

Towards Organic Data Processing

Paul Perez, London, 12/07/07
paul.perez@pymma.com



Preamble:

One day at university, one of our physics professors arrived in the lecture theatre with a crafty smile and after waiting for complete silence declared: "Students, today we will prove that God does not exist!" And with a certain mischief, he wrote on the blackboard the second principle of thermodynamics. In short, for those who's studies do not lie in this field, this principle indicates that the "disorder in a closed system can only increase with time". Having stated this, he continued his reflexion: "How can one imagine a very powerful creator whose creation would do nothing but be disorganized? Indeed, this very powerful creator should have created a system which is organized and not the reverse, therefore what it is necessary to show is that this type of creator cannot be considered for our world".

At this point students pointed out to him that contrary to what he said, everything in our universe tended towards an organization. The elementary particles form atoms, the atoms form molecules, the molecules form cells, the cells form living beings and even those living beings organize themselves socially. Annoyed, the professor had to acknowledge the accuracy of this counter argument.

Towards organic data processing:

Well, what is the relationship of all this with IT Architecture? you ask. It is interesting to see how a field such as application development or even information technology as a whole is approaching real life and begins to follow the same evolution - roughly speaking.

Report N°1: Like a physical system, an IT application tends towards disorder.

We have all noticed that with time an IT application becomes increasingly complicated to maintain and to modify. With time, the complexity of the application increases, its capacity for integration decreases and this, despite every effort of the architects, project leaders and developers. That resembles much the entropy which one finds in nature. Just like closed physical systems, the information processing systems are disorganized. One can do nothing about it, this is a natural law to which all the developers of the past, of today and of tomorrow will have confronted or will need to confront at some point.

Report N°2: But, as in nature, there are also things that organise in spite of the fundamental principle.

To support what I say here I present 2-3 stages which have marked the history of programming, and to

emphasis the point a little, I show an analogy between our programming and organic worlds. The first stage in the organic evolution of data-processing applications was the introduction of encapsulation into the programming object. It was the first time that we created a separation between a section of code and the remainder of the application. One can compare this with the first cells, which could, thanks to an external membrane, be isolated from outside. Once such components appeared and the objects became more complex and autonomous, they further resemble the complex multi-cellular organizations. Until recently, these components communicated with one another in an anarchistic way. But more recently it has become necessary to organize architectures full of components, and to find standards of communication between components, as well as rules of "good community". This is what we start to observe with the appearance and standardization of communication protocols. Although this last stage is more social than biological, the evolution towards an organization copied from the real world is undeniable.

I hear you argue that this phenomenon is perfectly normal, "Dogs make dogs and the cats make cats". There is thus nothing astonishing that a human creation copies and takes as a starting point the human organization. On this, I can only agree with you entirely. And then, why bother to write an article on this subject? Well, it often makes things clearer when the matter is said and written.

Now that we are convinced of an evolution towards an organic organization, let's try to be a little more precise and give slightly more concrete architectural examples showing the similarities between programming and everyday life. Let's see, thanks to two examples, how this principle applies in the work of an architect.

Ex N°1: Asynchronous protocols:

When making a choice of protocol between two applications and it is possible to choose between a synchronous or asynchronous protocol we tend to choose the latter because in a vast majority of the cases, it is more effective, more flexible and less resource heavy. In the same way in "everyday life", the asynchronous exchanges account for 99,9% of our communication because they are much more effective. Let us take another example a little more directed.

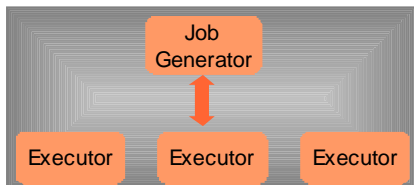
Ex N°2: Push or Pull:

In a banking information processing system, the first application A generates jobs to be carried out for other applications B1 B2... to Bn which will carry them out.

Towards Organic Data Processing

Paul Perez, London, 12/07/07
paul.perez@pymma.com

The communication between the generator and the executors of jobs can be of two types, either "pull" (where the executors ask for continuous work from the generators) or of type "push" (where the generator sends work onto the processors)



But how can this occur in "everyday life"? Generally speaking, in a company, it is the owner who distributes work and not the workmen who come to seek work. The natural rule is a communication between owner and workman of the type "push". And well, in the same way, in information processing systems, it is also the most effective solution. We are often brought in to carry out audits on the architectures of our customers to solve problems of performance. In performing such audits we regularly come across problems in pull type architectures (where it is the employee who will seek his work). They are often used because they are simpler to set up and function well for weak load. Alas, they are not very effective and create points of contention in production as soon as the workload grows. While migrating towards a "push" type architecture (where it is the owner who distributes work) we remove these points of contention (bottleneck) and obtain important gains in performance and scalability. Once again (**Careful:** it is not systematic!) the "everyday life" can inform us on the action to be followed.

From these observations, we can see that the evolution of our information systems towards organization of an organic nature has started and will accelerate in the future years. Future architectures will connect increasingly organized, complex and autonomous entities. And the architects will have to take "everyday life" and the relations between people as a starting point. In short, they will need to look at human organization in order to define and conceive the architectures of tomorrow.

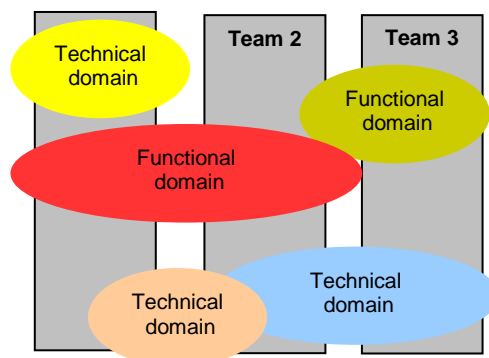
Not yet organic organization!

If this evolution of information systems seems obvious, it is regrettable that the organization of business teams and IT does not follow this evolution. They are still organized as they were it in the 70's and 80's. Teams are created around a project, and they live

and die with it. They are responsible for it and occupy themselves with it, as at the time when the applications were independent and where integration and communication with the outside was negligible work in comparison to the workload of the project and the processing within.

Obviously it is no more the case today. There is an obvious shift between the technical organization and business. This shift between the technological developments and the organization of the information services can run an important risk to the companies. The IT teams are no longer able to manage a perimeter of competence defined on obsolete criteria correctly. The perimeters of the various teams overlap and certain overlapping areas are not dealt with correctly.

The reorganization of the IT teams towards a more organic organization would facilitate the installation of a services architecture. The scope and area of each team could be correlated with either one (or several) technical fields, or one (or several) functional fields represented by one (or more) services.



Unfortunately, it is very difficult to make executives comprehend the need for a reorganization of the IT teams. The critical blocking point in this is the allocation of the budgets to the IT departments. Each service, as everywhere in the world, wishes to have the most important budget possible. If one limited the scope of the teams to a technical field (transversal for example) or to the supply of a certain number of services, it would considerably reduce their attributions and their budget. The most important teams, those with the largest number of members and therefore most influential, are often opposed to this type of reorganization. They have at the lead a number of influential managers who attempt to control the maximum number of key areas. By preserving a vertical attribution of the budgets such as those that have existed for decades, they preserve and even

Towards Organic Data Processing

Paul Perez, London, 12/07/07
paul.perez@pymma.com

increase their competences and attributions, but prevent any evolution towards an organic organization of the IT teams. It is thus with the general IT movement to become aware of this problem and to impose this reorganization. To be honest, few of our customers are ready to take the step.

Fortunately, certain companies take initiatives. One of our customers, a large airport trust company, recently defined a new organization with various types of teams to meet the needs for a more complex organization. A first type deals with the infrastructure (Bus - ESB for present case). A second team manages integration (legacy applications, internal and external partners). A third implements the trade services. Finally, a fourth closer to the business, deals with the orchestration of these services. What is this organization worth? It is worth only what the men are worth who make it up, of course. But whatever the result, there was a real intention by the Information Systems Director to try and evolve things and to take into account the evolution of their IT into a more complex organization, or at least different from those that have existed for decades. It is an interesting step which will have to be followed.



Conclusions.

By comparing the evolution of the IT applications with the evolution of living beings, we have tried to show in this paper that our IT evolves to what we call “organic data processing” where each entity like the beings of the living world become, whilst evolving/moving, more and more of complex, communicative and autonomous. Consequently, the rules of architecture must also follow this evolution and take as a starting point the rules of organization of the living.

Unfortunately, this evolution, met at the technical level, is only found very seldom in the organization of the business teams or those of IT teams. The vertical organization defined around the project is still the rule in many companies. As we said, this creates a bad correlation between the fields of competence allotted to the IT teams and the technical or functional fields

which they use.

A reflective period and an important reorganization awaits the companies wishing to remain competitive and ahead within their industrial field. This reflection must be carried out initially by business management and then the IT services. This reorganization must be undertaken by all those which take part in the development of the trade processes in the company, and must realise that organic data processing is already present in our information systems and will surely be there for several decades.