not know the person's sex, then how would we choose if he/she is supposed to be a man or a woman? Using a floating point allows to set the value to 0.5 if that information is missing. Therefore the prediction will correspond to an individual being an intermediate between a man and a woman.

Sex is considered as a special covariate in the sense that it could be retrieved from administrative data about the patient.

The following code illustrates a sex covariate. The only mandatory fields are the *covariateType* (*sex*), and the *dataType* (*double*).

```
<covariate>
    <covariateId>sex</covariateId>
    <covariateName>
        <name lang="en">Sex</name>
        <name lang="fr">Sexe</name>
    </covariateName>
    <description>
        <desc lang="en">Sex of the patient</desc>
        <desc lang="fr">Sexe du patient</desc>
    </description>
    <unit></unit>
    <covariateType>sex</covariateType>
    <dataType>double</dataType>
    <interpolationType>direct</interpolationType>
    <covariateValue>
        <standardValue>0.5</standardValue>
```

(continues on next page)

(continued from previous page)

```
    </covariateValue>
    <validation>
        <errorMessage>
            <text lang="en">Sex should be in the range [0,1].</text>
        </errorMessage>
        <operation>
            <softFormula>
                <inputs>
                    <input>
                        <id>sex</id>
                        <type>double</type>
                    </input>
                </inputs>
                <code><![CDATA[
                    return ((sex >= 0.0) && (sex <= 1.0));
                    ]]>
                </code>
            </softFormula>
        <comments/>
        </operation>
        <comments/>
    </validation>
    <comments/>
</covariate>
```

3.2 Age

The age of the patient corresponds to a special covariate, because of two factors:

1. As for the sex, it can be retrieved from administrative data, that is the birthdate.
2. In the drug models it can be used in years, months, weeks or days.

In a drug model, when an age is required, it shall be using the special type to specify if it is ageInYears, ageInMonths, ageInWeeks or ageInDays. A default value has to be indicated as well.

In the patient data that will be scanned to get the covariates used for calculation, there are two means of specifiying the age. It can be of the same type as the drug model covariates, or it can come from a birthdate.

In case the patient data contains the birthdate, it should have the id **birthdate**

Tucuxi is then able to calculate the age based on the birthdate, following the granularity defined in the drug model covariate. It will therefore be automatically updated by the software. For instance, in a model of gentamicin for neonates, the ageInDays will be updated every day automatically.

Here is an example of an age covariate in a drug model file:

```
<covariate>
    <covariateId>age</covariateId>
    <covariateName>
        <name lang="en">Age</name>
        <name lang="fr">Age</name>
    </covariateName>
```

(continues on next page)

(continued from previous page)

```
    <description>
        <desc lang="en">Age of the patient, in years</desc>
        <desc lang="fr">Âge du patient, en années</desc>
    </description>
    <unit>y</unit>
    <covariateType>ageInYears</covariateType>
    <dataType>double</dataType>
    <interpolationType>direct</interpolationType>
    <covariateValue>
        <standardValue>50</standardValue>
    </covariateValue>
    <validation>
        <errorMessage><text lang="fr"></text></errorMessage>
        <operation>
            <softFormula>
                <inputs>
                    <input>
                        <id>age</id>
                        <type>double</type>
                    </input>
                        </inputs>
                <code><![CDATA[return ((age >= 20) && (age <= 88));
                    ]]>
                </code>
            </softFormula>
            <comments/>
        </operation>
        <comments/>
    </validation>
    <comments/>
</covariate>


Dose
----

A specific type of covariate allows to let the software automatically handle the current␣
↪dose. The current dose
is then available as any other covariate. The unit shall be a weight, as the software␣
↪has to translate the
intake dose to the dose covariate.
```

4 Drug file validator

The drugfilevalidator application is meant to validate the drug files to ensure their data are correctly set. The drug file editor does some basic checks but is not able to deeply challenge the data. A second step of validation consists in a simple checker within the GUI application. Pressing CTRL+SHIFT+F2 allows to perform some checks on the drug file.

At the current stage the validator is able to check some integrity of the drug model and the a priori parameters computations. It requires two inputs: a drug file and a test file. A test file has the .dvt extension and contains test vectors in a json format.

The command line is the following:

```
Usage:
    ./tucuvalidator [OPTION...]

        -d, --drugfile arg  Drug file
        -t, --testfile arg  Tests to be conducted
        -l, --logfile arg   Log file
        --help          Print help
```

The drug file is a .tdd file containing a drug model. The test file is a .dvt file (described in the next section). The log file will contain the output log.

4.1 DVT files

The content of such a drug file is as follows:

```
{
    "parameterstests": [
        {
            "testId": "1",
            "sampleDate": "2018-01-03T0:0:0",
            "parameters": [
            ],
            "comment": "test 1",
            "covariates": [
            ],
```

(continues on next page)

(continued from previous page)

```
        },
        {
            "testId": "2",
            ...
        },
        ...
    ]
}
```

The testId is used to report errors. The comment is currently not used but serves as a documentation for the test file.

The sample date is in the form YYYY-MM-DDTHH:MM:SS”. This field is important if age is a covariate of the drug file. If this is the case, the test file will have to contain a birthdate variate. The birthdate and the sample date are used by Tucuxi to build the age (in days, weeks or years). The birthdate is in the form YYYY-MM-DD.

The covariates have the following fields:

```
"covariates": [
        {
            "dataType": "date",
            "unit": "-",
            "value": "1948-01-03",
            "id": "birthdate"
        },
        {
            "dataType": "double",
            "unit": "-",
            "value": "0",
            "id": "sex"
        },
        {
            "dataType": "double",
            "unit": "kg",
            "value": "70.0",
            "id": "bodyweight"
        },
]
```

Their id correspond to the Id expected as patient variates . They normally are the same as the covariates in the drug file except for the birthdate. A birthdate is not a drug file covariate, as the drug file embeds an age covariate for calculation.

Not all covariates need to be set. If this is not the case then Tucuxi will perform computation with the default values.

The parameters are simply identified by their Id and associated with the expected value. They have the following fields:

```
"parameters": [
{
    "value": 11.292000000000002,
    "id": "CL"
},
{
    "value": 300.8,
    "id": "V"
}
],
```

The validator checks the parameter value with the one it calculates and issues an error if they do not match. As these values are floating point numbers a delta of 1e-4 is tolerated for the comparison.

4.2 python scripts

A dvt file can be hand written, but this task is quite long and not very efficient. Therefore python scripts are used to generate these files. There is one file per drug model, each of them following the same template.

Here we explain the content of a standard python file following the example of imatimib.

First some imports:

```
import json;
from commonfunctions import newTest, newFullTest, setDefaults,
                            ageInYears, ageInDays, ageInWeeks, getBoolean;
```

The file **commonfunctions.py** contains useful functions used by every drug python file.

Then we define the default covariates values:

```
defCovariates = [
    {
        "id" : "birthdate",
        "dataType" : "date",
        "value" : "1968-01-01",
        "unit" : "-"
    },
    {
        "id" : "sex",
        "dataType" : "double",
        "value" : "0.5",
        "unit" : "-"
    },
    {
        "id" : "bodyweight",
        "dataType" : "double",
        "value" : "70",
        "unit" : "kg"
    },
    {
        "id" : "gist",
        "dataType" : "bool",
        "value" : "0",
        "unit" : "-"
    }
];
```

Here we can notice that the drug file uses an age covariate, and so here we define a birthdate covariate. We have to be careful with this date as it has to be coherent with the sample date. The difference between these two dates has to correspond to the default value of the age covariate of the drug file.!!

We then define the default parameters values:

```
defParameters = [
    {
        "id" : "CL",
        "value" : 15.0474
    },
    {
        "id" : "V",
        "value" : 393.2
    }
    ];
```

They simply correspond to the values found in the drug file.

We then set the defaults, by calling a function:

```
setDefaults(defCovariates, defParameters);
```

Then, we need to declare and implement a function to calculate parameters based on the covariates and the sample date:

```
def calculateParameters(sampleDate, covariates):
    # Get the covariates
    age = ageInYears(covariates[0], sampleDate);
    sex = float(covariates[1]);
    bodyweight = float(covariates[2]);
    gist = getBoolean(covariates[3]);

    # Define some thetas
    theta1 = 14.3;
    theta4 = 5.42;
    theta5 = 1.49;
    theta6 = -5.81;
    theta7 = -0.806;

    # Perform the calculation thanks to python math
    MEANBW = 70;
    FBW = (bodyweight - MEANBW) / MEANBW;

    MEANAG = 50.0;
    FAGE = (age - MEANAG) / MEANAG;

    if (gist):
        PATH = 1;
    else:
        PATH = 0;

    MALE = sex;

    CL = theta1 + theta4 * FBW + theta5 * MALE-theta5 * (1-MALE) + theta6 * FAGE +␣
↪theta7 * PATH - theta7 * (1 - PATH);

    theta2 = 347;
    theta8 = 46.2;
    V = theta2 + theta8 * sex - theta8 * (1 - sex);
```

(continues on next page)

(continued from previous page)

```
    # Create the parameters list
    parameters = [];
    parameters.append(CL);
    parameters.append(V);
    return parameters;
```

This calculation function is the core of the python file. It should implement the equations found in the published paper. The covariates have an order that needs to respect the order of the default covariates previously defined. The same apply for the parameters.

Now we can simply add the tests we want to perform, in an array, using the **newFullTest** function.

This function takes 5 parameters:

1. A test Id
2. A comment
3. A sample date
4. An array of covariate values. The array has the same size as the default covariates array, and shall respect the same order. It is possible to set a covariate value to "-" in order to use the default value.
5. A function for calculating the parameters (here we use the one previously defined)

```
parameterstests = [];
parameterstests.append(newFullTest("1","test 1","2018-01-03T0:0:0",["1948-01-03","-","70.
↪0","0"], calculateParameters));
parameterstests.append(newFullTest("2","test 2","2018-01-03T0:0:0",["1968-01-03","1","50.
↪0","0"], calculateParameters));
parameterstests.append(newFullTest("3","test 3","2018-01-03T0:0:0",["1968-01-03","0","50.
↪0","1"], calculateParameters));
```

Finally, create the final json from the parameterstests and print it on stdout:

```
jsonTest = {"parameterstests": parameterstests};

print(json.dumps(jsonTest,indent=4))
```

5 Indices and tables

- genindex
- modindex
- search