

# Life Is a Series of Trade-Offs and Change Is Accelerating!

By  
**John A. Zachman**  
© 1998 Zachman International

The older I get, the more I am aware that life is a series of trade-offs and as much as we would like to “have our cake and eat it too,” that is a figment of our imagination. It is simply not realistic. As soon as we start thinking we can have it both ways ... we are positioning ourselves for the inevitable, “rude awakening!” We would be far better served to clearly understand the choices we are making, set our expectations correctly and mitigate the downstream impacts of the choices we make.

Life’s trade-off in its most simplistic form is between the short term (“immediate gratification”) and the long term (what’s “good for us”). The way this trade-off reads out in the context of Enterprise Architecture is:

short term options vs long term options

implementation vs integration

point in time solutions vs infrastructure solutions

expense based approaches vs asset based approaches

implementation optimization      vs      Enterprise optimization  
(at the expense of the Enterprise)      (at the expense of the implementation)

pay me now or pay me later      vs      takes too long and costs too much

etc.

etc.

Some times, and for some aspects of the Enterprise, you may want to take the short term option and other times for other aspects, you may want the long. When you want the short, you want to mitigate the downstream impacts of the suboptimal approach ... when you pick the bng, you want to find clever ways to satisfy short term demand without compromising enterprise integration.

The Framework for Enterprise Architecture (the “Zachman Framework”), is a useful analytical tool to help you think about the trade-off, correctly set your expectations and devise strategies to mitigate the affects of your short or long term choices.

The Framework is a classification scheme for descriptive representations of any complex object ... in our case, an Enterprise. It identifies a total, comprehensive set of descriptive representations (that is, “models”) that are relevant for describing the Enterprise.

Some people seem to think that the Framework *prescribes* that all the models have to be made explicit before any implementations can take place. *That could not be farther from the truth!!* If that is your impression, you need to return to the basic concepts of the Framework. It is only *your* (short term or long term) methodological choices that determine which cells (or “slivers” of cells) are produced over the process of implementing systems, automated or manual. The Framework is totally neutral relative to methodologies. In fact, ANY methodology and EVERY methodology ... whatever actual work you are doing ... can be mapped against the Framework to determine what your choices and underlying assumptions are, simply by identifying which cells (or “slivers” of cells) you are producing and managing, in what sequence.

This misperception about the Framework being “all or nothing” probably stems from the fact that the Framework does, in fact, identify the total, comprehensive set of relevant models for the Enterprise, even though it presupposes nothing about which models (or “slivers” of models) you take the time and spend the money to make explicit.

Actually, if the Enterprise exists, all of the models already, by definition, exist. If you do not take the time and spend the money to make any one or more of them *explicit* does not mean that the models go away. It just means you didn’t make them explicit. They are *implicit*. That is, you are making assumptions about them. In fact, if no models have been made explicit, someone is making or has made assumptions about every one of them, and the assumptions are imbedded in the reality of the operating Enterprise, that is, in the glob of spaghetti, the implemented “systems,” automated and/or manual ... and it’s very difficult for anyone including the programmer, or methods analyst, or Industrial Engineer or whomever made the assumptions, to remember or to figure out what assumptions have been made, or whether they are any longer relevant, or whether they are coherent, or consistent, or integrated across the scope of the Enterprise, or integrated horizontally (with other types of models), or integrated vertically (with other models of the same type), and so on. In fact, it is especially difficult to figure out what assumptions have been made months or years after the fact.

It is obvious that in a changing environment, the ultimate, long term, best interests of the Enterprise would be served if “all of the models were made explicit, Enterprise-wide, horizontally and vertically integrated, at excruciating levels of detail.” In this case, anyone who was authorized could tell immediately what assumptions have been made, as the assumptions would be explicitly described in the models. It would be very easy to evaluate the assumptions in the light of changes taking place in the marketplace or the

environment external to the Enterprise and to determine what very specific components internal to the Enterprise have to be changed. You could know precisely what has to be changed and what cannot be changed, what other of the models would be impacted by the change, how much it would cost to change, how fast it could be changed, etc., etc. If the external environment starts changing more and more rapidly, it will become more and more imperative to have the models already made explicit so you can very quickly assess the impact of the changes and decide what parts of what models have to be changed and then actually effect the change.

This brings me to another criticism I have recently heard, that the “Zachman Framework” “is rooted in the slow change environment of the ‘80’s. It has little value to companies with high change environments.” *Obviously that could not be farther from the truth either!!* If this is your perception, you need to do some research into the fundamentals of change management. If you are going to manage change, it simply means that you are going to be driven to take long (or longer) term approaches to Enterprise Architecture. You are going to have to build (that is, make explicit) the models or “slivers” of models that you are going to want to change.

For example, let’s assume the worst for a moment ... that you are in a high change environment and that you do not have any explicit models, that you have been optimizing the implementations and not building or maintaining any models for the last 50 years, and that all the assumptions about all the implicit models of your Enterprise are imbedded in the implemented systems ... automated and/or manual ... and that suddenly you have to change. Basically, you have two options: first, you could build new systems. In this case you would then have to try to figure out how to turn off the old systems when you get the new ones built, or else at worst, have conflicting implemented systems or at best, redundant systems. And, without any models, it would likely be rather challenging to figure out what systems to turn off and how to turn them off. Also, the smaller the new systems you build and the faster you can implement them, the more discontinuity and redundancy you are likely to build into the overall set of Enterprise systems, because you wouldn’t know how the new systems relate to any of the existing systems, since no models exist that describe the old ones.

Clearly, this is the approach we have been using for the last 50 years. It is the *legacy creation approach*, which I doubt that I have to explain, has some severe shortcomings, *particularly when it comes to change!!!*

Your second option would be to try to re-create the models first, reverse engineering the assumptions out of the running (manual or automated) systems up to whatever row or column or cell you want to change so that you could assess what is impacted by the change ... then reanalyze, redesign, reconstruct, retest and rerun. If the Enterprise has any complexity at all, and if the changes you are trying to make are extensive or interrelated, this might take *a lot* of time. In fact, you may not be able to stay in business long enough to reverse engineer and get them changed!

Take General Motors for example. I have never been to General Motors, but I understand GM is around 3 years into a 5 year project in which they are attempting to take something like 28 months (over half of the cycle time) out of the current product cycle ... massive, wrenching changes to their business! And I suspect, *there are no models* ... and I further suspect, there is no one around anymore who knows the assumptions that are imbedded in the actual operating Enterprise, manual and/or automated, that is, probably, no one knows how the business works ... at least, no one knows how it all works together as a contiguous business. I understand that currently GM is sitting at about 48 months from “concept to customer” whereas Toyota is at about 28 months and Chrysler is down to around 18 months. GM’s market share has declined in the last 13 years from around 49% to about 32% today. I have a feeling, the big question is, are they going to get the models reversed engineered so they can figure out how to get the business changed in time to sustain their asset base and stay in business? If my suspicions are accurate, I could argue, it would have been *a lot* easier and faster and maybe cheaper and certainly safer if they already had “all those models, Enterprise-wide, horizontally and vertically integrated at excruciating levels of detail ... etc., etc.” I suspect that somebody is saying something like, “why in the world didn’t we take longer term approaches to our Enterprise Architecture” ... or worse, “why didn’t YOU take ... etc., etc.”

Anyone who does thoughtful research into change management will find in all of history, older disciplines that have had to deal with change to complex objects, for example, Architecture and Construction, Manufacturing and Engineering, ... they all start with the descriptive representations of the object ... the “architecture”... the drawings, functional specs, the bills of materials, etc., etc. No architect or contractor in their right mind would start making changes to a 100 story building (or ANY building for that matter) without first going to the “Architect’s Plans,” the descriptive representations of the building, the architecture, the “models.”

My friend Bernie Boar, who recently has written a book, “Constructing Blueprints for Enterprise IT Architectures” (John Wiley and Sons 1998), said at a conference that we recently ran under the auspices of ZIFA, “when the Titanic hit the iceberg, the Captain did *not* call for the architectural *principles*, he called for the architectural *blueprints*.” At the point in time when you hit the iceberg, you have to be able to look at the architectural representations, the “models,” to understand what has happened and to decide what you can do about it. I would observe, when the Enterprise gets into extremis (hits the iceberg) and has to change quickly ... if the models are not already explicit, there is going to be a real question whether you can build from scratch or whether you can reverse engineer and change what you are before you run out of time and the thing sinks! And the faster the changes start coming, the worse the problem is going to get!

If the world was a perfect place, in a changing environment, it is clear once again that the best of all possible worlds would be to have “all of the models, Enterprise-wide, horizontally and vertically integrated at excruciating levels of detail.” Therefore, this would represent the best, long term architectural strategy for the Enterprise.

On the other hand, it is also clear that the world is not a perfect place and you are going to be tempted to compromise the long term strategy every time you want to implement something, because it will take some time and cost some money to preserve the long term architectural continuity ... and the pressure for immediate gratification, because the stress levels are out-of-sight, is enormous!

The question you are faced with for *every* implementation is, “do you want the long term approach, or the short term approach?” The problem is, “you cannot have your cake and eat it too!” If you want quick implementations, you take the “you start writing the code and I’ll go find out what the users have in mind approach.” If you do this exclusively for any length of time, you will find you have built something that is impossible to change, a “legacy.”

On the other hand, if you want to build something that will accommodate change, you will have to build and manage the models that will constitute the base-line for managing change.

I would argue that it would be *very good to know* (be conscious about) which approach you are choosing for every implementation and to understand what its implications are ... and to make sure everybody else in the Enterprise who is significantly interested is conscious about it as well, so the expectations are correctly established. Otherwise you are very vulnerable to a lot of hate and discontent when the stark reality of the expectation gap, with *whichever* choice you make, inevitably presents itself!!

The Framework is helpful for analyzing the options to understand what they are, what the implications are and to communicate and have objective dialog about what they are.

For example, which cell or cells or “slivers” of cells are we going to make explicit? Are the cells (models) we make explicit going to be Enterprise-wide (integrated), and if not, do we care about the fact that they are not? (Note: the Enterprise *will* care whether some cells, or some “slivers” of some cells, are “integrated,” or not. Other cells or “slivers” of cells they *will not* care whether they are “integrated” or not.) In the cases where we do care, what (modest amount of) additional work could we do to mitigate the impact of not producing the entire Enterprise-wide model, or would we be better served by a longer term, more integrated approach?

Further, are there cells, or “slivers” of cells, above the one or ones we are working on that will not be made explicit and if so, are we willing to assume the risk of making assumptions about them? And again, are there other cells in the same row whose natural integrity we will compromise if they are not made explicit and related to the chosen cells? And, in both of these cases, are there things we can do to mitigate the impact of making the short term trade-off with a modest amount of additional work, or do we have to consider a longer term approach?

If you are going to compromise the long term strategy, you probably should have a pretty good reason for doing it, making sure the implications are exercised, expectations correctly set and mitigating actions taken appropriately.

I have a number of friends, notably Doug Erickson of Data, Applications, Technology Associates, who declare that it is actually cheaper and faster to take Enterprise-wide, top-down architectural approaches to systems implementations because the cost reductions of removing redundancy and the cycle-time reductions of leveraging reusability vastly reduce development cost and time to market. This is the lesson that has been painfully realized by Western Manufacturing over the last 20 years, that it is actually cheaper and faster to build products that have no defects (the long term approach) rather than to build defects into the products and then attempt to remove the defects after the products are built (the short term approach). This is the “quality is free” idea and as the argument goes, it applies to Enterprises as well as to automobiles, microprocessors, airplanes, bicycles, machine tools or whatever.

In summary, the Framework defines the total, comprehensive set of models relevant for describing the Enterprise and by so doing, it defines the optimum, long term architectural strategy. However, the Framework is inert relative to your methodology. It only helps you evaluate the methodology in terms of which models you will produce and how they relate to the optimum, total set. In this fashion, you can make a reasoned judgment whether to take longer term approaches (building and managing some models), or shorter term approaches (just getting the system implemented), and/or to attempt to balance the two by finding a kind-of middle ground, selected-cell, “sliver-based,” compromise.

If someone proposed to me to do some implementation work in my Enterprise *without* using the Framework, here are the kinds of things I would ask them:

1. What models are you going to build? (Then, I would map those models to the Framework cells or “slivers” of cells to see how I felt about it.)
2. How are you going to deal with quality ... making sure that what you produce meets the requirements of the Enterprise? (Then, I would see if the logic takes me straight up the column(s) from the cells being produced to row 2.)
3. How are you going to deal with integration? (I would look at how they intend to deal with “scope integration”, that is, Enterprise-wide integration, as well as “horizontal integration”, that is, integration with other models in the same row(s) they are building ... are they going to optimize one model at the expense of another model? Or, are they going to create redundancy?)
4. How are you going to deal with change? (Are they going to build models and keep the models they build as a baseline for managing change? Also, I would expect them to leave the models and the model of the models (the “meta-model”) with me because these are key for me to manage change when the model-builders/system-implementers are long gone.)

5. If they are not going to build any models at all, then I would ask myself, “do I really want these guys to help me or not?” ... because I happen to be convinced that change is a *big* problem and that longer term approaches are imperative.
6. Last, I would ask myself whether I am happy with the short term or long term choices that they are making explicitly or implicitly as a result of their methodology.

Life is a series of trade-offs between immediate gratification and what’s best in the long term. The Framework is helpful as an analytical tool to have meaningful, objective dialog about these choices and to make these choices or to find a right balance between the choices for the Enterprise. At the heart of managing change is Architecture, Enterprise Architecture. If you are going to manage change, you are going to take longer term approaches to Enterprise Architecture. The Framework for Enterprise Architecture could be extremely valuable as a management tool, central to making the long term/short term trade-offs and managing any complex Enterprise in an environment of increasing complexity and increasing rate of change. The Framework clearly is not “all or nothing” and at the same time, explicit, architectural representations (models) clearly are fundamental to managing change. In fact, I can find no examples in other, older disciplines where change is managed in complex objects without explicit models.

Even though the Framework is totally neutral to whether you are building and managing models or not, if you find yourself in an increasingly complex and changing world, “someday, you are going to wish you had all those models, enterprise-wide, horizontally and vertically integrated at excruciating levels of detail!” The key lies in figuring out how to find the balance between the short term versus the long term, between implementation versus integration, point in time solutions versus infrastructure, how to satisfy short term demand and at the same time build something coherent over time. The Framework *may be* your only hope for figuring that out.

*John A. Zachman  
Zachman International  
2222 Foothill Blvd. Suite 337  
La Cañada, Ca. 91011  
818-244-3763 (Phone and Fax)  
johnzachman@compuserve.com*