# On the Noise Modelling and Simulation

Dorel Aiordachioaie   Viorel Nicolau   Mihai Munteanu   Gabriel Sirbu
"Dunarea de Jos" University of Galati, Electrical and Electronics Engineering Faculty
Domneasca-47, Galati – 800008, ROMANIA
Emails: {Dorel.Aiordachioaie, Viorel.Nicolau, Mihai.Munteanu, Gabriel.Sirbu}@gal.ro

## Abstract

Physical modeling is referred to as the first representation of a process model and is represented as a set of differential and algebraic equations. Noise added to the model can improve the estimated behavior of the process and it is more close to reality. The work defines the metamodelling models in order to build tools for the management of the noise, i.e. generation and properly use of it. Some models for random sequences and signals are considered also, under various distribution functions. Thw way of adding noise to dynamic model of the simulated process depends on the structural model of the considered process, i.e. with noise at the output or with noise at the input of the model.

*Keywords:* Process Modeling, Noise Modeling, Metamodelling.

## 1 Introduction

Physical based and object-oriented modeling languages offer an interesting and useful approach in process modeling and simulation, very appreciated and useful in the world of engineers and scientists. Examples of such software-based environments are Omola, Dymola or MathModelica. All these environments are connected with basic features of the Modelica modeling language, as a neutral representation of physical processes. More, based on object-matching features, it is used as the standard representation formalism over the distributed simulation platforms.

Perhaps the first reference that emphasizes a strong call to new modeling principles is of [1], which clearly shows the constraints of pure mathematical models. Other examples could be of [2] and [3], the last one - a project under the resources of Foundation for Strategic Research of Sweden.

By applying the first modeling principles, a set of equations are obtained, which could be organized in two subsets: a subset of differential equations describing the dynamics of the process and a subset of algebraic equations describing the outputs and the constraints of the behavior of the considered process. In the context of real experiments and/or simulations of physical systems, where measurements should be considered as well for the purpose of identification and parameter estimation, the model is improved with noise information.

The reason to introduce noise in process's model is mainly related to un-modeled dynamics and disturbances acting on the process. Considering noise, the state-equations of the model of the process could have the form

$$\mathbf{E} \cdot \dot{\mathbf{x}}(t) + \mathbf{F} \cdot \mathbf{x}(t) = \mathbf{B_u} \cdot \mathbf{u}(t) + \mathbf{B_w} \cdot \mathbf{w}(t) \qquad (1.a)$$

$$\mathbf{y}(t) = \mathbf{C} \cdot \mathbf{x}(t) + \mathbf{e}(t) \qquad (1.b)$$

where a noise component, $\mathbf{w}(t)$, is added for state variables and a noise component **e(t)** is added for the output variables. The type of the noise regarding the power and the probability density function depends on the process. Usually, white noise is considered with variance connected with the dynamics of the process.

Noise modeling is an important task in process modeling. There are unmodelled phenomena and unknown parameters. A noise model should describe how the unmeasured inputs and the unmodeled dynamics could change the behavior of the considered system. Noise modeling also stands for the addition of one or more noise components to state variables, in order to model disturbances and/or some random or unknown behavior. Naturally, the physics of the process should indicate which variables should have noise and which one not, that could be done manually for simple processes. The problem has two aspects: first, there is a complex process, difficult to manage and, second, a software tool that is more efficient and comfortable for any modeler.

Adding noise to all equations can lead to derivates of white noise and – as results – to non-causal process, as infinite values of some variables.

Details on how the non-causality with respect to the input signal, $\mathbf{u}(t)$, can be handled are in [4] and [5]. The problem itself is considered and solved, however [6], where a band limited noise to avoid the problem suggest it.

The present work is looking to noise modeling and appropriate tools to generate different noise models. The reason of the subject is coming from the fact that Modelica, as far we know, does not have any considerations on noise models. The noise modeling tools are considered at the level of the metamodel, i.e. models of the methodology of noise modeling.

The object of the section 2 is related to methods of noise modeling. Section 3 contains the basic theoretical models to generate noise, looking to define the problem, to understand the method and to propose solutions related to the noise modeling. Section 4 is dedicated to a set of noise models, in Modelica language implementation. Simulation results are presented and discussed in section 5.

## 2 Noise metamodelling

Fig. 1 and 2 are looking to present the methodology of noise modeling in the context of physical modeling, i.e. the integration on noise models into physical process models.

The class diagram in Fig. 1 shows the hierarchy of different models, which is used in the building of the system model with physical constraints and – possible – under different representation formalisms. A model is an abstract representation and a generalization of a process model. A process model is an aggregation of one or more models based on equations. An equation-based model is an aggregation of some models, part with noise and part without noise. A noise model should be compliant with the laws of physics and the variables involved in the model should have a physical meaning. Symbolic tools are used to decide where and how to add noise to the model of the process, in order to have a valid representation of the reality.

As Fig. 2 shows, a noise model is generalized as a model. A noise model is an aggregation of some model noises, e.g., of white noise model and band-limited noise, i.e. colored noise. The last one has constraints from a physical model concerning the parameters, e.g. the power of the noise and the frequency bandwidth.

Fig. 3 presents the point of view of the signal domain, which is the output of the noise models. The figure describes a hierarchy on classes, as the object-technology is supposed to have, starting with the class signal as generalization of the class noise. In turn, the class noise is an aggregation of three different noise classes, considering probability distribution function, starting with Wiener distribution and going to white noise and finally to colored noise. Each class has specific methods and attributes, e.g, the power of the white noise and the time constant or, equivalently, the frequency bandwidth, as parameter in the transfer function from white noise to colored noise signals. What is not considered here is the type of the signal, discrete or continuous. This is more complicated, mainly because it requires the interactions with a solver and is out of the paper's horizon.

## 3 Random variables

Random process generation is usually made in two steps: first, generating imitations of independent and identically distributed random variables having the uniform distribution over the interval $(0,1)$ and, second, applying transformations to these variables in order to generate random vectors with arbitrary distributions. These two steps are independent.

The next subsections sketch the theoretical background in order to sustain the declarative models for the generation of various types of the noise. Methods for generating a *sequence* of random numbers have been extensively studied and are well understood. Widely accepted is the method of the *linear congruential* generators. These numbers have the general form

$$U(k+1) = \left(a \cdot U(k) + c\right) \bmod m \qquad (2)$$

where $U(k)$ is the $k$-th element of the sequence and $U(0), a, c$ and $m$ are parameters.

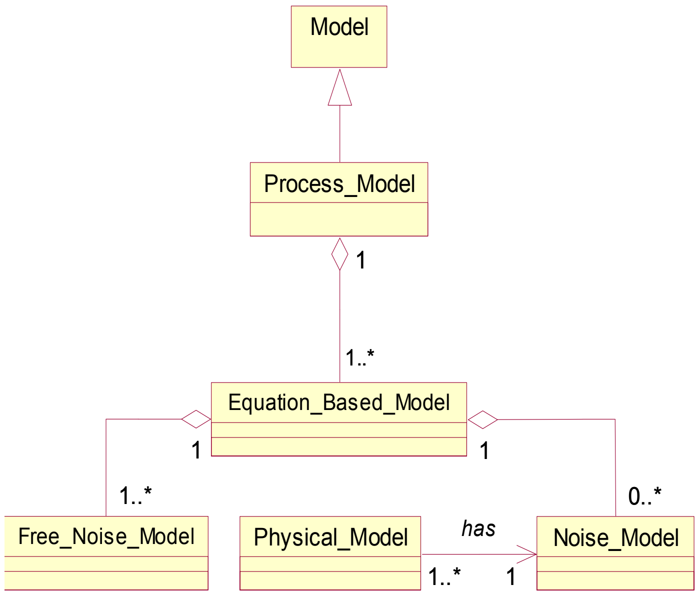

Figure 1: Different types of models

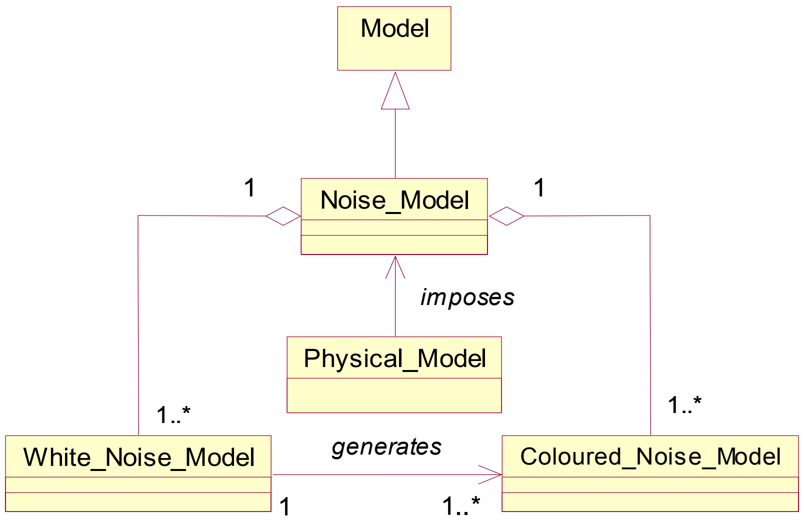

Figure 2: Connections among different types of noise models

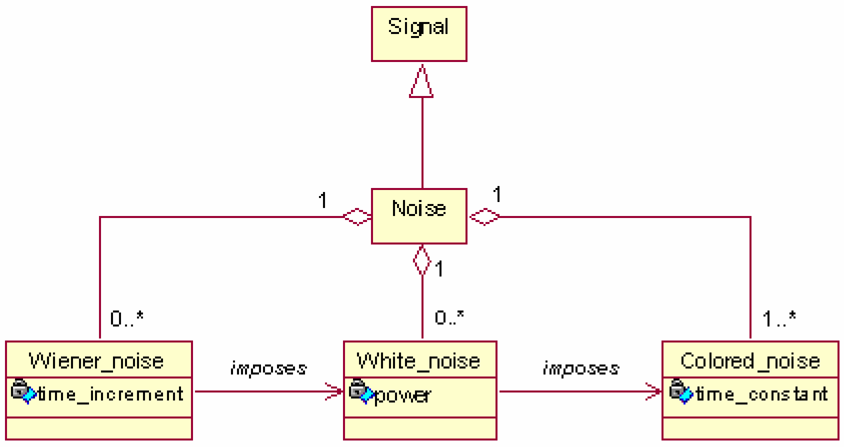

Figure 3: Class diagram of the noise signals

The second variable of the modulus function, *m*, is a large integer which must be prime; the multiplier *a* is an integer in the range (2, 3, …, *m*-1); the additive constant *c* is an integer, often equal to zero. These are chosen to make the sequence look as random as possible. This generator has period *m*-1. Common choices for these values are $U(0) =$ any positive integer, $a = 16807$, $m = 2^{31} - 1$, $c = 0$. Other common choices and optimum algorithms are presented and described, e.g., in [13],[14],[15], [16] and [17].

A Box-Muller method [18] could be used in order to obtain a unit normal random, say *X*. Two uniform random variables $U_1$ and $U_2$ are necessary and the relation

$$X(k) = \sqrt{-2 \cdot \log(U_1(k))} \cdot \sin(2 \cdot \pi \cdot U_2(k) \quad (3)$$

Given a uniform random variable *U(k)*, a Rayleigh random variable *R(k)* can be obtained by:

$$R(k) = \sqrt{2 \cdot \sigma^2 \cdot \ln\left(1/1 - U(k)\right)} \quad (4)$$

where $\sigma^2$ is the variance of the Rayleigh random variable.

## 4 Noise models

Considering an interval $I = [0, T]$, a standard Wiener process is a random variable *W(t)* which satisfies the properties:

1). $W(0) = 0$

2). $W(t_2) - W(t_1) \cong \sqrt{t_2 - t_1} \cdot N(0,1)$, with $t_2 \geq t_1$, $\forall t_1, t_2 \in I$. $N(0,1)$ is a random number under normal distribution with zero mean and unit variance.

3). For $t_1 \leq t_2 \leq t_3 \leq t_4 \in I$, the samples $W(t_2 - t_1)$ and $W(t_4 - t_3)$ are independent.

White noise $N(t)$, with unit variance, is the formal derivative of a Wiener process $W(t)$, as

$$N(t) = \frac{dW(t)}{dt} \quad (5)$$

It may be possible that the noise in a physical system has correlations that are not satisfied by white noise. A colored noise, $\xi(t)$, may be calculated from a stochastic equation as

$$\frac{d\xi(t)}{dt} = -\frac{1}{\tau} \cdot \xi(t) + \frac{\sigma}{\tau} \cdot N(t) \quad (6)$$

where $N(t)$ is a Gaussian white noise, with unit variance and zero mean.

## 5 Modelica implementation

Based on mathematical considerations of above section and on metamodel of Fig. 3, declarative models based on Modelica modeling language will be presented.

Real type numbers makes the interface communication over different models. The Modelica code for interfaces can by as

```
connector PortNumber
    Real n;
end PortNumber;
```

The uniform random number generator needs a function *mod* defined as

```
function mod
    input Real x, y;
    output Real z;
algorithm
    z := x - div(x, y) * y;
end mod;
```

where the assignation in the algorithm section imposes the causality of input and output variables.

A separate model describes global simulation parameters

```
model parameters
// the numbers of random values:
    parameter Integer n=100;
// sample period:
    parameter Real dt = 1;
// start time moment:
    parameter Real start = 1;
// the index of arrays used in simulation:
    Integer j;
algorithm
    j := integer(time/dt) + 1;
end parameters;
```

The uniform random generator over the interval $(0,1]$ is described by

```
model UNG  //uniform_number_generator
    extends parameters;
    constant Integer m =  2^31 - 1;
    constant Integer a= 7^5;
    constant Integer c=10;
    Real xmax,  x[n];
    Integer j;
    PortNumber OUT;
algorithm
    x[1] := 1.0;
    for k in 1:n - 1 loop
        x[k + 1] := mod(a*x[k] + c, m);
    end for;
    xmax := max(x);
    for k in 1:n loop  // normalization
        x[k] := x[k] / xmax;
    end for;
    OUT.n := x[j];
end UNG;
```

The model defines the basic bricks of the noise modeling and generations tools and has two outputs, related to two consecutive random numbers over a set of *n* preimposed values. All variables are part of the generator and the causalities are assigned in the algorithm section.
The normal distribution needs two uniform generators that could be developed by using two independent uniform generators or only one generator but with multiple independent outputs. With the last assumption the model of the normal random generator is as

```
model NNG  // normal_number_generator
    extends UNG_MULTI;
    PortNumber OUT;
    Real x[n];
algorithm
    x[j] := OUT.n;
OUT.n:=sqrt(-2*log(OUT1.n))*sin(6.28*OUT2.n);
end NNG;
```

The model for a Rayleigh distribution is

```
model RRG // Rayleigh random generator
    extends parameters;
    parameter Real sigma=1;
    Real xr;
    PortNumber OUT;
    UNG ung;
algorithm
xr:=sqrt(2*sigma^2*log(1/(1e-5+1-ung.OUT.n)));
    OUT.n := xr;
end RRG;
```

The normal white noise model needs a normal random numbers. The model is described by

```
model white_noise
    extends parameters;
    parameter Real sigma = 1;
    Real xw[n], xa[n];
    PortNumber OUT;
    NNG nng;
algorithm
    xw[j] := sqrt(dt)* nng.OUT.n;
    xa[j] := sigma*(xw[j] - xw[j-1]) / dt;
    OUT.n := xa[j];
end white_noise;
```

There are two random variables, one is Wiener and another one is of white type.
The colored noise is described by the sequence

```
model colored_noise
    extends parameters;
    PortNumber IN, OUT;
    parameter Real tau = 1;
    parameter Real sigma = 1;
    Real xc;
algorithm
    OUT.n := xc;
equation
    when sample(start, dt) then
        der(xc) = -xc/tau + sigma*IN.n/tau;
    end when;
end colored_noise;
```

and has two parameters with the names *tau* and *sigma*. The function **sample**(start,interval) returns true and triggers time events at time instants (*start + i*interval)*.

The final simulation model, in order to generate a colored noise with the imposed parameters, is

```
model sim
    colored_noise cn;
    white_noise wn;
equation
    connect(wn.OUT,  cn.IN);
end sim;
```

In addition, the links between the used models are graphically presented in Fig. 4.

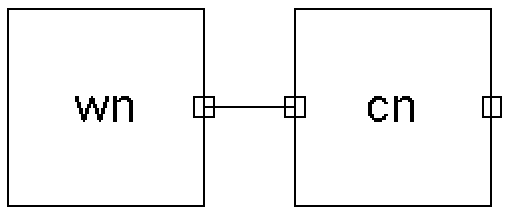

Figure 4: The model for colored white noise

## 6 Simulation results

For discrete sequences, simulations where conducted for different lengths. Figure 5 presents the result for $m = 2^{31} - 1, a = 7^5, c = 10, x(0) = 1$ as parameters of the uniform number generator. The parameter of the obtained random sequence are 0.4277 for mean and 0.0833 the variance. Rising the length to $n$=100 the distribution's parameters are improved to 0.4350 for the mean and 0.0995 for the variance. It is remembered that the ideal values are 0.5 for mean and 1/12.

Figure 6 shows a normal random sequence. The parameters for the second uniform number generator were $m = 2^{31} - 1, a = 7^3, c = 0, x(0) = 2$. For a length of 50 a sequence of 0.0939 and 1.0741 is obtained, as values for mean and variance. With $n$=100 the parameters are -0.0191 and 0.9885.

Fig. 7 shows samples of the simulation results from the outputs of the continous noise models, i.e. white and coloured. It seems that the results are satisfactory over the behavior of the signals.

More statistic tests will be developed in order to improve the structure of generators and to check the distances between the real and the imposed behavior.

## 7 Conclusions

The objective of the work was to define noise models, with different density distribution functions, at the level of metamodels and to implement these in a neutral declarative modeling language, here Modelica.

The study is on the beginning and - as a first trial - the obtained models are compliant with the reference behavior. In the future more study will be done in order to make distinction between continuous time and discrete time random processes and to built a set of models ready to use in noise modelling.

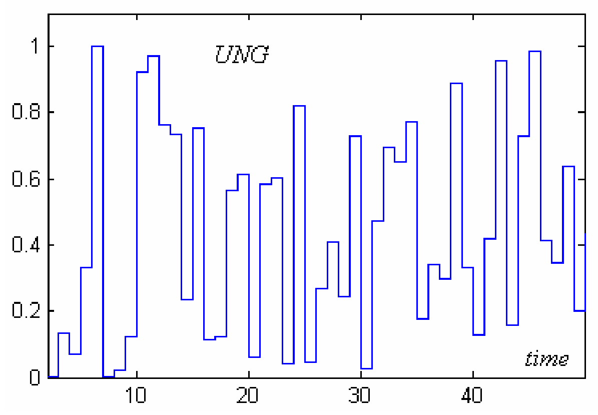

Figure 5: Uniform random number sequence

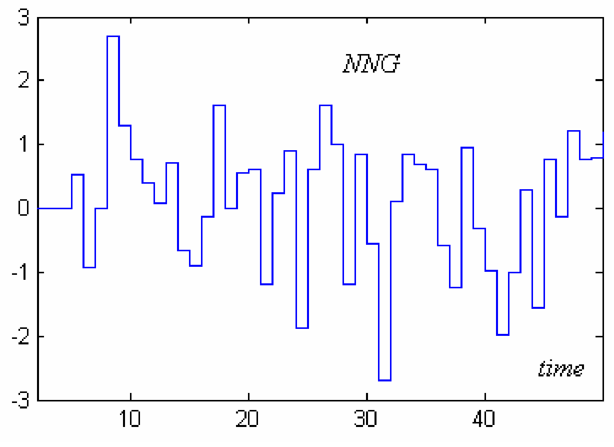

Figure 6: Normal random sequence

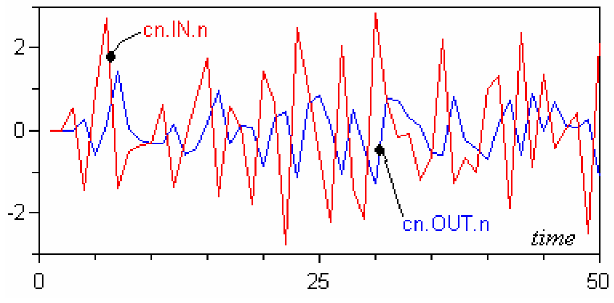

Figure 7: White and colored noise signal
$(dt = 1, \sigma^2 = 1, \tau = 2)$

## Acknowledgements

The authors would like to thank the Romanian National University Research Council for supporting part of the work, under the research grant 1350.

## References

[1] Cellier, F.E., *Continuous System Modeling*, Springer Verlag, New York, 1991.

[2] Åström, K.A., Hilding Elmqvist, Sven Erik Mattsson, *Evolution Of Continuous-Time Modeling And Simulation*, 1998.

[3] VISIMOD - A project funded by the Swedish Foundation for Strategic Research, http://www.visimod.org/.

[4] Schein O., and G. Denk, Numerical solution of stochastic differential-algebraic equations with applications to transient noise simulation of microelectronic circuits, *Journal of Computational and Applied Mathematics*, 100, 1998, pp. 77-92.

[5] Schon, T., Markus Gerdin, Torkel Glad and Fredrik Gustafsson, A Modeling and Filtering Framework for Linear Differential-Algebraic Equations, *In Proc.of the 42nd Conf. on Decision and Control*, Maui, Hawaii, USA, December 2003.

[6] Campbell, S.L., Descriptor systems in the 90's. *Proceedings of the 29th Conf. on Decision and Control*, Honolulu, Hawaii, USA, Dec., 1990.

[7] Tiller, M., *Introduction to Physical Modeling with Modelica*, Kluwer Academic Publisher, 2001.

[8] Fritzson, P., *Principles of Object-Oriented Modeling and Simulation with Modelica* 2.1, Wiley-IEEE Press, New York, 2004.

[9] Modelica and Modelica Association, *Modelica Rationale,* http://www.modelica.org/, 2004.

[10] Elmqvist, H., S.E.Mattsson and M. Otter, Modelica-A Language for Physical System Modeling, Visualization and Interaction, *Proc. of the 1999 IEEE Symp. on Computer-Aided Control System Design*, Hawaii, Aug., 1999.

[11] The Global CAPE-OPEN Consortium, http://www.global-cape-open.org, 2000.

[12] Braunschweig, B.L., Pantelidis, C.C., Britt, H.I., Sama S., Open Software Architecture for Process Modeling: Current Status and Future Perspectives. *Proc. of the FOCAPD'99 Conf.*, Breckenridge, Colorado, USA,1999.

[13] L'Ecuyer, P., Random number generation. In Banks, J., editor, *Handbook of Simulation*, Wiley, 1998, pp. 93-137.

[14] L'Ecuyer, P., Simard, R., Chen, E. J., and Kelton, W. D. *An object oriented random-number package with many long streams and substream*s, Operations Research, 50(6), 2002.

[15] Knuth, D. E. *The Art of Computer Programming*, Volume 2: Seminumerical Algorithms. Addison-Wesley, Reading, Mass., 3rd Ed., 1998

[16] Kahaner, D., *Numerical Methods and Software*, Prentice Hall Series in Computational Mathematics, 1977.

[17] Stephen K. Park, Keith W. Miller, *Random Number Generators: Good Ones are Hard to Find*, Comm. of the ACM, 31-10, 1988.

[18] Sheldon M. Ross, *A First Course in Probability*, Prentice Hall College, 1997.