

Vanilla ERP Evolved

Application of Open Source Compiere ERP as an Agile Business System Development Platform

Allan Regenbaum, Director Emerging Technologies, SolTech Inc.

----- 000 -----

With over 11 years of building custom software and business solutions for clients, we see some fundamental changes. Skilled developers are in short supply, as are funds and risk tolerance for innovative projects. We need to find ways to deliver more functionality, with less risk, faster. We need to keep top engineers solving new problems rather than re-inventing commodity software.

In Compiere we find a powerful business software development platform with massive functionality that reduces time to value, and dramatically removes the need for rebuild of commodity software, freeing development dollars for top engineers to solve specific problems fast. Compiere is a best of breed development platform for business applications that has been designed from the ground up to leverage the best of an iterative, agile deployment approach.

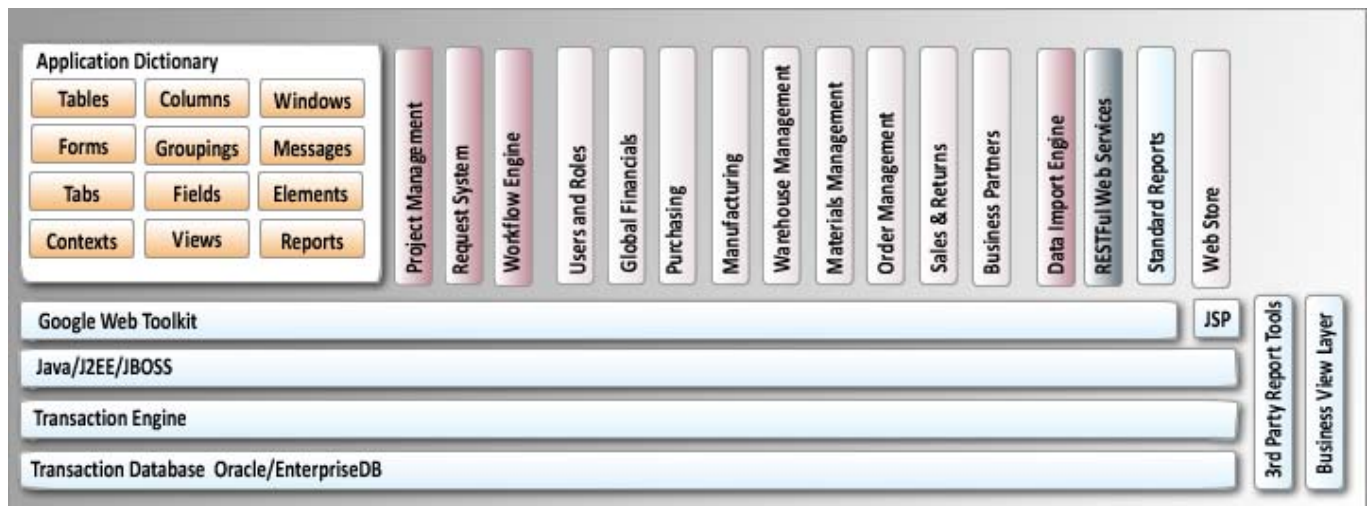


Figure 1 - Compiere Application Development Platform

Having an integrated, organization-wide ERP software solution can help leverage and unlock key information that has become so necessary in order to compete. First generation ERP systems are one of the key drivers that shaped the efficiency of American industry into the 21st century. ERP systems, enabled access to buried information resources that fueled efficiency,

performance and corporate profits. However, it is important to note that most first generation ERP systems today were designed 20 to 25 years ago. The technology and commerce landscape has undergone such radical change as to render these outdated systems, for all practical purposes obsolete and unable to provide their owners with an ability to compete in a modern, interconnected business environment.

Until recently, the installation of ERP systems often required that the host company completely change its way of doing business. It required running parallel software for months to assure a successful transfer of data, often for a multi-year period. TCO for legacy thick client/server implementations also suffered as manpower and equipment required to deliver the required reliability and uptime, became prohibitively expensive. Modern thin client web deployments, coupled with data integration tools and web services interfaces offer massive cost savings **Error! Bookmark not defined.** Businesses with legacy ERP will need to re-examine the costs of maintaining these legacy systems in light of new, simpler technology.

Every modern manager has learned that that maintaining a competitive edge requires an evolutionary ability to manage and thrive with day-to-day change. Business has changed. The way people communicate has changed. Virtually every legacy ERP including historical leaders such as JD Edwards, Lawson, SAP, and others, were created 10 to 20 years ago, or more. In order to promote market longevity these systems have become “internet enabled”, through multiple layers of patchwork, but in reality this enablement often is nothing more than backend screen scrapers. A fully web enabled system, able to interface to all of the modern computing devices, and web services is but a dream for many legacy ERP players.

All of the current, first generation ERP vendors are racing to convert their aging offerings to the new, object-oriented, Internet-based, thin client model--a task that is technically daunting for these vendors whose platforms and source code is, fundamentally, unsuited to the Web, nor do they have the ability to make use of the native functionality of the Web browser. This often requires the creation of a completely new product. This leaves customers with big decisions to make.

The only systems able to move at the current speed of web evolution, are the emerging class of modern, commercial open source ERP systems that have been created using the best functionality gleaned from the legacy world and evolved using the power of community developments. These next generation systems are deployed on architectures and platforms that deliver the scalability, usability and functionality required for web driven commerce. Compiere is the acknowledged leader in world of commercial and GPL open source ERP.

The dramatic cost, and time-to-deploy benefits of these new low cost ERP systems ensures that smaller companies, previously shut out of the ERP space due to the requirements of time and resources, are now able to rapidly implement competing end-to-end ERP systems that give

them competitive capabilities against companies with far greater resources. Older, larger implementations are the proverbial Titanic. When they realize the extent of the threat created by these new agile systems, it may well be too late. Commercial open source ERP is the most significant new internet-age tool available to level the playing field.

Besides being great platforms with modern programming structure and architecture, lightweight ERP systems also lend themselves to the best of modern software development and system deployment. Systems like Compiere have been created specifically to leverage modern agile deployment methods.

Proponents of agile deployment methodologies acknowledge that many legacy ERP implementations failed not because of the software, or the knowledge of the implementation team or the desire of the business community, but rather because the implementation used the 'big bang' approach where nothing is released for review or comment until the complete system is implemented.

According to the Agile Manifesto, a document describing the basic principles of Agile software development, **“Our highest priority is to satisfy the customer through early and continuous delivery, of valuable software.”** So we consider... can we deploy ERP using the same principles that have reshaped how software itself is delivered ?

In his March 2009 Blog, Vjekoslav Babic, a Microsoft ERP expert, outlined a 5 step guide on how to map agile process to the ERP deployment process. The steps in the blog offer suggestions on minimizing and overcoming the inherent risk in deploying the Microsoft Dynamics NAV and other legacy ERP systems.

Considering that even "modern" systems from established players still dissuade innovation and change, further highlights the extent to which Compiere is ahead of the curve as an ERP that is designed around the agile process.

“Deploy vanilla ERP first: *Most customers suffer from “the old software syndrome”, that says whatever you bring to the table will be judged from the perspective of what the old incumbent software could do; if their new software can’t offer at least that, it won’t be received well. Naturally, the customer wants their new software to be better, but this is often based upon perceived needs and not real needs. With vanilla deployment the theory is to first deploy it, and make the client use it without much theory about what they are going to get. What they see is what they get. Give the customer a chance to use the vanilla ERP in daily operations—they will be much more qualified to tell you what you need to customize, and you’ll be much surer that they are actually right.* “ - Babic

While there is much to be gained from distracting the client from preconceived notions about required functionality, from a Compiere perspective, much can be done to promote a positive outcome and help wean users off legacy systems and get acclimated to the Compiere system. This includes the ability to expose or hide fields on every screen by role. This simplifies usability and decreases user objections by exposing only those screens and fields that are specific to a specific role. Besides exposure of fields, the system can ease the migration burden by easily exposing certain users to certain simplified menus that are tailored by role. These changes can be made in real time without the need to interrupt user processing. This allows for a “somewhat vanilla” rollout that can be changed iteratively in a managed manner without causing system wide disruption. To further enhance the adoption of the new system, Compiere also provides for the creation of graphical workflows. Workflows can be created to echo the legacy process but in fact guide the user through the new Compiere screens.

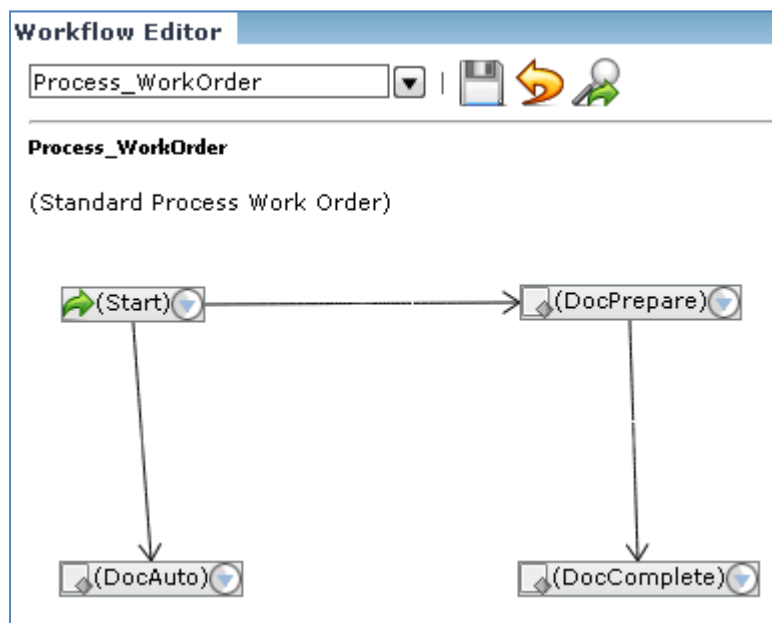


Figure 2 - Graphical Workflow Editor

“Deploy gradually: *instead of big-banging your project, deliver it function-by-function. It can add more value and add it more consistently over time, than traditional big-bang ERP projects.. Gradual deployments have an overwhelming benefit: they start generating value quickly. As soon as you deploy a function, if it was properly aligned with a project goal, it starts generating value for the client immediately. If you deploy for example 10 functions over 10-months period (1 function a month), then by the time all of them are in production, you have generated value. No matter how little value these functions generate individually, it is infinitely larger than zero, which is how much value is generated by a big-bang ERP before it goes live. “ - Babic*

Obviously a slower approach will spread the pain, but it isn’t without problems. The slower the phasing, the more temporary integration you have to build to the legacy system. Naturally, it costs money but modern ETL tools make the process quite do-able. You need to take these integrations carefully into account, and handle them properly. Even though they cost, this cost can be significantly lower than the cost value of the benefit you gain.

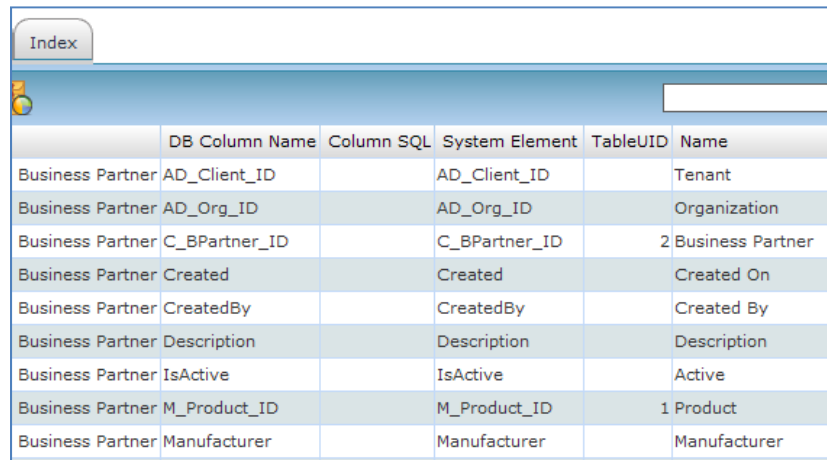
In terms of integration, Compiere provides a comprehensive ability to define scheduled data synchronization with external systems through a series of import tables. The import tables fire required triggers that ensure data consistency across the schema. The ability to define import profiles and then schedule execution of these loaders on a defined basis simplifies the ability to allow parallel deployment and a phased approach. Compiere also offers a comprehensive, patented web services API that facilitates interaction to external services without programming. The Compiere RESTFul WSI is driven directly from the data dictionary and is unprecedented in its simplicity and ability to deliver reduced time to value. Also to be noted is that any custom tables that are added via the Compiere application dictionary are automatically accessible by the web services interface.

Sequence	Name	Active	Column	Data Type
10	Vendor Product Code	<input checked="" type="checkbox"/>	VendorProductNo_Partner Product Key	String
20	Image URL	<input checked="" type="checkbox"/>	ImageURL_Image URL	String
30	Vendor	<input checked="" type="checkbox"/>	Manufacturer_Manufacturer	Constant
40	Product name	<input checked="" type="checkbox"/>	Name_Name	String
50	Description	<input checked="" type="checkbox"/>	Description_Description	String
60	Limit Price	<input checked="" type="checkbox"/>	PriceLimit_Limit Price	Number
70	Cost Price	<input checked="" type="checkbox"/>	PricePO_PO Price	Number
80	list price	<input checked="" type="checkbox"/>	PriceList_List price	Number
90	depth	<input checked="" type="checkbox"/>	ShelfDepth_Shelf Depth	Number
100	Width	<input checked="" type="checkbox"/>	ShelfWidth_Shelf Width	Number
110	Height	<input checked="" type="checkbox"/>	ShelfHeight_Shelf Height	Number

Figure 3 - Custom Import Loader

“Focus on value, and value alone: Before you customize, you might find out that a standard feature suffices, or that a simpler design will do, or that it is not necessary at all. Before you jump into defining requirements, you should define clear and measurable project goals. Once your goals are clear, your job is much easier. If a requirement doesn’t contribute to achieving a goal, directly or indirectly, then dump it. This is equally true about both standard features and customizations. You shouldn’t strive to implement a standard feature if it won’t contribute to achieving your goals. “ - Babic

There is a fine line between “nice to have” and “must have”. The prohibitive cost of system changes in legacy systems has distorted the analysis to the point that “must have” is being reshaped into “nice to have” and is being dumped , when in fact these requirements often form the basis of core competitive advantage.



	DB Column Name	Column SQL	System Element	TableUID	Name
Business Partner	AD_Client_ID		AD_Client_ID		Tenant
Business Partner	AD_Org_ID		AD_Org_ID		Organization
Business Partner	C_BPartner_ID		C_BPartner_ID	2	Business Partner
Business Partner	Created		Created		Created On
Business Partner	CreatedBy		CreatedBy		Created By
Business Partner	Description		Description		Description
Business Partner	IsActive		IsActive		Active
Business Partner	M_Product_ID		M_Product_ID	1	Product
Business Partner	Manufacturer		Manufacturer		Manufacturer

Figure 4 - Table Definition via browser GUI

Compiere’s ability to rapidly implement changes to fields, screens, processes and workflows through the user interface with little or no programming **restores the ability to innovate** without the threat of prohibitive, extended development costs.

Also to be noted, is that with Compiere’s ability to map callouts to each field change, and the ability to define process and workflow from the UI, it may well be that a function can be achieved by a slightly different configuration approach rather than discrete customizations or removing the requirement out of fear.

“Stay light with customizations: *if you want your solution to sustain and survive any long-term or high-impact change, keep it as close to standard as you can. It doesn’t mean you shouldn’t do customizations at all.” - Babic*

This is one item in the 5 point plan that is made redundant by Compiere. Off-the-shelf software, that exposes source for customization has a reputation that, “once a change is made, you own it”. The inherent inability of these systems to manage the change process continues to stifle innovation. Compiere is built for change, to the extent that all changes made by the UI to the application dictionary are automatically flagged as “user maintained”. All changes to core code can be manually flagged as “user maintained” . Compiere’s automated migration process ensures that when the system is upgraded, all changes that are “user maintained” are automatically re-applied after core code migration. The single largest roadblock to innovation on legacy ERP customers on legacy systems has been reduced to a trivial process with Compiere.

“Integrate where ERP is not absolutely necessary: *you and your customer must understand and define clear boundaries between ERP and everything else. Integrating ERP with external applications might be less costly, and bring more value, than customizing your ERP to the nth degree. Not everything belongs inside the ERP system, and the modern world of web services and ETL data integration now enables easy integration with discrete 3rd party systems, or other WEB deployed systems.”- Babic*

The Compiere platform is designed to facilitate an Agile ERP deployment. It should also be noted that due to its MVC architecture and the ability to adjust the application dictionary through the UI, Compiere can be used as the base launch platform for any generic business application whether that application will use core Compiere functionality or not.

In many situations with Legacy ERP, the standard response to lack of functionality is that “an external point solution would be required”. Compiere’s inherent ability to extend and customize via the application dictionary , has seen many clients use the platform as a RAD tool for adding new tables, forms, screens and reports, with the option of tying those into the underlying ERP schema, or not. The ability to call external classes on status change on any field of custom forms and windows, allows that linkage to take place without the need to re-invent the system. The recent addition of the dictionary driven WSI, to an extent repurposes Compiere as a custom web services generation tool

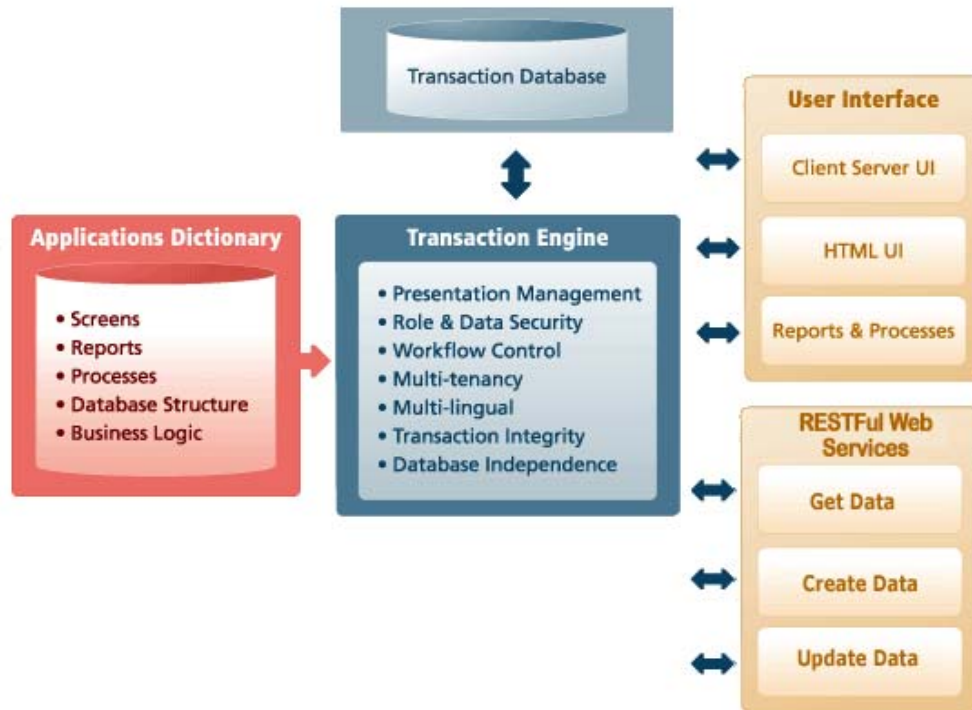


Figure 5 - External Interfaces

The “vanilla” Compiere platform, delivers out-of-the-box, all of the fundamental components of a business system from financials to warehousing and manufacturing. The user can choose to use, or not use, any of the core functionality. By considering Compiere as a launch point for app development, we get a large amount of value before the first line of code has been written.

SolTech, is a specialist custom software developer that uses the Compiere platform as a means to accelerate delivery of value to clients, while at the same time ensuring that its valuable programming resources are used to write high value, interesting software, rather than spending scarce financial resources on repetitive development of basic functionality. SolTech engineering practices apply the best of agile development methods to each project. The fact that Compiere is built from the ground up to suit an agile ERP approach makes it a powerful addition to SolTechs core offering.