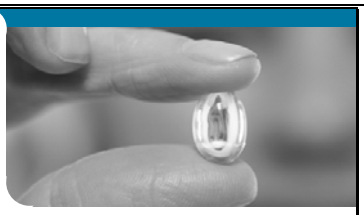
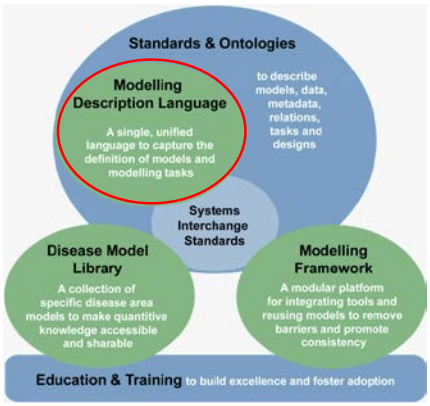
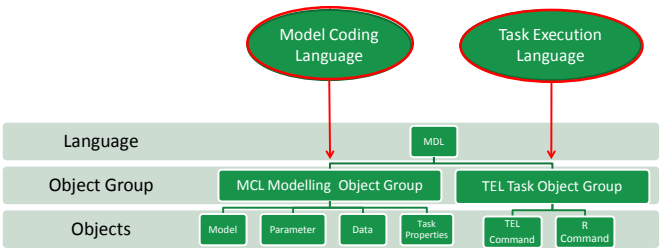
Slide 1	Drug Disease Model Resources  **Modelling Description Language** Nick Holford and Mike K Smith On behalf of the DDMoRe consortium	Presented at PAGE, Glasgow, June 13 2013
Slide 2	**Drug Disease Model Resources** **"Builds and maintains a universally applicable, open source, model based framework, 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"

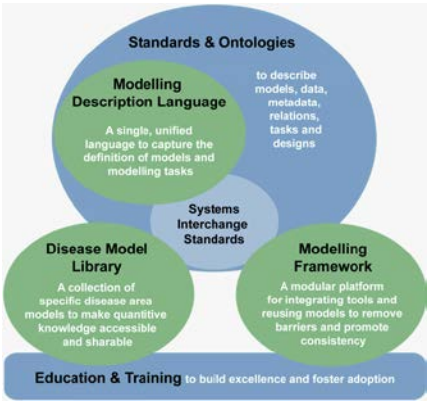
MCL (alternative random effects)	MLXTRAN	BUGS	Phoenix Modeling Language (PML)

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

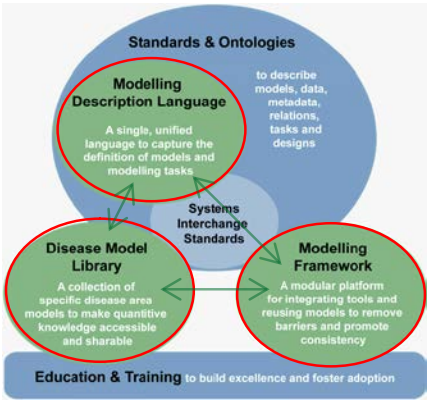
MDL and the rest of DDMoRe



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.

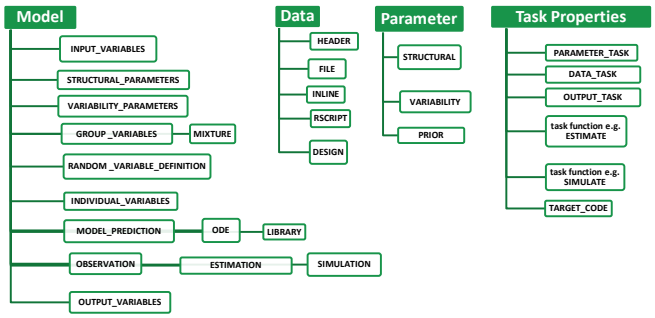
17

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