Live Application Design (LAD) – From Evolution to Revolution

January 23, 2024

By: Chet Hall and Lance Kerwin

Curious Learners Enhance Problem Solving Techniques

For as long as I can remember I have been mired in how things work while trying to understand how to make processes more efficient. You can say it is my constant state, to make everything I experience more efficient. My first remembrance of thinking about process efficiency was when I was about 11 or 12; I was cutting our lawn and since it was taking away from my play time, I started thinking about how I could be more efficient at cutting the grass and cutting it properly so my Dad wouldn't make me recut it.

As I mowed, I thought – should I cut from the edges first, make as many straight lines as possible, cut in rectangles or circles, start in the center and work my way outwards, are 180 or 90 degree turns more efficient? What about hills, do you mow top to bottom or side to side, if there are dirt areas, do you mow over them are go around them. Yes, it is an obsession that I still have to this day; I still mow my lawn every week and think critically about the path, turns, hills, edges, shapes, and the time it takes me to do it. A few weeks ago, as I was mowing the grass, I contemplated finally testing some of my long-studied theories and hypothesis but figured if I built a spreadsheet to complete the analysis and then tested the results, my wife would call the authorities.

Learning at a Faster Rate through Communication

Over the years my processing efficiency obsession has served me well – most of the time. It led me to the strategic planning world about 20 years ago and I have developed many models from it, where I have practiced the theories on small businesses and non-profits. The models I have developed have allowed me to turn a typical 3-month strategy process into a successful 8-hour process for many happy clients.

When I started working with a small company named Intact Partners to help bring their ReAccess solution to market, I never dreamed it would lead to the efficiency breakthrough it has. ReAccess was developed to help support the sunsetting of MS Access as Windows 7 was phased out by Microsoft. These two software products are now considered non-compliant with Federal and State security laws.

I knew right away ReAccess was going to be more efficient than standard development processes, just not to the extent it is now. The speed of LAD/D with ReAccess for "idea to operations" is still hard to comprehend, especially for the software development industry. In case you are not familiar with the typical software development lifecycle (SDLC) it is: Planning, Analysis, Design, Development, Integration, Testing, Implementation, Operations.

Building Systems the "Old Way" is Broken

To better explain the SDLC I will use the analogy of building a house. Building a house or building software applications has pretty much been done the same way for the past 30 years. There has been an influx of new tools to build houses (and software), however, the process has been the same; and what is important, the time to build a house or software still takes about the same amount of time – considering size – as it did 30 years ago. I know what you're thinking, how can this be true given all the advances we have seen in software technology? Yes, you are correct, we are lightyears ahead of where we were 30 years ago; however, we are only building more modern solutions, the premise is still the same – applications take some input, process it, and provide some output. Yes, the systems were more basic 30 years ago, but the development process was the same and the time to build a software system is still just as long.

One reason for the consistent time to build a system over the years, is the number of resources has been reduced in both building processes. These building processes are straight forward - need, imagine, analyze, design, document, construct, inspect/test, complete, acceptance. The tools for building have evolved over the past 30 years. For home building we now have computers & Virtual Reality to help with imagine, analyze, design, and documentation. We have new power tools for construction. For software, we have the exact same thing; and still the process and time to complete a project has not changed much. Until now.

New Rapid Prototyping Tools Leads the Way

With the new Azure cloud tools and services Microsoft (MS) has developed over the past five+ years, Intact Partner, Inc (Intact) has been able to create a SaaS tool (ReAccess) which provides the business user (as Microsoft calls it – Citizen Developer) with the capability to build a large house (I mean application) themselves, with built-in security, room for everyone, ability to easily add or change rooms, and an emergency preparedness plan.

These tools have been around for some time now, ReAccess is the new tool on the block. ReAccess is the old MS Access database app - moved to the cloud, without the limitations MS Access had. The combination of this new tool (ReAccess) and a new process created by Intact named LAD/D (Live Application Design/Development℠) has revolutionized software development. Before we discuss the details, let me tell you of a recent actual event of ReAccess and LAD/D in action.

Rapid Building in Action

On Friday at 2:59 PM I received a phone call from a long-time friend and colleague, Lance. We spoke for about an hour to catchup. Lance mentioned some work he was discussing with a local university. After he told me what the work was about, I suggested he use our ReAccess tool to do a proof-of-concept. He wanted to know more since he thought it could be a good fit for the data collecting and analysis he needed. I told him all we needed was two hours to design, develop, and deploy the app, he said he would let me know; I could feel the skepticism. Later that evening, at 6:55 PM, to my surprise Lance sent me a text and said he had some free time on Saturday (the next day), and he could meet at 12:30. Since the solution we discussed needed heavy data analytics, I called our Power BI analyst, Jaron to ask him if we could join us for the LAD/D session.

We met at our LAD/D room which has 3 tables, 2 large whiteboards, and several large monitors. We all three arrive around the same time, exchange pleasantries, and briefly discuss the app Lance was hoping to create. Then for about 30 minutes I showed him a demo of the ReAccess app and some apps we have built for clients; and then two apps we built using the LAD/D process. Lance asked a lot of questions about ReAccess and LAD/D. We discussed the Azure Cosmos database and how ReAccess does not create typical relational database tables like we were custom to developing in the past. Cosmos is technology where we can "flatten" the files which makes it easier to support the citizen developer. We discussed that with ReAccess and the LAD/D process, where we use Entities as our menu items, create Static DDL's (drop down lists) for common lists/types, identify Relationships between the Entities, and finally how using Relationships and Entities we can configure Dynamic DDL's. These items are the foundation for LAD/D.

After the 30-minute demo and discussion, we dug into the LAD/D solutioning. On the whiteboard where we had previously laid out the four components of LAD/D – Entities, Static DDL's, Relationships, and Dynamic DDL's – we begin to build the list for each.

This is where a good facilitator asks the right questions to guide the app owner through the LAD/D process. It's not going to be perfect, it just the start. My experience with facilitation is to get the app owner(s) to think through the solution outcome they want to design/develop and let's get the four LAD/D components filled out as much as possible.

As mentioned previously, we are creating an app for local university Sports Athlete analysis. We determined the Entities for this app will be Sports Teams (Organizations), Athletes, Coaches, and Performance Profile. We came up with two Static DDL's of Coach Type and Sports Teams. We determined the Relationships will be Athlete-Team, Athlete-Coach, Athlete-Performance Profile, and Coach-Team. We then decided we will need Dynamic DDL's for Athlete's Name and Coach's Name. This initial step of creating these components took about 20 minutes. The key here is to not over think it. Get these items on the board the best you can, and understand it is going to change/evolve as you proceed. One thing you may have noticed is we have not discussed Entity Attributes (data fields) yet. There will be plenty of opportunities to add the Attributes once we get to the next step of formatting the configuration file. For the solution Attribute configuration, Lance already knew the performance data that is currently being collected on the Athlete; this will allow us to easily add to the Configuration file which will support the data feed into the app.

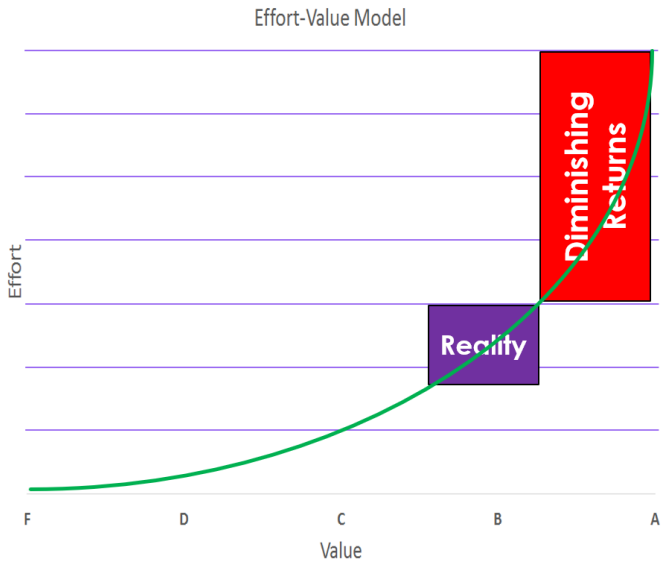
As we discussed the LAD/D concept and how it fits into the ReAccess solution, we discussed the basic concept of Apps – collect data (input), process the data, then produce a formatted output. Since we do not know the

specs on these data feeds, we will have to get this info later in order to configure the import feature.

Maximizing Human Intellect to Design Solutions

Before we move on to discussing the formatting of the configuration file for the ReAccess app, we need to take a step back and discuss the most important and most difficult component for LAD/D; the human element. Several years ago, I wrote a book called "VIGOACRE" and I constantly refer to the human element area as "Chapter 8" - the Effort-Value Model. Here are some excerpts from Chapter 8 to give you an understanding of the foundational development of LAD/D.

The Effort-Value Model[SM] (EVM) chart below is a concept we created which reflects the effort vs. value of an individual or organization which is results driven. The model's result is based on the exponential amount of effort needed to drive the value of the results. If you view this from a student's perspective, then there is little difference in effort between a failing grade (F) and an average grade (C), and a huge difference between an above average grade (B) and a top grade (A).



This concept is based on many factors, including how we perceive results. Most of us think the school grading process is linear, meaning the amount of effort to get from a "D" to a "C" is the same effort to get from a "B" to an "A". My observations over many years provide a different conclusion. What does this have to do with communication and LAD/D? Wait a minute and you will get there.

Why is EVM an exponential model and not linear? Well, let's look at the model as a numerical value. Zero to 59 is an F and then each grade level after are divided into equal values of 10, 90 to 100 would be an "A". We are conditioned to view the value from 0 to 59 as no effort and this is not true. Our perception in life is we all start somewhere in the 50's with no effort, and to get to the "D" or "C" level all we have to do is show up and put in a small amount of effort. Just showing up and putting just a little effort may get you an average "C" in school, and it will get you average in life. I believe the reason for this misconception is we don't clearly understand the effort vs. the value we receive.

My experience with this EVM is if you make the effort needed to be an A performer you will not be as efficient and effective as you can be as a "B" performer, this is the "Diminishing Returns" represented in the above graphic. Diminishing Returns is where you can exert a tremendous amount of effort with very little increase in value.

Success Defined in a New Lens

In my experience, success is not about getting an A; success is about the sustainability of above average work and how far we can take it by continuing to improve. The Japanese call this Kaizen - a Japanese business philosophy of continuous improvement of working practices, personal efficiency. Kaizen was first implemented in Japanese businesses after the Second World War, influenced in part by American business and quality management teachers. Kaizen has been credited for the high quality and efficiency of rebuilding Japan after the war. This kind of improvement and sustainability will make you awesome!

In business, this is where a lot of effort is spent to drive perfection, when there is little or no chance of getting to perfection, and usually there is little need to obtain perfection. Why, because the need for perfection only lasts for a very, very short period of time; the business changes and perfection then becomes a point of diminishing returns.

Why can't a well-planned and managed project be completed to perfection? The answer is simple: a perfect solution does not exist. You may believe you always complete your projects on time, on budget, and they are exactly what the project owner wants. Sorry, you are in fantasy land, and here's why. By definition if you have

more than one person involved in the project, no matter what stage, two people will never have the same vision or expectation of what, how, when, and why about the project. Therefore, there are no perfect projects.

Secondly, if there is only one person working on the project and they always agree with themselves, then there is still a problem with the perfect project; and it's because when the vision for the project is first established and planned, things will change. The world is not static. The world as well as business is fluid, very dynamic; it is constantly changing. You can only be perfect for an instant. With every rule, there is an exception; there are times in an organization when perfection may be necessary such as the airline industry, medical industry, etc.

Look at Henry Ford and the Model T. It was seen at the time as the only car you would need, and you could get it in any color, as long as it was black. This lasted from 1908 to 1927 and Ford sold more than 16 million Model T's. Then, in the 1930's, less than 10 years after the Model T was last built, we had a completely new type of car; luxury cars with radios, heaters, large fast powerful engines and automatic transmissions.

Designing and Solving for Big Problems is the Way

If the product is not changing with the world, then you are developing an obsolete product. How do you solve this problem? The same way you solve most problems, by communicating and being ready to update your solutions as driven by the demand. Team communication is about compromising and getting to the best solution as fast as possible. If you wait until you have a perfect solution, you will never get out of the starting gate.

My solution for this challenge is to use the 80/20 rule. Try to get 80% of what you want and keep building on the 20% you did not get. This is easy if you can get your team to understand that there is never a perfect solution. If you keep inventing, keep improving on the 20% left over at 80% each time, you will never get to 100%. However, after three iterations, you will get to over 99%, which is a solid solution. Perfection is only obtainable if you have only one stakeholder, the stakeholder never changes their mind, and the world around us never changes. We know this state is not obtainable. So, put perfection on the shelf and support your team to drive an awesome solution using the LAD/D hyper-agile process.

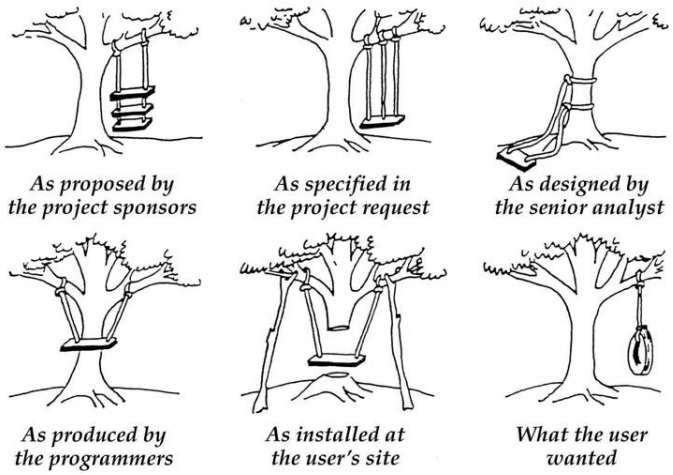
I wrote my first line of code in high school in 1976. The program I wrote was to add, subtract, multiply, or divide two numbers. Not sure exactly how many lines of code I wrote to make this app work, but it was probably around 20 lines. It took several hours to design, code, and test the app until it worked correctly; I was the Citizen Developer.

Let's just say for a minute I was not the Citizen Developer for this calculator app, but instead I was the person with the business problem, and I needed a team to develop the calculator. First I would need a budget, a steering committee to approve, assign a project manager, create a project plan, assemble a team of business analyst and system designers, create a use case, develop test plans/scripts, create app specs, schedule a software development team, ready the server for development, get a Database Architect to create the DB, setup security, perform system testing, perform user testing, log the errors, developer makes the updates, retest, rinse, repeat, approve the app for production, have the system admin launch the app, and implement change management. Abracadabra! Now you have a useable app. So, tell me again why is it so expensive to develop software apps? I guess you can say this has been a question of mine since I was introduced to this formal software development process many years ago.

Our hyper-agile process of LAD/D and the integration with ReAccess (with the PowerLine middleware API) give us the tools and services we need to streamline this solutioning. The ReAccess Azure cloud services that give us this opportunity include: Microsoft's Modern Security management, Cosmos Database configuration, Import/Export services, built in analytics using Power BI, Release Management, Backup, Data Encryption at Rest, Client Server Configuration & Management, Audit Transaction Logs, and Updates to New Releases from Microsoft & ReAccess.

Confirm Expectations as Early as Possible

I can remember my very first tech training conference at my first job out of college. I was a COBOL developer and attended a one-day analysis and design training hosted by IBM. In one of the sessions, the speaker handed out the following diagram.



When I saw this diagram, it was eye opening. Ever since this day I have pursued the ability to understand how this process can be more efficient and effective. Now with the creation of LAD/D and ReAccess rapid prototyping, we can communicate directly with the user, build the prototype app with them in real time, allow the user to "touch and feel" the app, then update it with the user to get it operational. With ReAccess rapid prototyping and our hyper-agile LAD/D process, this can be done in days, not months as with traditional development practices. Imagine building a tire swing prototype for the user as the user worked with you to describe the idea, watch you attach the rope to the tree and tire, then test the swing – that would be efficient!

Solution Configuration becomes Natural to the Business

Now that we understand the philosophy behind the EVM, we can move to the configuration stage. ReAccess is a SaaS platform that brings an app to life through our solution configuration module. The ReAccess configurator is simply a group of tables that capture the business rules and formatting for the application solution (Isn't that all apps are anyways, just a lot of formatted business rules imbedded in software code?).

When configuring the local university Sports Athlete Performance solution with ReAccess, it allowed us to quickly take the information we gathered in the LAD/D session and move it directly to formatting the configuration tables. There are two types of configuration tables – one for Entities and one for Static DDL's. There is also a Dynamic DDL creator table which allows us to use the Entity Attributes (fields) to create them. The Dynamic DDL's are key in creating the relationships within the app.

We open the configuration tables file on a large monitor so we can all see the formatting process. ReAccess is then set up to create the Entity tables, add the static DDL tables, and load the Attributes. When we finish this process, we can create the Dynamic DDL's based on the required relationships.

As we enter the Attributes, we establish business/edit rules for each field. You can set some of these rules on the database side and some at the field level. Some options for Attribute rules are Numeric, Alphanumeric, Email, Phone Number, Required, Search, Editable, and others.

We finished all configuration formatting and saved the tables. Once we had the file saved, we launched ReAccess and from the Home screen we selected the new configuration file to load into ReAccess. This triggered ReAccess to connect to our PowerLine API which connected to Azure and created the Cosmos DB along with setting up security through Azure Active Directory, and setup the app on the cloud server.

The app came alive on the screen (all from about two hours of work), we had a working Azure cloud application where we could add data, search on the data, select and edit the data, add multiple users, and connect Power BI in order to perform the analytics required for the app owner.

Lance turned to me and said, "that would have taken a team about 6-months to do on a typical development platform." I just looked at him and smiled.

Join the Rapid Design and Solution Movement

For more information on ReAccess Rapid Prototyping or LAD/D you can contact us at: