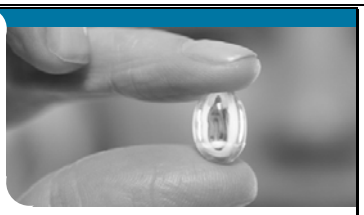
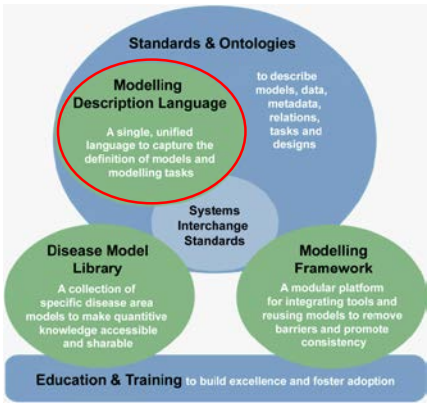
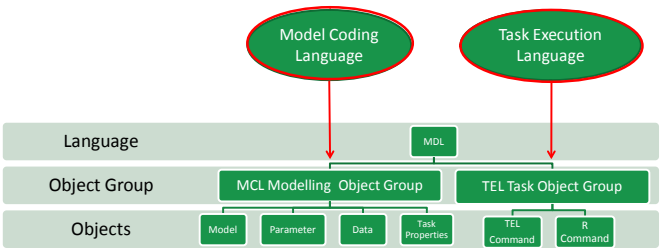
Slide 1	Drug Disease Model Resources  **Modelling Description Language** Nick Holford and Mike K Smith On behalf of the DDMoRe consortium	Presented at PAGE, Glasgow, June 13 2013
Slide 2	**Drug Disease Model Resources** **"Builds and maintains a universally applicable, open source, model based framework, intended as the gold standard for future collaborative drug and disease Modelling & Simulation"** http://www.ddmore.eu/content/project  2	
Slide 3	**MDL Structure**  3	

Slide 4

Evolutionary step from NM-TRAN

Model Coding Language -- Model Object 1

```
warfarin_PK_CONC_mdl = mdlobj{

GROUP_VARIABLES{
GRPCL=POP_CL*(WT/70)^0.75
GRPV=POP_V*WT/70
GRPKA=POP_KA
GRPLG=POP_TLAG
}

INDIVIDUAL_VARIABLES{
CL=GRPCL*exp(eta_PPV_CL)
V=GRPV*exp(eta_PPV_V)
KA=GRPKA*exp(eta_PPV_KA)
ALAG1=GRPLG*exp(eta_PPV_TLAG)
}

...
```

```
$PK


GRPCL=THETA(1)*(WT/70)**0.75
GRPV=THETA(2)*WT/70
GRPKA=THETA(3)
GRPLG=THETA(4)



CL=GRPCL*EXP(ETA(1))
V=GRPV*EXP(ETA(2))
KA=GRPKA*EXP(ETA(3))
ALAG1=GRPLG*EXP(ETA(4))
```

4

Slide 5

Evolutionary step from NM-TRAN

Model Coding Language -- Model Object 2

```
RANDOM_VARIABLE_DEFINITION{
eta_PPV_CL ~ (type=Normal, mean=0,
variance=PPV_CL,level=ID)
eta_PPV_V ~ (type=Normal, mean=0,
variance=PPV_V,level=ID)
eta_PPV_KA ~ (type=Normal, mean=0,
variance=PPV_KA,level=ID)
eta_PPV_TLAG ~ (type=Normal, mean=0,
variance=PPV_TLAG,level=ID)
eps_RUV_PROP ~ (type=Normal, mean=0,
variance=RUV_PROP,level=DV)
eps_RUV_ADD ~ (type=Normal, mean=0,
variance=RUV_ADD,level=DV)
}
```

(Implicit in NM-TRAN/NONMEM)

```
eta_PPV_TLAG ~ (type=Normal, mean=0, variance=PPV_TLAG, level=ID)
eps_RUV_PROP ~ (type=Normal, mean=0, variance=RUV_PROP, level=DV)
```

5

Slide 6

Evolutionary step from NM-TRAN

Model Coding Language -- Model Object 3

```
MODEL_PREDICTION{

LIBRARY {
 F=PK(input=first-order, distribution=1,
 elimination=first-order,
 parameterization=vcl-k,
 param=list(
   cl=CL, v=V,
   DCMT=0, tlag=ALAG1, ka=KA
 )
}
CONC=F.A1/V
}

OBSERVATION{
Y = CONC*(1+eps_RUV_PROP)+eps_RUV_ADD
}

OUTPUT{
ID
TIME
Y
}
```

```
$SUBR ADVAN2 TRANS2




$ERROR
CONC=A(2)/V


Y = CONC*(1+EPS(1))+EPS(2)


$TABLE
ID TIME Y
NOPRINT ONEHEADER FILE=warfpk.fit
```

6

Slide 7

Evolutionary step from NM-TRAN

Model Coding Language -- Data Object

```
warfarin_PK_CONC_dat = dataobj{
FILE{
data=list(source="warfarin_conc_pca.csv",
ignore="#",
inputformat="NONMEM")
}
HEADER{
ID=list(type=categorical)
TIME=list(type=continuous)
WT=list(type=continuous, units="kg")
AGE=list(type=continuous, units="")
SEX=list(type=categorical(female=1,male=0))
AMT=list(type=continuous)
DVID=list(type=categorical)
DV=list(type=continuous)
MDV=list(type=categorical)
}
}
```

```
$DATA warfarin_conc_pca.csv
IGNORE=#
$INPUT ID TIME WT AGE SEX AMT DVID
DV MDV
```

7

Slide 8

Evolutionary step from NM-TRAN

Model Coding Language -- Parameter Object

```
warfarin_PK_CONC_par = parobj{
STRUCTURAL{
POP_CL=list(value=0.1,lo=0.001)
POP_V=list(value=8,lo=0.001)
POP_KA=list(value=2,lo=0.001)
POP_TLAG=list(value=1,lo=0.001)
}

VARIABILITY{
matrix(
PPV_CL=0.1,
0.01, PPV_V=0.1)

diag(
PPV_KA=0.1,
PPV_TLAG=0.1)

RUV_PROP=list(value=0.01 )
RUV_ADD=list(value=0.05, units="mg/L" )
}

}
```

```
$THETA
(0.001,0.1) ; POP_CL L/h/70kg
(0.001,8)   ; POP_V  L/70kg
(0.001,2) ; POP_KA h-1
(0.001,1) ; POP_TLAG h


$OMEGA BLOCK(2)
0.1 ; PPV_CL
0.01 0.1 ; PPV_V

$OMEGA
0.1 ; PPV_KA
0.1 ; PPV_TLAG

$SIGMA
0.01 ; RUV_PROP
0.05 ; RUV_ADD mg/L
```

8

Slide 9

Evolutionary step from NM-TRAN

Model Coding Language – Task Properties Object

```
warfarin_PK_CONC_task = taskobj{


DATA{IGNORE=if(DVID==2)}


myEST=function(t,m,p,d) {

ESTIMATE{
target=t
model=m
parameter=p
data=d
algo=list("COND  INTER")
max=9990
sig=3
cov="y"
}

}
```

```
$DATA
IGNORE (DVID.EQ.2) ; ignore PCA
observations


$EST


METHOD=COND INTER
MAX=9990
SIG=3
$COV
```

9

Slide 10

Task Execution Language

- MCL = **Nouns**, TEL = **Verbs**, MCL Task Properties = **Adverbs**.

  GET <<Model + Data + Parameter initial values>> and

  DO <<Estimation>>

  (**LIKE THIS** <<Task Properties>>)

- Tasks define what MCL ***objects*** are required:
  - Estimation = Model + Data + Parameters (initial, bounds) + Task Properties (Settings)
  - Simulation / Optimal Design = Model + Data Design + Parameters (point estimates or distributions) + Task Properties (Settings)

10

Slide 11

Evolutionary step from NM-TRAN

Model Coding Language – Task Properties Object

```
warfarin_PK_CONC_task = taskobj{

DATA{IGNORE=if(DVID==2)}

myEST=function(t,m,p,d) {

ESTIMATE{
target=t
model=m
parameter=p
data=d
algo=list("COND  INTER")
max=9990
sig=3
cov="y"
}

}
```

```
$DATA
IGNORE (DVID.EQ.2) ; ignore PCA
observations

$EST

METHOD=COND INTER
MAX=9990
SIG=3
$COV
```

11

Slide 12

Evolutionary step from WFN, PsN, etc.

Task Execution Language – TEL Command Object

```
warfarin_PK_CONC_tel = telobj{

# Fit model using NONMEM
warfarin_PK_CONC_fit=myEST(t="NONMEM",
m=warfarin_PK_CONC_mdl,
p=warfarin_PK_CONC_par,
d=warfarin_PK_CONC_dat)

# Update parameter estimates with final
estimates
warfarin_PK_CONC_par=update(warfarin_PK_CONC
_fit,warfarin_PK_CONC_par)

}
```

```
# Windows Command line using Wings
for NONMEM

# Fit model using NONMEM
nmgo warfarin_PK_CONC

# Update parameter estimates with
final estimates
nmctl warfarin_PK_
```

myEST(t="NONMEM",

NONMEM
Monolix
Matlab
BUGS
R

- TEL defines basic tasks that can build to more complex workflows.

12

Slide 13

Use R for general data and statistical tasks

Task Execution Language – R Command Object

Your favourite R script	Your favourite R script

13

Slide 14

Translation to other languages

"Rosetta Stone"

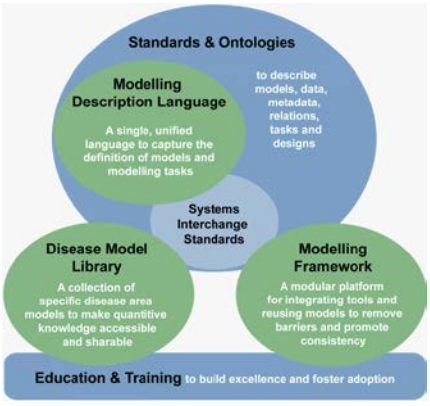
MCL (alternative random effects)	MLXTRAN	BUGS	Phoenix Modeling Language (PML)

GROUP_VARIABLES{
GRPV=POP_V*WT/70
...
}
 	
RANDOM_VARIABLE_DEFINITION
{
lnV ~(type=Normal,
mean=log(GRPV),
variance=PPV_V,
level=ID)
...
}
 	
INDIVIDUAL_VARIABLES{
V=exp(lnV)
...
}
	EQUATION:
	GRPV=POP_V*(WT/70)
	...
	}
	
	DEFINITION:
	
	V =
	{distribution=logNormal,
	prediction=GRPV,
	standardDeviation=PPV_V}
	...
	}
		LOG.GRPV=log(POP_V)
		+ log(WT/70)
		
		for (i in 1:N){
		lnV[i] ~
		dnorm(LOG.GRPV,PPV_V
		)
		...
		
		V[i] = exp(lnV[i])
		...
		}
			grpV=POPV*WT/70
			...
			
			ranef(
			nV = PPVV
			...
			)
			
			stparm(
			V = grpV * exp(nV)
			...
			)

14

Slide 15

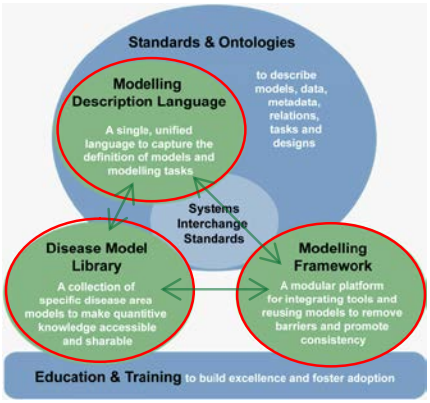
MDL and the rest of DDMoRe



15

Slide 16

MDL and the rest of DDMoRe



16

Slide 17

What else?

Model Coding Language

- Modular models combining **library** functions
- Levels of random effect
- Non-normal distributions
- "odd type data" statements
  - POISSON
  - CATEGORICAL
  - HAZARD

Task Execution Language

- Target software appropriate to task using the same model
- Mix and match using **library** modelling object groups
- Workflow of tasks through **framework**

Active engagement with target software developers

- ICON on future developments in NONMEM
- Pharsight considering using MDL to enhance PML
- Metrum on implementation with BUGS.

Valuable discussion and input from DDMoRe participants

- Subject matter experts contributing to MCL language features & TEL task definitions.

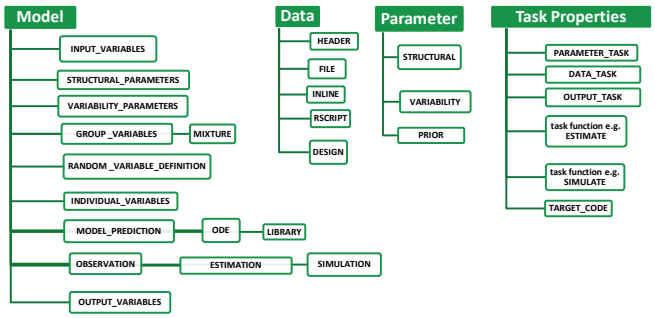
17

Slide 18

18

Slide 19

Modelling Object Group



19

Slide 20

Modelling Object Group (MOG)

- MOG objects
  - Model, parameter, data, task properties
  - Required inputs to TEL task object
  - The "model" is the MOG
  - User may combine objects from Repository MOG with user objects
    - E.g. Repository model+parameter with user data +task properties
- MOG Types
  - Defined & curated, static, { public }

  or
  - User defined, read/write, { private, group, public }

20

Slide 21

Some Ways to Use a MOG

- Full Model (D,P,M,T)
  - Run the model to verify previous results
- Model (M)
  - Library call for model predictions (mixed effect)
  - User supplied D, P, T
- Simulate(M,P)
  - User supplied D and T
- Estimate(D,P,M)
  - User supplied estimation T using library D
- Data Transform(D)
  - User supplied T to transform library D

21

D=data, P=parameter, M=model, T=task_properties

Slide 22

MDL PK Library Function 1

A one compartment model with first-order input and first-order elimination. Dose is administered to compartment zero. Central compartment is 1 even if input is changed to zero-order or bolus.

```
LIBRARY {
 F=PK(input=first-order, distribution=1, elimination=first-order,
 parameterization=vcl-k,
 param=list(
   cl=CL,
   v=V,
   DCMT=0,  # input (depot) compartment is 0
   tlag=ALAG0,
   ka=KA
 )
}
CONC=F.A1/V
```

22

Slide 23

MDL PK Library Function 2

A one compartment model with bolus input and parallel first and mixed-order elimination. The first-order elimination pathway is the formation route for a metabolite. The metabolite disposition is described by two compartments with mixed-order elimination. The metabolite has a delayed effect described by an effect compartment linked to the metabolite compartment.

```
LIBRARY {
F=PK(input=bolus, distribution=1, elimination= parallel-first-mixed-order,
metabolite-formation=first-order, metabolite-distribution=2, metabolite-
elimination=mixed-order, metabolite-link=effect, parameterization=vcl-t,
param=list(v1=10, clfo=1, vmax=3, km=1,  # parent
FCMT=1,  # metabolite is formed from 1st compartment of parent
clpm=clfo,  # Assume all first-order parent elimination leads to metabolite formation
v1m=10, vmaxm=2, kmm=0.1 , v2m=100, cl2m=4,  # metabolite
LCMTm=1m, teqm=1 # effect is determined by linking to metabolite compartment 1
))
}

cem=F.effectm # effect compartment concentration of metabolite

emax=100; c50=1

E=emax*cem/(c50+cem) # effect of metabolite
```

23

Model Coding Language

Rosetta Stone

Nick Holford and Mike K Smith on behalf of the DDMoRe consortium

MDL v5.3 Random Variable Expression	MDL v5.3 Distributions	MLXTRAN	NM-TRAN	BUGS	Phoenix Modeling Language (PML)

#MODEL OBJECT
warfarin_PK_CONC_mdl = mdlobj{
INPUT_VARIABLES{
ID=list(use=id, level=2)
TIME=list(use=idv, units="h")
WT=list(type=continuous, units="kg")
AGE=list(type=continuous, units="")
SEX=list(type=categorical(female,male))
AMT=list(use=amt)
DVID=list(use=dvid,type=categorical)
DV=list(use=dv, level=1)
MDV=list(use=mdv)
}
STRUCTURAL_PARAMETERS{
POP_CL;POP_V;POP_KA;POP_TLAG
}
VARIABILITY_PARAMETERS{
PPV_CL;PPV_V;PPV_KA;PPV_TLAG;RUV_PROP;RUV_ADD
}
GROUP_VARIABLES{
GRPCL=POP_CL*(WT/70)^0.75
GRPV=POP_V*WT/70
GRPKA=POP_KA
GRPLG=POP_TLAG
}
 	
RANDOM_VARIABLE_DEFINITION{
[eta_PPV_CL ~ (type=Normal, mean=0,
variance=PPV_CL,level=ID)
eta_PPV_V ~ (type=Normal, mean=0, variance=PPV_V,level=ID)
eta_PPV_KA ~ (type=Normal, mean=0, variance=PPV_KA,level=ID)
eta_PPV_TLAG ~ (type=Normal, mean=0,
variance=PPV_TLAG,level=ID)
eps_RUV_PROP ~ (type=Normal, mean=0,
variance=RUV_PROP,level=DV)
eps_RUV_ADD ~ (type=Normal, mean=0,
variance=RUV_ADD,level=DV)
}
 	
INDIVIDUAL-VARIABLES{
CL=GRPCL*exp(eta_PPV_CL)
V=GRPV*exp(eta_PPV_V)
KA=GRPKA*exp(eta_PPV_KA)
ALAG1=GRPLG*exp(eta_PPV_TLAG)
}
 	
MODEL_PREDICTION{
LIBRARY{
amount=PK(input=first-order, distribution=1,
elimination=first-order,
parameterization=vcl-t,
param=list(
cl=CL, v=V,
# depot compartment is 0 	
DCMT=0,
tka=ln(2)/KA,
tlag0=ALAG1)
)
}
CONC=amount.A1/V
}
 	
OBSERVATION{
Y = CONC*(1+eps_RUV_PROP)+eps_RUV_ADD
}
	#MODEL OBJECT
	warfarin_PK_CONC_mdl = mdlobj{
	INPUT_VARIABLES{
	ID;TIME;WT;AMT;DV;MDV }
	
	STRUCTURAL_PARAMETERS{
	POP_CL;POP_V;POP_KA;POP_TLAG
	}
	VARIABILITY_PARAMETERS{
	PPV_CL;PPV_V;PPV_KA;PPV_TLAG;RUV_PROP;RUV_ADD
	}
	GROUP_VARIABLES{
	GRPCL=POP_CL*(WT/70)^0.75
	GRPV=POP_V*WT/70
	}
	
	RANDOM_VARIABLE_DEFINITION{
	ln(CL) ~ (type=Normal,
	mean=ln(GRPCL),variance=PPV_CL,level=ID)
	ln(V) ~ (type=Normal, mean=ln(GRPV),
	variance=PPV_V,level=ID)
	ln(KA) ~ (type=Normal, mean=ln(POP_KA),
	variance=PPV_KA,level=ID)
	ln(TLAG) ~ (type=Normal, mean=ln(TLAG),
	variance=PPV_TLAG,level=ID)
	}
	
	INDIVIDUAL-VARIABLES{
	CL=exp(ln(CL))
	V=exp(ln(V))
	KA=exp(ln(KA))
	ALAG1=exp(ln(TLAG))
	}
	
	MODEL PREDICTION{
	LIBRARY{
	amount=list(library=nmadvan,model=2,trans=2,param=list(V,CL,KA,S2,ALAG1,F,A))
	}
	
	CONC=amount.param.A1/V
	}
	
	OBSERVATION{
	ln(Y) ~ (type=Normal, mean=ln(CONC),
	variance=CONC^2 *(RUV_RUVPROP)^2 + RUV_RUVADD^2)
	}
		[INDIVIDUAL]
		input={Tlag_pop,
		omega_Tlag,ka_pop,omega_ka,V_pop,omega_V,Cl_pop,
		omega_Cl,weight,beta_V,beta_Cl}
		
		;GROUP VARIABLES
		EQUATION:
		lw70=log(weight/70)
		
		DEFINITION: ; plus INDIVIDUAL VARIABLES
		Cl = {distribution=lognormal,
		reference=Cl_pop,sd=omega_Tlag,covariate=lw70,coefficient=beta_Cl}
		V = {distribution=lognormal,
		reference=V_pop,sd=omega_Tlag,
		covariate=lw70,coefficient=beta_V}
		ka = {distribution=lognormal,
		reference=ka_pop,sd=omega_ka}
		Tlag = {distribution=lognormal,
		reference=Tlag_pop,sd=omega_Tlag}
		
		[OBSERVATION]
		input={Tlag,ka,V,Cl,a,b}
		
		EQUATION:
		Cc = pkmodel(Tlag, ka, V, Cl)
		
		DEFINITION:
		concentration = {distribution=normal,
		prediction=Cc, errorModel=combined1(a,b)}
			;VARIABLES
			$PK
			
			;GROUP VARIABLES
			GRPCL=THETA(1)*(WT/70)**0.75
			GRPV=THETA(2)*WT/70
			GRPKA=THETA(3)
			GRPLG=THETA(4)
			
			;Implicit definition
			
			;INDIVIDUAL VARIABLES
			CL=GRPCL*EXP(ETA(1))
			V=GRPV*EXP(ETA(2))
			KA=GRPKA*EXP(ETA(3))
			ALAG1=GRPLG*EXP(ETA(4))
			S2=V
			
			;LIBRARY
			$SUBR ADVAN2 TRANS2
			$ERROR
			
			CONC=A(2)/V
			
			; OBSERVATION
			Y=CONC*(1+ERR(1))+ERR(2)
				#GROUP VARIABLES
				for(i in 1:nsub){
				#CL
				logthetaMean[i, 1] <- logCLHat
				+0.75*log(weight[start[i]]/70)
				#V
				logthetaMean[i, 2] <- logV1Hat +
				log(weight[start[i]]/70)
				# ka - alpha1
				logthetaMean[i, 3] <- logDkaHat
				# tlag
				logthetaMean[i, 4] <- logtlagHat
				theta[i,5] <- 1 # F1
				theta[i,6] <- 1 # F2
				theta[i,7] <- 0 # tlag2
				
				# RANDOM (INDIVIDUAL) VARIABLE DEFINITION
				logtheta[i, 1:2] ~ dmnorm(logthetaMean[i, 1:2],
				omega.inv[1:2, 1:2])
				logtheta[i,3] ~ dnorm(logthetaMean[i,3],tau.Ka)
				logtheta[i,4] ~
				dnorm(logthetaMean[i,4],tau.tlag1)
				}logCLHat ~ dnorm(0,1.0E-6)
				logV1Hat ~ dnorm(0,1.0E-6)
				logDkaHat ~ dnorm(0,1.0E-6)
				log(CLHat) <- logCLHat
				log(V1Hat) <- logV1Hat
				log(DkaHat) <- logDkaHat
				tau.Ka <- 1/(sigma.Ka*sigma.Ka)
				sigma.Ka~dunif(0,1000)
				tau.tlag1 <- 1/(sigma.tlag1*sigma.tlag1)
				sigma.tlag1 ~ dunif(0, 1000)
				tau <- 1/(sigma*sigma)
				sigma ~ dunif(0,1000)
				omega.inv[1:2, 1:2] ~
				dwish(omega.inv.prior[1:2, 1:2], 2)
				omega[1:2, 1:2] <- inverse(omega.inv[1:2, 1:2])
				eps <- 1.0E-6
				
				## INDIVIDUAL PREDICTIONS
				for(i in 1:nsub){
				logthetaPred[i, 1:3] ~ dmnorm(logthetaMean[i,
				1:3], omega.inv[1:3, 1:3])
				thetaPred[i,5] <- 1 # F1
				thetaPred[i,6] <- 1 # F2
				thetaPred[i,7] <- 0 # tlag2
				for(j in 1:4){
				log(theta[i,j]) <- logtheta[i,j]
				log(thetaPred[i,j]) <- logthetaPred[i,j]
				}
				
				## Define structural model
				xhat[start[i]:end[i],1:2] <- OneCptModel(
				time[start[i]:end[i]], amt[start[i]:end[i]],
				rate[start[i]:end[i]],
				ii[start[i]:end[i]], evid[start[i]:end[i]],
				cmt[start[i]:end[i]], addl[start[i]:end[i]],
				ss[start[i]:end[i]], theta[i,])
				
				xhatPred[start[i]:end[i],1:2] <- OneCptModel(
				time[start[i]:end[i]], amt[start[i]:end[i]],
				rate[start[i]:end[i]],
				ii[start[i]:end[i]], evid[start[i]:end[i]],
				cmt[start[i]:end[i]], addl[start[i]:end[i]],
				ss[start[i]:end[i]], thetaPred[i,])
				} #end subject specific loop
				
				# OBSERVATION
				for(i in 1:nobs){
				logCobs[i] ~ dnorm(logCHat[i],tau)
				logCobsCond[i] ~ dnorm(logCHat[i],tau)
				CHat[i] <- 1000*xhat[i,2]/theta[subject[i],3]
				logCHat[i] <- log(max(CHat[i],eps))
				
				logCobsPred[i] ~ dnorm(logCHatPred[i],tau)
				CHatPred[i] <-
				1000*xhatPred[i,2]/thetaPred[subject[i],3]
				logCHatPred[i] <- log(max(CHatPred[i],eps))
				}
					#VARIABLES
					
					#GROUP VARIABLES
					covariate(wt)
					grpV=popV*wt/70
					
					#Implicit definition
					
					#INDIVIDUAL VARIABLES
					stparm(
					Cl = popCl*(wt/70)^0.75 * exp(nCl)
					V = grpV * exp(nV)
					Ka = popKa * exp(nKa)
					Tlag = popTlag * exp(nTlag)
					
					#MODEL PREDICTION
					#LIBRARY
					cfMicro (A1, Cl / V, first = (Aa = Ka))
					dosepoint(Aa, tlag = Tlag)
					
					CONC = A1 / V
					
					#OBSERVATION
					observe(CObs = CONC + CEps*(1 + CONC * RUVcv))

#DATA OBJECT
warfarin_PK_CONC_dat = dataobj{
FILE{
data=list(source="warfarin_conc_pca.csv",
ignore="#",
inputformat="NONMEM")
}
HEADER{
ID=list(type=categorical)
TIME=list(type=continuous)
WT=list(type=continuous, units="kg")
AGE=list(type=continuous, units="")
SEX=list(type=categorical(0="female",1="male"))
AMT=list(type=continuous)
DVID=list(type=categorical)
DV=list(type=continuous)
MDV=list(type=categorical)
}
}
	#DATA OBJECT
	warfarin_PK_CONC_dat = dataobj{
	FILE{
	data=list(source="warfarin_conc_pca.csv",
	ignore="#",
	inputformat="NONMEM")
	}
	HEADER{ID;TIME;WT;AGE;SEX;AMT;DVID;DV;MDV}
		[DATA]
		datafile = 'D:/MLXTRAN/warfarin_data.txt'
		header = {id, time, amt, wt, sex, age, dv, dvid}
		
		id = {use = group}
		time = {use = time}
		amt = {use = amount}
		weight = {use = variable, level = id, type = continuous}
		sex = {use = variable, level = id, type = categorical}
		age = {use = ignore}
		dv = {use = observation, name = {'concentration', 'pca'}}
		dvid = {use = observationType, categories = {1, 2}}
			;DATA
			$INPUT ID TIME WT AGE SEX AMT DVID DV MDV
			$DATA warfarin_conc_pca.csv IGNORE=#
			; ignore PCA observations
			IGNORE (DVID.EQ.2)
				xdata <- read.csv("warfarin_conc.csv",
				header=T,ignore="#")
				## create WinBUGS data bugsdata <- list(
				nobs = nobs,
				nsub = length(unique(xdata$subject)),
				start = start,
				end = c(start[-1]-1,nobs),
				subject = xdata$seqID,
				weight = xdata$weight,
				time = xdata$time,
				amt = xdata$dose,
				rate = rep(0,nobs),
				ii = xdata$ii,
				evid = xdata$evid,
				cmt = ifelse(xdata$evid > 0, 1, 2),
				addl = xdata$addl,
				ss = rep(0,nobs),
				logCobs = ifelse(xdata$cobs <=0, NA,
				log(xdata$cobs)),
				omega.inv.prior=diag(rep(0.05,3))
				)
					#DATA
					#in warfarin_conc.csv
					#ID,time,wt,age,sex,amt,rate,dvid,dv,mdv
					0,0,66.7,50,1,100,-2,0,.,1
					0,0.5,66.7,50,1,.,.,1,0,0
					#in Cols.txt (column mapping file):
					id(id)
					time(time)
					covr(wt<-wt)
					covr(age<-age)
					covr(sex<-sex)
					dose(a<-amt)
					obs(cObs<-dv)
					mdv(mdv)

#PARAMETER OBJECT
warfarin_PK_CONC_par = parobj{
STRUCTURAL{
POP_CL=list(value=0.1,lo=0.001)
POP_V=list(value=8,lo=0.001)
POP_KA=list(value=2,lo=0.001)
POP_TLAG=list(value=1,lo=0.001)
}
VARIABILITY{
matrix(type="VAR"){
PPV_CL=0.1,
0.01, PPV_V=0.1}
diag(type="VAR"){
PPV_KA=0.1,
PPV_TLAG=0.1}
RUV_PROP=list(type="VAR",value=0.01)
RUV_ADD=list(type="VAR",value=0.05)
}
}
	#PARAMETER OBJECT
	warfarin_PK_CONC_par = parobj{
	STRUCTURAL{
	POP_CL=0.1
	POP_V=8
	POP_KA=2
	POP_TLAG= =1}
	VARIABILITY{
	matrix(type="VAR"){PPV_CL=0.1,0.01, PPV_V=0.1}
	diag(type="VAR"){PPV_KA=0.1,PPV_TLAG=0.1}
	RUV_PROP=list(type="VAR",value=0.01)
	RUV_ADD=list(type="VAR",value=0.05)
	}
	}
		;PARAMETERS (as part of an estimation task)
		estimatePopulationParameters(
		initialValues={
		pop_Cl = 0.1,
		pop_V = 8,
		pop_ka = 2,
		pop_tlag = 1,
		omega2_Cl = 0.1,
		omega2_V = 0.1,
		omega2_ka = 0.1,
		omega2_tlag = 0.1,
		a_Conc = 0.225,
		b_Conc = 0.1
		}
		)
			;PARAMETERS
			$THETA
			(0.001,0.1) ; POP_CL L/h/70kg
			(0.001,8) ; POP_V L/70kg
			(0.001,2) ; POP_KA h-1
			(0.001,1) ; POP_TLAG h
			$OMEGA BLOCK(2)
			0.1 ; PPV_CL
			0.01 0.1 ; PPV_V
			$OMEGA
			0.1 ; PPV_KA
			0.1 ; PPV_TLAG
			$SIGMA
			0.01 ; RUV_PROP
			0.05 ; RUV_ADD mg/L
				## create initial ##estimates
				bugsinit <- function() {
				list(logCLHat = rnorm(1,log(10),0.2),
				logV1Hat = rnorm(1,log(70),0.2),
				logDkaHat = rnorm(1,log(1),0.2),
				logDtlagHat = rnorm(1,log(1),0.2),
				omega.inv =
				solve(diag(exp(2*rnorm(2,log(0.25),0.5)))),
				sigma.Ka = runif(1,0.1,2),
				sigma.tlaga=runif(1,0.1,2),
				sigma = runif(1,0.1,2)
				)
				}
					#PARAMETERS
					#Stuctural
					fixef(
					popCl = c(0.001, 0.1,)
					popV = c(0.001, 8,)
					popKa = c(0.001, 2,)
					popTlag = c(0.001, 1,)
					RUVcv = c(0, 0.1,)
					)
					#Variability
					ranef(
					block(nV, nCl) = c(0.1,0.01, 0.1)
					nKa = 0.1
					nTlag = 0.1
					)
					error(CEps = 0.1)

#TASK PROPERTIES OBJECT
warfarin_PK_CONC_task = taskobj{
DATA{IGNORE=if(DVID==2)}
myEST=function(t,m,p,d) {
ESTIMATE{
target=t
model=m
parameter=p
data=d
algo=list("COND INTER")
max=9990
sig=3
cov="y"
}
}
	#TASK PROPERTIES OBJECT
	warfarin_PK_CONC_task = taskobj{
	DATA{IGNORE=if(DVID==2)}
	myEST=function(t,m,p,d) {
	ESTIMATE{
	target=t
	model=m
	parameter=p
	data=d
	algo=list("COND INTER")
	max=9990
	sig=3
	cov="y"
	}
	}
		;TASK PROPERTIES
		TASKS:
		globalSettings={
		withVariance=yes,
		
		settingsGraphics="%MLXPROJECT%/warfarin_PK_CONC_project_graphics.xmlx",
		
		settingsAlgorithms="%MLXPROJECT%/warfarin_PK_CONC_project_algorithms.xmlx",
			;TASK PROPERTIES
			$EST METHOD=COND INTER
			MAX=9990 SIG=3 NOABORT
			$COV
				#TASK PROPERTIES
				# within R
				parameters = c("CLHat","V1Hat",
				"DkaHat","omega",
				"sigma","logCobsCond",
				"logCobsPred")
				n.chains = 3
				n.iter = 10000
				n.burnin = 4000
				n.thin = 6
					#TASK PROPERTIES
					rem Windows Command line using PML
					set method=3
					set iterations=200
					set model=warfarin_PK.mdl
					set map=Cols.txt
					set data=warfarin_conc.csv

Acknowledgement: The research leading to these results has received support from the Innovative Medicines Initiative Joint Undertaking under grant agreement n° 115156, resources of which are composed of financial contribution from the European Union's Seventh Framework Programme (FP7/2007-2013) and EFPIA companies' in kind contribution. The DDMoRe project is also financially supported by contributions from Academic and SME partners.