# A formal model of EIA-632 standard: An approach for emergent properties analysis

Abd-El-Kader Sahraoui

# A Formal Model of EIA-632 Standard : An approach for Emergent Properties Analysis

Abd-El-Kader SAHRAOUI
LAAS-CNRS, 7 avenue du Colonel Roche, F-31400 Toulouse, France
Université de Toulouse; UTJJ, F-31100 Toulouse, France
sahraoui@laas.fr

***Abstract:*** **This paper gives a preliminary investigation on emergent properties analysis in a systems engineering framework. the contribution is twofold, the first is to give a formal model of requirements in EIA-632 standards and then propose an approach for emergent properties analysis. The formal model is built upon existing best practice in systems engineering by mapping Pre-Post conditions for each requirement mainly the technical requirements. The emergent property as safety issues, as assumption in the model as an identified property that can be generated either through the process or through non reliable component. The process can be either a transformation (machining) or a composition (assembly).**

***Keywords : Emergent properties, EIA 632, Pre-Post conditions, Safety***

## I. INTRODUCTION AND PROBLEM STATEMENT

### A. *Introduction* :

the EIA 632 proposed requirements practices and processes for engineering a system. Many sectors have used such standards for its openness to any user on methods to be used in specifying design implementing and evaluating such systems. The issue of emergent properties appeared in most studies linked to the thematic of SoS (systems of systems) and complex industrial systems. Up to our knowledge the emergent properties study and analysis was never tackled with. Honour E. pointed out the fundamental question : "*The Missing Theory. What is the connection between emergent properties and a combination of components and interactions?"*

Designing systems was traditionally follow a lifecycle prone by many approaches and standards, the verification and validation processes were meant to identify if the systems was exempt of errors and design the right was (based on best practice. Units test were used for components and integrated tests for the final system. This was accepted mostly by the software engineering community.

978-1-4799-4647-1/14/$31.00 ©2014 IEEE

As emergent properties are these properties that cannot be predicted by analysis from the components; the question is : how can we study such issue from the design and systems approach as it was the case in software systems developing compositions proofs, or technique of obligations proofs in formal method VDM (Vienna Design Method).
The objective of the paper is to highlight related studies in emergent properties analysis and how the systems approaches developed to design a system can be beneficial to address such issue. Our work is limited to industrial systems , systems that human being designs and hence no extension to bio inspired systems or bio or chemical systems even though a lot of researchers are concerned by emergence issue in all fields as the research done at MIT by the NECSI (New England Complex systems Institute) for more details see www.necsi.org.

The paper is structured into five main parts; the first part sets the problem of emergence in systems design and gives insight of past studies on the subject mainly rom systems of systems thematic and complex systems analysis. The second part gives a general overview of emergence notion and its importance in our study. The third part covers the general approach proposed in the paper. The fourth part is devoted on formalising EIA 632 standards by identifying each of the 33 requirements among the 13 processes. Such simple formalisation comes up to set up *Pre Post condition*s on each requirement. The fifth parts proposed how emergence is analysed through such approach. The last part concludes on the approach and gives the basics for a research roadmap in emergent properties analysis in systems design within systems engineering framework.

### *B. Problem statement*

First we give a definition that addresses the emergent properties proposed by Johnson "*Emergent properties' represent one of the most significant challenges for the engineering of complex systems. They can be thought of as unexpected behaviours that stem from interaction between the components of an application and their environment. In some contexts, emergent properties can be beneficial; users adapt products to support tasks that designers never intended. They can also be harmful if they undermine important safety requirements. There is, however, considerable disagreement about the nature of 'emergent properties'. Some include almost any unexpected properties exhibited by a complex system. Others refer to emergent properties when an application exhibits behaviours that cannot be identified through functional decomposition. In other words, the sys-*

*tem is more than the sum of its component parts. This paper summarizes several alternate views of 'emergence'. The intention is to lend greater clarity and reduce confusion whenever this term is applied to the engineering of complex systems.*"

More simply to say that we can say the systems is not sum of systems as if all components are safe the whole system is safe ; similarly in operations research field a global optimisation does not rise from local optimal of its components and reversely the local optima does not ensure absolutely a global optimum.

The emergent property is not addressed explicitly as all requirement in EIA 632 don't mention explicitly but we can see , however, from all requirements related to verification and validation process that such intent is embedded in the outcome for each requirements. Let take *requirement 24* risks analysis "*The developer shall perform risk analyses to develop risk management strategies, support management of risks, and support decision making*" with one task among five of related tasks to consider include the following:

*a) Identification of technical risks, and resulting project risks, based on exposure to the probability of an undesirable consequence and the effect of that consequence for each trade-off analysis option or each physical solution representation.*

The outcome is

*a)Risk identification Technical risks, and resulting project risks, are identified, based on exposure to the probability of an undesirable consequence and the effect of that consequence for each trade-off analysis option or each physical solution representation option. Considerations include how expectations from a decision or design selection are affected by (1) commitments resulting from a choice, (2) validity of assumptions, (3) capabilities to implement and control, and (4) other organizational or technical constraints such as resources and time.*

We can see from this simple case published in EIA 632 that such standard has indirectly highlight the eventuality of risks, that is an emergent property but does says how we can avoid such risk in the rest of other task of such requirements

We can state the problem explicitly: are there necessary and sufficient conditions to reduce or eliminate unsafe emergent properties that is to say, if we want to have safe systems , how can we design our system knowing the unsafe behaviour is an unexpected behaviour that cannot be detected at the component levels composing the whole system; or we may say such solution is utopic !!!

We can structure past work into two main directions, the first developed by some author was focussed mainly on complexity of industrial safety, the second was focussed on the general issue of mergence by making abstraction of applications but can be useful in future work. We propose in the sequel key contributions in the analysis of emergence for complex industrial systems.

Hsu and Butterfield (2009) highlighted Major Impacts of emergence

• Top down development via functional decomposition
• System-of-System specification structure & contents
• Functional and performance requirements flow down
• Requirements traceability
• Requirements validation and verification

Functional decomposition will not work due to the emergent behaviours between component systems.

• There is no simple way to relate the functions of component systems to the functions of SoS.
• Lack of functional decomposition prevents allocation and flow-down.

And in that corresponding work, conclusions have been drawn as Modelling the Emergent Behaviour

• Predicting the emergent behaviour for a to-be-designed SoS will be a formidable challenge.
• Are we able to predict all the emergent behaviours? – Most likely our predictions will be probabilistic.
• Agent-based modelling is expected to be used to examine emergent behaviour as structure and pattern develop from the micro-level interactions.

– Agent-based modelling focuses on how local interactions
among agents serve to create larger and perhaps global structures and patterns of behaviour.

– Many IF-and-THENs will be defined for the bondage situations between component systems.

From such sample of earlier work, there is recognition for advanced study and the emergent properties analysis need to further study, our concern was how to deal with it based on specific standard as EIA 632 by integrating some formality and at the same time by giving a more practical approach

## II.ANALYSIS OF EMERGENT PROPERtIES

The emergence notion is widely used principally in other fields, the citations about mergence in industrial systems or man made systems date end of last century

*A.The essence of Emergence for industrial systems:*

it appears contradictory to expect unexpected event that is the first fundamental point, however unexpected events not wanted can be avoided , as an example a bad component must be avoided whether it will cause expected event (failure of the whole system) or other event

### *C. Safety and liveness properties*

we can summarize these two type of properties by
Concepts: properties: true for every possible execution

safety: *nothing bad happens*
liveness: *something good eventually happens*
Models: safety: *no reachable ERROR/STOP state*
progress: *an action is eventually executed*
fair choice and action priority

both properties can be expressed by temporal logic formulae, many tools are available composition of state transition models for expressing the systems behaviour as state charts (state diagrams in UML/SysML ) and temporal logic for assertion language expressing in variant or liveness or even safety properties

***Aim: property satisfaction***

*C. Interactions and interfaces as sources of emergent properties :*

From such basis, we can identify that the mergence is linked to design processes described previously, transformation or assembly process.

The interface and interactions are the elements that generate such emergent properties and hence the analysis is done on the components properties but on how the design has been done

For example , if we have two parts *part A* and *part B* to assemble with a *nut* and *screw*. We may have good properties for either component but safety properties can emerge from the assemble process

For such example, we set *Pre* and *post* conditions

## III.THE GENERAL APPROACH

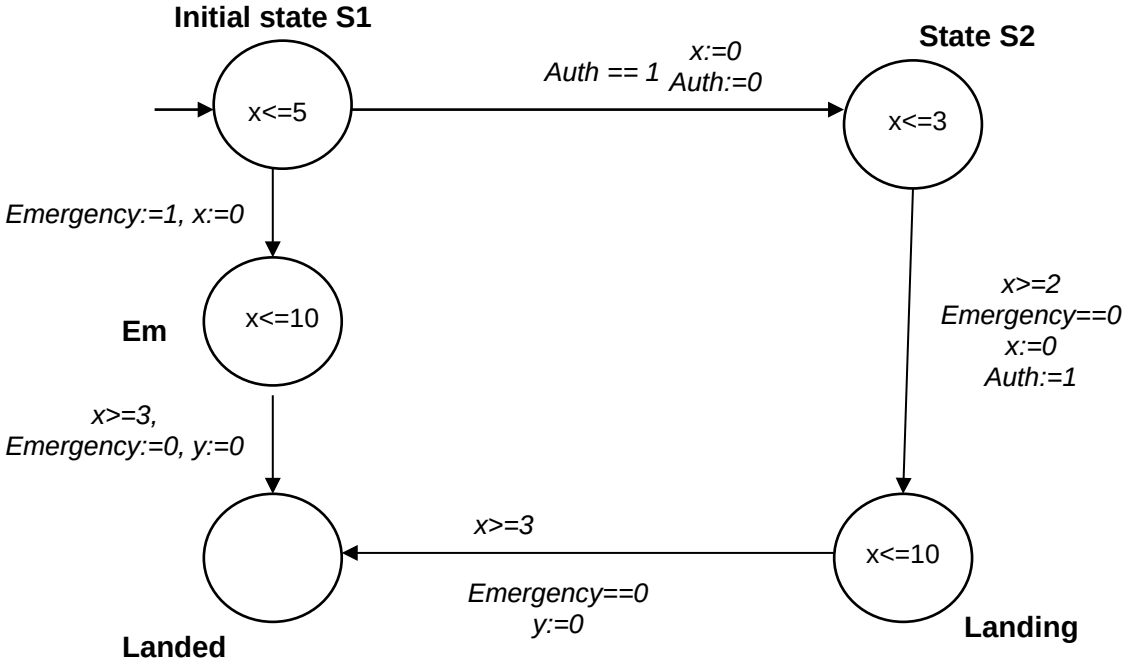
The preliminary approach is to consider that any system is composed of subsystem through some "assembly process" say assembling two parts and any component has undergone a transformation process say a machining/melting process for a mechanical part.

*A. Allocating Pre Post condition for each process*

this was applied in many applications in programming known as Dijsktra's guarded commands, in Use case specifications proposed by I. Jacobson now integrated in UML/SysML as well in business processes by Grondelle and Gulpers.

Pre-conditions and post-conditions are used in formal specification to specify an operation, which is essentially a state change. Typically, any conditions not specifically stated as post-conditions are not constrained. For example, a deletion operation in which element *n* is deleted from set S has a *post*-condition that n is not in S. To work properly, it also needs a post-condition that says that all other elements of S before the deletion, except for n are still in S afterward. When describing operations in terms of *pre*-conditions and *post*-conditions, you are stating only the essential what, carefully avoiding the how. Note the close relationship between specifications and tests.

The formalism is based on the notion of activities, the *pre conditions* that have to be met for these activities to be performed and the consequences that result from these activities, expressed in terms of post conditions.




Authorisation event *auth* is a *pre condition* added that the state is in state *airport approached* S1 and x<=5, next state S2 *landing granted*

This can be formulated

*Pre* (in state s1, x<=5, authorized )

And so on for the landing state. The invariant properties must be valid in all states
WE can illustrate this by considering either process assembly or transformation , let take a machining process on a part

***Pre** (machining)** this sets all condition being observed before launching the process*

*Machining_process description*

***Post** (Machining) ** this set condition on our come (as validation of requirements)*

*B.Allocating Pre-Post condition to EIA 632 requirements :*

this enables to derive all *pre condition* set up in EIA 632 but also do a systematic mapping of each outcome of each requirement to a *post condition*
This can be accomplished for each requirements,

*C.Invariants :*

id Pre and post condition are predicate that must be valid before and after process execution respectively, invariants are predicate that should be valid always. In this case unwanted properties are invariant as safety related properties

*D. Behaviour automata based modelling for Pre , post and invariants*

we can consider pre all previous state added to the condition and constraints before change state where the activity is executed, let take a simple example taken from case study by M. Brandozzi (2001)[2] concerning aircraft approaching airport

## IV. FORMALISING eia REQUIREMENTS

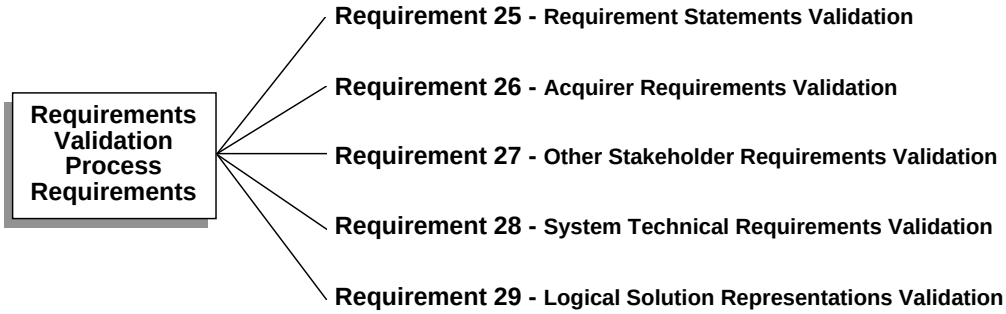
*A .Brief introduction of EIA 632 : Processes and requirements :*

EIA 632 is a SE standard "processes to engineer a systems the mostly used in Industry

*B. From semi-formal to formal assessment of requirements*

we give a flavour to such formal basis

Let consider the validation process consisting of 5 requirements




If we select requirements 5,

*Pre* (identified stackholder, method of validation
*Process* description of validation
-
-
-
*Post* (statement valid)

a. A basic formal basis for emergent properties analysis :

This enables to analyst mergence through pre post not validated , safety properties can be assessed through such predicates

b. Investigation for proof system for Evaluation process :

As it will be seen later we prototype the approach on the VDM formal notation, the proof is done through simulation

C. **Emergent properties analysis [20]**

The emergent properties through co-simulation on a case study was developed in [20]. The rules for enabling such approach was based on steps.

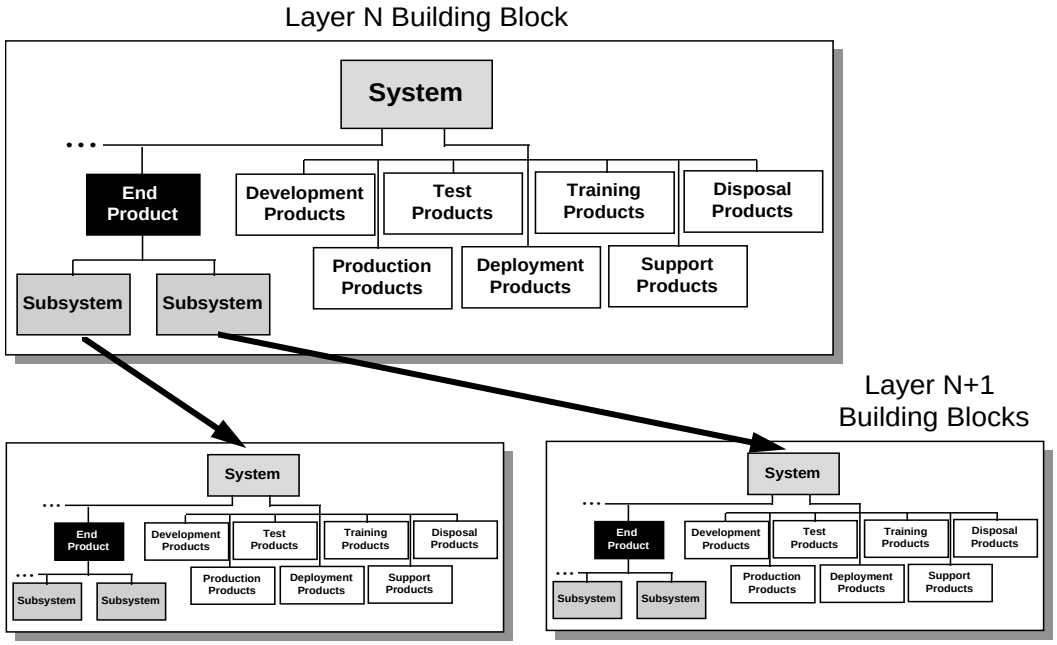
a. Orthogonal dual approach for EIA requirement and application process

We come now the essence of approach, as we have pre post conditions set for EIA 632 requirements and also for application (transformation or assembly process)

b. Methodology :

Such preliminary approach gives rise to a methodology described by the algorithm which consists at visiting the tree composition of EIA configuration

structure of a systems described by the following figure




*use recursion, you need an auxiliary data structure.*
*structure <- new block*
*push root onto structure*
*while structure is not empty*
*node <- pop top off of structure*
*visit(node)*
*for each child of node*
*push child onto structure*
*loop*

in this way all systems from top to the root are visited and hence application Pre and Post at each element

c. Guideline :

We summarize this preliminary approach as guideline,

- Allocate Pre-Post condition to each EIA requirements (there are 33 requirements)
- Allocate Pre Post condition on both design processes of the application (transformation and assembly process)

d. a tool approach with VDM formal method :

We can apply an existing formation tool that support such approach, the choice was is VDM-SL, a lot of tools are available [17, 18 19]
The Vienna Development Method (VDM) is one of the longest-established Formal Methods for the development of computer-based systems., it has grown to include a group of techniques and tools based on a formal specification language - the VDM Specification Language (VDM-SL). It has an extended form, VDM++,which supports the modelling of object-oriented and concurrent systems. Support for VDM includes commercial and academic tools for analysing models, including support for testing and proving properties of models and generating program code from validated VDM models. There is a history of industrial usage of VDM and its tools and a growing body of research in the formalism has led to notable contributions to the engineering of critical systems,

The *pre*-condition and *post*-condition together form a contract that to be satisfied by any program claiming to implement the function. The precondition records the assumptions under which the function guarantees to return a result satisfying the *post*-condition. If a function is called on inputs that do not satisfy its precondition, the outcome is undefined (indeed, termination is not even guaranteed).

VDM-SL also supports the definition of executable functions in the manner of a functional programming language. In an explicit function definition, the result is defined by means of an expression over the inputs.

*public validate : () ==> BlockItem*
*Validate () ==*
*pre Requirement<> []*
*post requirement~ = [RESULT]^Requirement;*

The toolkit has lots of useful features from syntax checking to code generation:

- Syntax checking: The syntax-checker veri_es whether the syntax of the

selected _les matches the VDM++ language specifications. If the check
passes, it gives access to the other features of VDMTools.

- Type checking: The type-checker tests misuses of values and operators

and can also show places, where runtime errors may occur.

- Code generation: VDMTools is able to generate a fully executable code

for about 95% of all VDM++ constructs. Code generation is available for

Java and C++.

- Specification manager: A manager-window displays all classes and less in the specification. It also shows the status for each _le.
- Interpreter and Debugger: VDMTools allows to execute all executable

VDM++ constructs. Debugging is also supported.

**Conclusions and further work**

The paper has given a preliminary approach how to address the difficult problem of emergent properties analysis. Two main challenges are to be addressed, a final methodology tested in many industrial cases and an associated computer aided tool for modelling

**References**


[1] ANSI : pocesss to engineer a system, EIA 632, ver 2004

[2] M. Brandozzi: From goal oriented requirements specifications to architectural prescriptions, text book, University Texas at Dallas, 2001.

[3] Blanchard, B. S., "System Engineering Management", John Wiley & Sons, Inc., 1991.

[4] Booch, G., "Object-Oriented Design with Applications", Benjamin Cummings Redwood City CA, edition 1991.

[1] EERE, "Energy Efficiency and Renewable Energy" EERE's home-webpage,2006http://www.eere.energy.gov/buildings/building_america/se_research.html

[5] EIA-632, "Processes for Engineering a System (ANSI/EIA-632)", Proof Copy by Electronic Industries, edition 1998.

[6] DSMC, "Systems Engineering Management Guide", Proof Copy by Defense Systems Management College, Fort Belvoir, VA, edition 1990.

[7] Girault, C. and Valk. R., "Petri Nets for Systems Engineering", Springer, Tokyo, 2003.

[8] Harel, D., "Statecharts: A Visual Formalism for Complex Systems", Journal of Science of Computer Programming, Vol. 8, pp. 231-274, 1987

[9] Hatley, D. J. and Pirbhai, I. A., "Strategies for Real-Time System Specification", Technical Proceedings of Dorset House, New York , 1988

[10] NASA, "Return on Investment for Independent Verification & Validation", 2B Final Report, NASA, 2004.

[11] Hoang, N., Jenkins, M., Karangelen, N., "Data Integration for Military Systems Engineering." Proceedings of IEEE Symposium & Workshop on Engineering of Computer Based Systems, USA, 1996

[12] INCOSE, "International Council on Systems Engineering". INCOSE's website, 2010 http://www.incose.org/

[13] Mathers, G. and Simpson, K. J., "Framework for the Application of Systems Engineering in the Commercial Aircraft Domain", Report Version 1.2a, American Institute for Aeronautics and Astronautics, USA, 2000

[14] Shishko, R, "NASA Systems Engineering Handbook", Proof Copy by National Aeronautics and Space Administration, USA, edition 1995.

[15] Sheard, S. A. and Lake, J. G., "Systems Engineering Standards and Models Compared", Proceedings of 8th Symposium of INCOSE, Vancouver, Columbia, 1998.

[16] Sazonov, E. S., Klinkhachorn, P. and Klein, R. L., " Hybrid LQG-Neural Controller for Inverted Pendulum System", Proceedings of 35th SSST Symposium, Morgantown, 2003.

[17] M.Messadia , A.E.K.Sahraoui: PLM as linkage process in a systems engineering framework
International Journal of Product Development, Vol.4, N°3/4, pp.382-395, 2007

[18] A.E.K.Sahraoui , D.M.Buede , A.P.Sage : Systems engineering research Journal of Systems Science and Systems Engineering, Vol.17, N°3, pp.319-333, Septembre 2008

[19] A.E.K.Sahraoui : The rationale paradigm in system development lifecycle. accepted for Int'l Journal of software and systems engineering

[20] A.E.K.Sahraoui : On emergent properties when complexity is dealt with simplicity: a case study in co-simulation. International Conference on Complex Systems (ICCS 2011), Boston Marriott (USA), June2011, 11p.