Minimizing the negative impact of emergent properties in component based complex systems

Ndabezinhle Soganile 1, Benson Moyo 2

1 Department of Computer Science and Information Systems, University of Venda, Limpopo, South Africa
2 Department of Computer Science and Information Systems, University of Venda, Limpopo, South Africa

Abstract The Software Engineering field is faced with many engineering challenges. Of particular note is the emergent property phenomenon. The concept of emergent property in software engineering originates from component based software development, where in a bid to reduce development time and costs, developers build their systems from readymade components. The emergent property is thought of as an unanticipated system behaviour that is exhibited as a result of interaction between system components. These behaviours can either be positive, meaning that they may add value to the systems original intended functionality or they can be negative, meaning that they may cause harm and at times compromise the entire security of the system. To contain the negative impact and to better understand emergent properties researchers have and are coming up with ways to predict these properties afore hand. In this paper we discuss what emergent properties are, their impact on complex systems, the different types of emergent properties and we also look at the existing methods of predicting emergent properties. Finally we offer suggestions to minimize the impact of these emergent properties.

Keywords: Emergent property, EEEP, LEEP, Complex Systems

3 Introduction

In Software Engineering emergence describes the new behaviours and patterns of complex systems, arising at the systems integrative level due to a multiplicity of fairly simpler components interacting. This interaction is not only component to component but also it is the interaction of these components or the whole system with the environment.[1] points out that it is hard to design and understand complex systems because of the interaction between different components. Different authors have diverse interpretations of the concept of emergent property “There is, however, considerable disagreement about the nature of ‘emergent properties’” [1]. CW. Johnson[1] attempts to give several alternate views of ‘emergence’ so as to reduce confusion associated with the use of the word. In this paper we will concentrate on the impact of the emergent property from a software engineering perspective and then try to suggest ways of reducing negative impacts associated with these emergent properties. Emergent properties arise more often when the system is in its deployment phase, though they can be observed in a phase during development. The diagram below shows the emergent property assessment cycle

![Emergent property assessment cycle](image)

Figure 1: Emergent property assessment cycle
Adopted from Manyphay Viengkham, IEC 2012 Young Professional Leader

2 Emergent Properties

Unforeseen behaviours and patterns of a system that are observed as a result of an application component interacting with the environment define what is referred to as ‘emergent properties’.Johnson[1] highlights that such emergent properties are used to differentiate complex systems from complicated applications. Emergent properties are not always negative, at times users benefit from them. Robust applications are delivered to users; they end up adopting products with more functionality than originally planned by designers. The biggest challenges come when these emergent properties exhibit negative behaviours in the system. The more complex the systems become the higher the potential for unpredictable emergent properties[2].

Emergent properties may arise as a good feature of a system hence adding a functionality that was not originally intended for by the developer, but at a larger scale they are usually problematic since they undermine important safety requirements.
Some engineers try to take advantage of emergent features by trying to model the emergent feature of the real world hoping that they might achieve new functionality. As an example Steve Forest said “interesting systems can be developed by taking advantage of interaction among components.

Thus in a nut shell I can say Emergence is a higher level property which cannot be deduced from or explained by properties of low level entities.

4 Types of Emergent Properties

4.1 Functional Emergent Properties

This is when parts of a system interact to achieve a particular task [4]. Components are combined to give a particular functionality of the system. An example of a car can give us a clear picture of what functional emergent properties are. A car can only be considered as a transformational vehicle only if it’s a whole. Car parts are useless as individual components but can only function when combined with other parts to form a whole which is a vehicle. The main aim of systems development is to create a system with the desired functional emergent properties.

4.2 Non Functional Emergent properties.

These are properties that represent behaviours of a system in its operational environment. They are critical because failure to meet their minimal requirements might render the system unusable [4]. Non-functional emergent properties are defined by its reliability, its security strength as well as its performance. Reliability, security and performance are dependent on the operational environment. A system can only be declared reliable when all its components are reliable that is the hardware, software and operators. Ian Somerville in his book Software Engineering explains how an error can propagate from one component to another until it brings the whole system down. Therefore to ensure reliability of a system all the components need to be tested as a whole.

The performance of a system cannot be predicted based on individual components but rather it can only be measured after all components have been integrated. Emergent properties types can further be divided into weak properties and strong properties. Weak properties are those at a low level and can be predicted using observation. These do not need priori analysis. On the other hand Strong emergent properties are at a high level, making them difficult to predict. These are the properties that emerge to the system as a whole.

In this paper we identify and categorise these emergent properties into two, the Early Exposed Emergent Properties (EEEP) and the Late Exposed Emergent Properties (LEEP). We define EEEP as those emergent properties that are detected during the component integration stage and are mostly code related. On the other hand the LEEP are those that are detected during system rollout, when the system components begin to interact with the environment. It is relatively easy to deal with the EEEP than the Handle LEEP. the challenge for many engineers is on how to predict and handle the LEEP.
5 The Impact of Emergent Properties on Complex Systems

An object like a car is made of separate parts from different manufacturers, such parts as the body, engine, wheels, steering, axel etc. These individual components have their specific functionality which doesn’t emerge up until they are all combined into a whole, i.e. these components are useless on their own, and their purpose only becomes apparent if they have been combined to form a vehicle. Thus we deduce yet again that they are properties that only emerge when the system is combined. In an equal analogy, a system or application is the sum of its simpler standalone components. Ian Somerville [4] defines a component as an independent executable entity that can be made up of one or more executable objects. These components exhibit lesser functionality as individuals than when they are combined into a system. Once built into a more complex system, new behaviours begin to surface as a result of component interaction with the environment. New advantageous functionalities begin to be seen, adding robustness to the whole system. John C. Hsu[5] highlights that these emergent properties also provide abundant potential for applications not only to overcome the problems of interoperability but also to achieve high levels of adaptability, scalability, and cost-effectiveness not possible in traditional systems. Systems exhibit new behaviours as a way of adapting to the environment. Thanks to the emergent properties systems continue to operate in the ever-changing environments and that factor increases cost-effectiveness since these systems are self-organising, there is no need for costly upgrades that are motivated by environmental changes. Taking advantage of these new behaviours, engineers are able to trace, test and validate the requirements to the good of the development process.

On the contrary, emergent properties can be harmful or undesirable. Their occurrence might have a serious negative impact to the entire system. They may compromise the important security features of the system, thereby escalating the total costs of developing the system. Additional budgets focussed on fixing the problem must then be put in place. The problem is compounded by the fact that in complex systems, it is difficult to identify the origins of the error. More time is wasted on fixing errors. The challenge facing the software engineering world is how to minimize these negative behaviours

5 Existing Strategies

Proper prediction and handling of the emergent properties helps to reduce the risk of negative impact brought by these emergent properties. Engineers and researchers are coming up with useful tools and strategies for dealing with negative emergent properties. The common methods used in predicting the emergent properties are discussed below.

5.1 Simulation models

Before the system is rolled out in the real world simulation models that mimic the actual world are developed and the system is tested on that virtual environment. It is a method implemented at lower level with weak emergence. It involves creating a model that simulates the actual system. The model is studied to derive the behavior of the combined parts. The engineers’ first work on a model to see if what they want to achieve is feasible.

5.2 Observation

This prediction mechanism is also applicable at a lower level. This involves closely watching how the individual components work then try and figure out how they would perform when combined.

5.3 Priori Analysis

This method simply means deriving some conclusion without any physical experiment. It involves trying to figure out how the whole system will perform without actually putting it into use. The only major weakness of this strategy is that crucial emergent properties are always left out since the analysis is dependent on the expertise of the engineer doing the analysis.

6 Suggested Solution

Considering the gravity on the system and the risk on business should the negative emergent properties be exhibited during the run of business, it is therefore important to find ways of eliminating these negative properties as early as possible before the system is deployed. In the following section we give our suggestion on how best the negative properties can be reduced

6.1 Piloting

Piloting is a strategy that categorises emergent properties into two views, the Early Exposed Emergent Properties (EEEP) and the Late Exposed Emergent Properties (LEEP). The first category, the EEEP is mostly associated with the code and can be detected and dealt with during the component integration phase. We recommend the use of any available strategy as define in section 5 above. The second category, the LEEP are those behaviours that the system exhibits during the deployment phase, they come as consequence of the system interaction with the real environment. They are often risky and harmful if they happen to be negative. We suggest dealing with the LEEP category in two ways. Firstly, the system must be tested on a simulation platform, this will provide an opportunity to identify and eliminate a number of these mal emergent properties. Secondly, the system must be tested on a pilot real-world platform before deploying or delivering to the client. This will give engineers an opportunity to further
identify and deal with these properties before the final deployment of the system.

7 Conclusion

It is practically impossible to fully predict all the behaviours of a complex system since the environment is not a constant. It is ever changing, therefore complex systems are bound to exhibit some strange behaviours that are dependent on the environmental changes. However the impact can be minimized by adequately testing the systems before putting them to work. In this paper we recommended piloting as one strategy that engineers can use to minimize the impact of the emergent properties. Predicting and eliminating the emergent properties before the deployment is a good way of ensuring that their impact is not felt when the system is put to work.

Reference List