ObjectBuilders

WHITE PAPER

Software Assembly

Assemble 100% of a Business Solution
Composite Application
Product Selection

WHITE PAPER

Software Assembly

© ObjectBuilders
20134 Valley Forge Circle
King of Prussia PA 19406
Phone (610) 783-7748 • Fax (610) 783-0389

Restricted Rights

Printed in the United States of America

This document is subject to change without notice, and ObjectBuilders does not warrant that the material contained in this document is error-free. If you find any problems with this document, please report them to ObjectBuilders in writing.

This document contains or may contain statements of future direction concerning possible functionality for ObjectBuilders' software products and technology. All functionality and software products will be available for license and shipment from ObjectBuilders only if and when generally commercially available. ObjectBuilders disclaims any express or implied commitment to deliver functionality or software unless or until actual shipment of the functionality or software occurs. The statements of possible future direction are for information purposes only and ObjectBuilders makes no express or implied commitments or representations concerning the timing and content of any future functionality or releases.

Document Purpose and Scope

This document provides an overview of Software Assembly. It is written for a business and technical audience that is familiar with application development. It is intended to provide the reader with a general understanding of Software Assembly. For additional information, please contact ObjectBuilders at 610-783-7748 or visit us on the Web at www.ObjectBuilders.com.

Introduction

Since the early days of computers, we have been making our everyday tasks easier and easier. In computing, the pinnacle for ease of use is defined as a Productivity Tool. We use Productivity Tools everyday in our work – word processing, spreadsheets and presentation software. Productivity Tools are also used in software development for reporting, database configuration, workflow and business rules – just to name a few. In fact, when designing a modern solution, the ability to configure (or assemble) the solution is a critical success factor.

In Software Development, there are many technologies and products that provide powerful flexible solutions to various functional areas. The best products are highly configurable and do not require the writing of code. There are Productivity Tools that allow the user to effortlessly configure the product's power while still tailoring it to a particular solution.

Software Assembly is focused on assembling 100% of a Business Solution without coding, code generation or compiling. Software Assembly can virtually eliminate the risk associated with the construction of a Business Solution, while dramatically reducing the risk in many other areas of the application lifecycle.

Organizations are currently using assembly in many areas of application development such as presentation (user interface), reporting, business rules, workflow, testing. What ObjectBuilders is advocating is an approach that will "max-out" the assembly ability with products that are currently available in order to achieve a 100% assembled Business Solution. The resulting Business Solution will be easy to construct, flexible, adaptable as well as easy to maintain. This is Software Assembly.

Software Assembly solves current business challenges. Software Assembly overcomes some typical challenges in application development:

- **Doing More with Less** – Usually a higher output of work is needed with less time and resources to do the work.

  Software Assembly simplifies complex software development tasks by reusing components and emphasizing configuration rather than coding. Simplifying software development means reduced time and lower effort to deliver functionality. Software Assembly is designed around the concept of reusing existing assets and eliminating repetitive tasks by using sophisticated tools that have much of this capability built-in. By using WYSIWYG drag and drop tools, Software Assembly also increases the output by eliminating throw-away click through demos and mockups. In many cases, working software can be created in the time it previously took to create a throw-away mockup.

- **Constant Changes** - Respond to changing requirements, conditions and regulations quickly and effectively.

Software Assembly provides the power in configuration rather than in coding. This means that real changes to software can be made by using drag and drop. Additionally, changes are typically isolated in a series of configurations, thereby reducing interdependencies and correspondingly the risk. Such architecture dramatically reduces the testing time needed to deploy functionality.

- **Disparate Information** - Need to access and leverage information across multiple systems and disparate data sources quickly and easily.

  Software Assembly is focused on connecting existing data sources, services and other assets without coding. This means that easy to use tools can be used to create applications that combine multiple data sources.

- **Security** - Provide highly secure solutions while still rapidly delivering on changing requirements.

  By using configuration instead of code to build an application, Software Assembly reduces vulnerabilities by using tried and tested components, rather than creating new ones.

  Software Assembly combines multiple data sources together with a layer of abstraction, isolating the application from the data sources and creating a built-in audit trail of access between the applications and the back end.

  Software built with Assembly tools is inherently more secure because the runtime engines that are used to reduce the development time also serve as a single, robust and restricted code base, thereby reducing vulnerabilities.

- **Flexibility** - Solutions need to be flexible, empowering the user to do more and adapting to changing business needs.

  By using tools and components, Software Assembly results in highly configurable solutions that allow users to make changes to business rules without needing to rely on IT to write code. Changes to business process routing, content structures, business rules, user interfaces, forms and more are configured using drag and drop tools.

- **Many Stakeholders** - Need to get buy-in from large numbers of stakeholders without slowing down the delivery cycle.

  Software Assembly can be used to rapidly create prototypes that are closer to the real application than a typical paper based static model. The ability to see, touch and feel an application almost always results in better, more concise stakeholder feedback. The prototypes are not throwaways, and any functionality created for the prototype can be leveraged in the deployed solution.

The Role of Assembly

Assembly already plays an important role in most parts of an application lifecycle. Today's solutions are made up of a multitude of different products that each solves a need of the entire Business Solution. The resulting Business Solution is a Composite Application. When selecting products for a Business Solution, the driving factors tend to be:

- Powerful and feature rich
- Flexible and adaptable
- Easy to leverage
- Functionality that can be configured with Productivity Tools

For example, there are many choices for the functional area of report writers: *IBM Cognos*®, *SAP Crystal Reports*®, *Oracle Hyperion*®. And for each functional area of a Business Solution, there are a number of products lining up to fill the role. These products are the result of significant investments in R&D and are proven solutions that have been deployed successfully over and over again.

Most organizations have a variety of products and solutions. More often, end users usually have to search multiple solutions or different vendor products to get what they need. Or, when looking to create new solutions, business users often ask why they can't combine the existing good functionality with the best from the new products.

The problem becomes getting all of these products to speak and play well together. This is where coding or code generation comes into play. But with each line of code, the risk increases, decreasing the value of the selected product.

Software Assembly uses a Composite Application Builder to make existing products speak and play well together. A Composite Application Builder allows existing and new products to be published to a Productivity Tool where the entire end Business Solution is 100% assembled - effortlessly!

Products, frameworks and services (or "components") often come "ready to code" but not "ready to use". A component, for example, that provides a credit score that requires many lines of code to initialize and then even more lines of code to get the actual score, is "ready to code". However, a component that provides a credit score that requires one line of code to get the credit score is "ready to use". This is often recognized in the difference between an older Application Programming Interface (API) style and a modern Web Service style.

The components that are ready to use will be easier to consume and therefore assemble.

With Software Assembly, we have an additional driving factor in our product selection:

- Consumable – Does the product provide components and services that can be assembled? If not, can the product's functionality be easily wrapped into a component or service that can be assembled?

Most organizations are already assembling a large part of the Business Solution. However, there is still a considerable amount of coding – and therefore risk – in producing the complete Business Solution. Today it is possible to assemble 100% of a Business Solution utilizing consumable products and a Composite Application Builder. Code is now limited to product vendors, custom services and integration (which is just a onetime effort to create a consumable).

The Benefits to Software Assembly

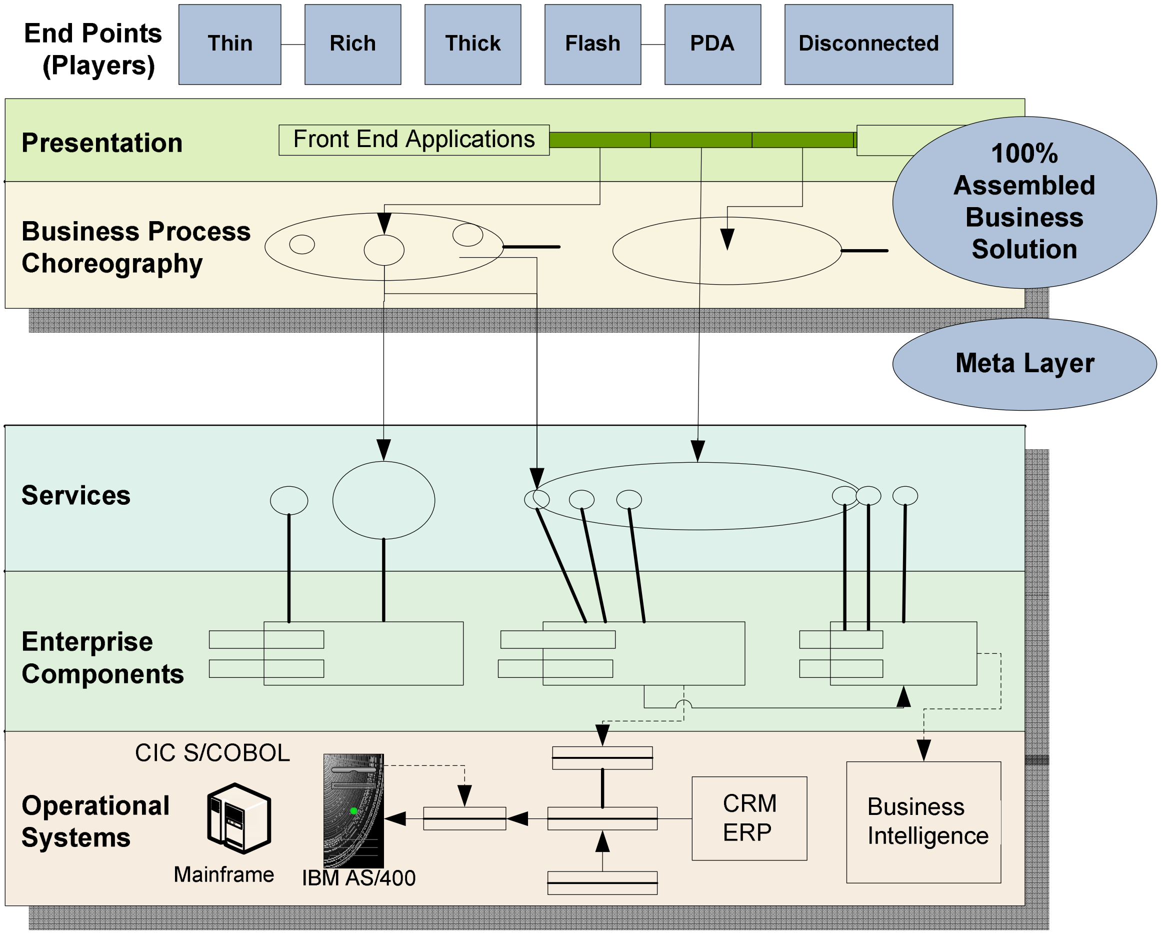
The benefits of using Productivity Tools are well known: Reduced Cost; Reduced Time; Reduced Risk; and Reduced Skill Set. However, when assembling 100% of a Business Solution, we can achieve so much more.

Benefit	Description
Eliminate Risk	Risk is virtually eliminated from the construction of a business solution.
Reduce Risk	Risk in many other areas is greatly reduced: Specification, Performance, Scalability, Security, Certifications and more.
Simplified and Better Specifications	There is already an understanding of the parts, applications, Productivity Tools and technology that will be used to create a business solution.
Easier and Higher Quality Methodologies	Assembling a solution is analogous to a package solution methodology. These methodologies tend to be simpler, more agile and of higher quality than a custom solution methodology.
Scalability	A specification can be divided into Jobs (examples include a report, a window, a workflow) and given to many developers or assemblers at once.
Consistent Quality	By utilizing time proven techniques from industrial manufacturing, we can achieve consistent quality over a large number of developers and assemblers.

While the art of coding is not going away, it can be eliminated when it comes to creating Business Solutions. And the key point is there is no practical limitation to the type of Business Solutions that can be assembled.

Principles of Software Assembly

The number one rule of Software Assembly is to assemble 100% of the Business Solutions – code only when creating new reusable components. The graphic below depicts the layers that should be assembled and the meta layer that must be present.



No coding, code generation or compiling

Code should only be used to create new products, components and services that are published or available for assembly or configuration by the Productivity Tools. Coding is a one time practice to create new functionality. There should be no code, code generation or compiling in order to create Business Solutions.

Software Assembly helps ensure that the full advantages of the infrastructure products and components can be achieved and their value maximized.

Coding on top of these products or trying to force products to do something they were not made for can result in a liability. If you are hard coding on top of your assembled infrastructure, the Productivity Tools and products embedded in the application will be significantly masked, minimized and marginalized, while risk will significantly increase.

Encapsulate complexity into tools and parts

Encapsulating complexity into tools and parts means that repetitive software tasks should be isolated and reused instead of recreated on every system. Encapsulating complexity into parts has been advocated by many strategies starting with the advent of Object Oriented Programming. Software Assembly extends the concept of encapsulating functionality into services and components further into the concept of tools. Much like an automobile assembly robot encapsulates functionality that is useful in building a car, a configuration tool for BPM may make the process of constructing workflow driven applications much easier.

Encapsulating the functionality means that high quality, highly configurable tools can be used to rapidly deliver business value. A variety of tools encapsulate the complexity involved in common business application tasks such as User Interfaces, Reports, Workflow, etc. Assembled solutions are therefore not only constructed faster, but they are typically of a higher quality and more secure. As an example, consider modern automobile manufacturing instead of building a car directly from a set of raw materials, a vast array of tried and tested components are brought together to deliver the end product. Much of the assembly process is often completed using automated assembly lines, tools and robots. This results in a higher quality output that can be delivered consistently in a shorter amount of time.

Originally, manufacturing applied to software was just a dream. Coding at any level makes manufacturing impossible. Imagine an assembly line being stopped because two parts are incompatible! With assembly, we have base components and services, but we also have completely assembled templates and even solutions. Consider creating a new presentation from a template or even from a finished presentation, modifying it for your specific need. Software, unlike hardware does not have to be physically manufactured, so after the design is described or assembled, it is ready to use.

Assembly Components

One of the fundamental principles of industrial manufacturing is the use of components that can be assembled together without additional customization. In software development, that means without additional code – no wonder the software industrial revolution never went anywhere!

Imagine the incredible problems that would arise if a car was moving down the assembly line and the fuel injection did not fit. To a software developer this would be no problem, just another bug in a long list of expected bugs. This would not be tolerated in manufacturing. If we want to achieve the results of industrial manufacturing, consistent high quality, scalability, less skilled resources and speed to market, we need to make sure that the components can be assembled together without additional coding.

Minimize process to maximize consumables

The purpose of process in the software development lifecycle is to produce a high quality output that meets the business objectives. Such process inherently contains overhead, some of which is necessary, and some of which can be streamlined. The principle of minimizing process to maximize consumables means that strategies should be used to eliminate unnecessary overhead in a process to maximize those things that can actually be used and are not "throw-aways" at the end.

In Software Assembly, instead of throw away mockups or demos, the solution is assembled directly so these pieces can be used going forward. Use Cases are enhanced to create Business Scenarios and test cases.

Templates are used for collection with Subject Matter Experts. The application is visually demonstrated instead of being described in text.

As the tools and components encapsulate more of the difficulties, they also remove the need for much of the process overhead. How often do we make changes to a presentation without regression testing the entire presentation? If the presentation depended on the creation of custom code, this process would be much different.

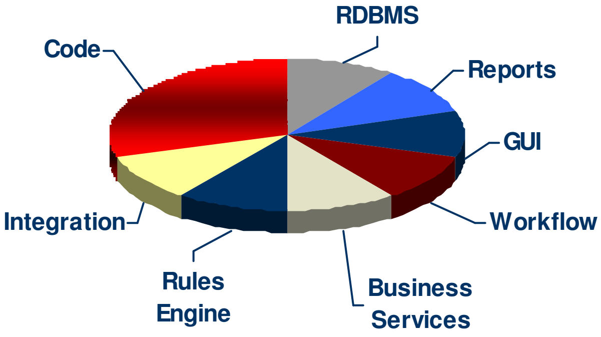
Decentralize and Decouple

Software Assembly changes the make-up of an application to a series of visual configurations rather than thousands of lines of code. Each configuration can be maintained in a decentralized, separate fashion. Data sources are also decoupled from the application (isolating it from change), while a layer of abstraction is provided to interface with these existing assets. A key trait of an assembled solution is the fact that it uses many existing assets rather than combining them together. This decentralized strategy distributes load across many separate systems while eliminating any single point of failure, further increasing not only the scalability but also the built-in fault tolerance of an application.

Composite Application

Today's Business Solutions are Composite Applications and are made up of many different products. When it comes to making these products play nice together, we have traditionally used code. A Composite Application Builder allows disparate products and technology to play nice together in addition to being 100% assembled.

The chart below depicts a typical solution that is made up of different types of products and services. The products available on the market today typically solve all the functional areas of the application. The only thing remaining is to pull it all together into a finished composite solution. As the chart shows, this is where code comes into play.



A Composite Application Builder addresses and replaces this need to code with a Productivity Tool.

Software Assembly Product Selection

The table below lists the functional areas of the solution and the recommended product to achieve 100% assembly of the Business Solution in Software Development today. This is Software Assembly. Since Software Assembly decouples products from the Business Solution, substitutions of products in each category can easily be made. The different tables demonstrate only a few of the almost endless possible combinations.

Functional Area	Product
Transactional Data Storage	SQL Server®
Enterprise Content Management (ECM)	IBM® FileNet®
Business Process Management (BPM)	IBM FileNet
Business Rules (BRMS)	IBM® ILOG®
Reporting	SSRS
eForms	Adobe® LiveCycle®
Composite Application	ObjectBuilders LiveApp Player Suite
Records Management	IBM FileNet

Functional Area	Product
Transactional Data Storage	SQL Server
Business Process Management (BPM)	ObjectBuilders LiveApp Player Suite
Business Rules (BRMS)	ObjectBuilders LiveApp Player Suite
Reporting	SAP Crystal Reports
Composite Application	ObjectBuilders LiveApp Player Suite

Functional Area	Product
Transactional Data Storage	Oracle®
Enterprise Content Management (ECM)	EMC Documentum®
Business Process Management (BPM)	IBM FileNet
Business Rules (BRMS)	ILOG
Reporting	SSRS
Composite Application	ObjectBuilders LiveApp Player Suite
GIS	Google® Maps®

Most companies are already utilizing products like those listed in the above tables. Your organization is probably only a couple of steps away from achieving 100% assembly – that is, achieving Software Assembly.

Business Solutions that are 100% assembled – without code – are being successfully delivered today. Not only small departmental solutions are being assembled, but also large, highly secure, high volume enterprise solutions.

This approach is not risky or new. It is a natural progression enabled by advances in products and tools from the world's largest vendors. The products and tools used in Software Assembly have been widely accepted and validated.