Dynamic Composite Applications and Enterprise Mashups with Convertigo.

A Convertigo White Paper,
By Eric Samson, Product Marketing Manager, July 2010
Summary

Exec Summary

The New User-centric information Paradigm

- Competitive business pressures drive need for increased productivity
- A new, elegant solution to meet business challenges
- Too many applications, too much data
- Too many technologies
- The complexity of desktop environments

Towards Dynamic Enterprise Composite Applications

- C-EMS: a comprehensive platform for Enterprise Composite Applications
- XML and SQL integration
- API-less legacy mainframe and Web applications
- Mashup Sequencer
- Yet Another Scraping Tool?

Enterprise Composite Application development

- Convertigo Studio
- Mashup Composer

Enterprise Composite Application deployment

- Convertigo Administration
- Convertigo Cloud

C-EMS use cases

- Smartphone-ization
- Consolidate partial business views
- New desktop
- Leverage existing applications

Conclusion
Exec Summary

In all industries, and especially in Financial Services and Insurance, business users are required to interact with endlessly sprawling data sources and applications. This trend is reinforced by an accelerating number of mergers and acquisitions in the industry. Composite applications, or mashups, are a way to reintroduce user efficiency to the heart of business applications, by enabling agile design and deployment of customized applications created by simply picking and connecting together visual components, known as widgets.

In this paper, we will discuss industry trends and challenges and introduce the Convertigo Enterprise Mashup platform and how it enables the promise of dynamic composite applications connected to enterprise legacy assets, while preserving IT governance rules.
The new user-centric information paradigm

The explosion of the consumer Web has forever changed how people think of computing and software and how applications need to be built. This shift – which could be described as moving from a system-centric to user-centric orientation - is arguably one of the most important changes to computing since the advent of the personal computer. Users benefit from information that has been freed from proprietary databases and transaction systems, and then aggregated and presented in ways that map to users’ needs in terms of information viewing and interaction.

More and more, IT departments are being asked to present data in business applications with the same look and feel, robust graphics, video, multimedia and dynamic content found online today. One of the most valuable things IT organizations can do to help the business units succeed is to provide easier access to business-critical information and display it in comprehensive, intuitive way.

Until now, financial services companies have been trapped in the old paradigm – where system-centric design rules the IT landscape. Resulting from years of major industry consolidation, many financial services firms are burdened with legacy systems inherited from various acquisitions. Additionally, core data is highly transactional, complex, stored in multiple silos, and more often processed in batch mode rather than in real-time.

Competitive business pressures drive need for increased productivity

More so than ever before, these companies feel the pressure of the information economy. In order to obtain or maintain market leadership, they must master the management, integration and delivery of the vast amounts of information upon which their businesses rely. Despite the companies’ desire to keep up with the pace of change and migrate all of their internal applications to be more internet like, the investment in this kind of change is often extremely prohibitive. However, those companies who do invest in these changes will then be poised to define the new efficiencies in customer service and in the streamlining of operations.

Financial Services institutions rely heavily on customer reference data – mission critical information about customers and their many past interactions with various touch-points throughout the organization. Because reference data is stored and managed in many places, whether it is across the front-office, back office, business units or product lines, front-line employees, like call center representatives, must use multiple business applications to gain the information required to assist with customer interactions.

This inability to access a 360-degree view of vital customer information has a negative impact on operational efficiencies and customer satisfaction. Lengthy delays occur as agents have to access multiple systems and screens to find the relevant data. Calls are often forwarded to different departments, and customers are frustrated because they can’t get their issues resolved quickly. Given today’s highly competitive markets for financial services, customers who are unhappy with the level of service from their current institution can easily find alternatives.
Too many applications, too much data

Applications are typically organized by business function, but employees are often required to establish communications with customers that transcend functional boundaries. Consequently, employees often must use an ever-increasing number of applications for their day-to-day operations. For instance, bank employees may need to connect to dedicated applications for loan management, checking account management, portfolio management, etc. And, as the lines blur between the banking and insurance industries, banks may need dedicated insurance applications like life insurance, property insurance, car insurance management, and so on.

Banks are working to streamline their portfolio of applications, but it often takes a significant amount of time (multiple quarters and years), and M&A processes often happen much faster than rationalization of these systems and processes. As a result, the number of applications at a given institution is continuously increasing.

Not only are employees confronted with too many applications, but each application often displays too much data. This is because applications are designed to be used by multiple categories of users, and have few customization capabilities. Users often must filter data displayed in complex and overloaded forms and many times must navigate between multiple screens before reaching the relevant part of the application.

Too many technologies

For technical and historical reasons, all of these applications have been developed using various technologies, infrastructures, programming languages and frameworks, ranging from centralized mainframe to distributed Web and SOA, for the most recent ones. In the last two decades, we have seen numerous application programming paradigms emerge, including:

- Client-server (Windows clients talking to Unix based RDBMS through 4GL integrated environments),
- Distributed objects computing (with the CORBA standard specified by OMG),
Early Web development (CGI scripts),

Simple Java Web development (with MVC-oriented JSP / Servlets frameworks like Struts),

Heavyweight Java managed environments (J2E application servers, EJB specified by the JCP),

Lightweight Java managed environments (Spring),

Service-oriented architecture (with comprehensive standards like SOAP specified by OASIS),

Business Process Management with tools like workflows and XML, SOAP-based languages like BPEL and modeling notation like BPMN,

And more recently the advent of Web-oriented architectures (WOA), based on the “supposedly simpler” principles of the Web.

This evolution is nowhere near its completion. Emerging development trends include Domain Specific Languages (DSL), highly parallel computing (with new programming languages like Scala and new programming models like multi-agents systems and actor languages).

This continuous change is not limited to application programming models. Methodologies and user interfaces (Ajax, RIA, 3D worlds, augmented reality, touch surfaces, movement detection) are also evolving. Even well-known domains like security and databases are quickly changing, with physical recognition and RDBMS superiority being questioned by new highly scalable grid-based “NoSQL” engines designed for very large Web applications like Amazon or Google.

The same applies to deployment models, where traditional on-premises physical servers are now combined with virtualized environments and external Cloud-based solutions with an increasing number of heterogeneous client technologies, including desktop computers, laptops, touch pads, mobile devices and smartphones.

New generations of technologies almost never replace previous ones. They always complement what existed before, adding new layers and new concepts to take into account in real-world information systems environments. Not only are applications sprawling, but heterogeneity is growing even faster.

The complexity of desktop environments

This is why, today, a desktop almost always hosts native applications, terminal emulations and Web-based applications (and sometimes even a mix of them as Web browsers support native components).

Any non-trivial business task involves the following actions:

1. Find an application on the desktop or on the net
2. Launch this application
3. Connect to the application using user credentials
4. Navigate through menus, screens and forms
5. Input some data
6. Perform some queries
7. Retrieve, read, print some results
8. Analyze and interpret raw results, by eliminating redundant data, filtering relevant data, transforming and normalizing heterogeneous formats
9. Repeat steps 1 to 8 as many times as necessary with different applications
10. Mentally aggregate the different partial results in order to compile a possibly complete vision of the required data.

Most of these steps are time consuming and some of them (8 and 10) can lead to errors and therefore to potentially bad business decisions. Navigation within applications is dependent on the technology used to develop these applications. A Web application has a totally different user interface from a legacy mainframe green screen application. This means users need to be trained on different user interfaces.

If users keep all applications opened, they can save connection time (steps 1 to 3) but then they have to switch between dozens of windows, which is often not much better in terms of both ease of use and efficiency (memory saturation leads to disk swapping and performance issues).

Sometimes, for efficiency reasons, even step 3 is shortened by bypassing security principles: the same passwords are often used for all applications or no password at all is required.

Step 10 indicates that users are often the real “integration layer” between applications. Technologies like data and application integration (EAI, EI and so on) are used within organizations, but they normally address integrations issues at the infrastructure level, between systems. They do not impact desktop applications and users are often required to manage this by themselves – leading to partial and sometimes incorrect customer information.
In summary:

- Users must simultaneously manipulate too many applications in order to gather necessary data. This is because applications are fulfilling given business functions, while users must transcend these boundaries to get the information they require.
- Each application has a different user interface requiring significant training.
- Each application presents too much data, and users have to filter relevant data by themselves. This is because applications are generally designed to be used by different categories of users, having different requirements, leading to screens overloaded with the union of all the data needed for all users.
- Users have to compile data from multiple applications by themselves.
- Each application uses significant CPU and memory leading to poor response times.

Towards Dynamic Enterprise Composite Applications

Composite applications, or mashups, are a way to dynamically assemble existing visual components (usually known as widgets) and services together in order to quickly deliver new business functionality. “Power users” can then combine widgets into mashups that can be later customized at runtime. It is then possible to deliver tailored composite applications to specific categories of users. Users of a given type will get all the data they
need within a single application and be provided with just the data they need.

Composite applications are similar to portals but are more dynamic in nature. They can be real transactional production applications, and the widgets can be made to interact by exchanging data and triggering functions.

Composite applications are already known by consumers – the concept of mixing widgets together was introduced with other Web 2.0 technologies in products such as Yahoo! Pipes or iGoogle.

An Enterprise Composite Application is simply a mashup with at least one enterprise widget or service connected to an enterprise resource. Enterprise resources are:

- Databases,
- Web Services,
- Packaged applications (e.g., ERP, PLM, SCM, CRM, SFA, etc.),
- Traditional Web applications (intranet, extranet, internet) without Web services,
- Legacy “green screen” applications running on mainframes (TN3270) or AS400 (5250).

**C-EMS: a comprehensive platform for Enterprise Composite Applications**

Convertigo Enterprise Mashup Server is a complete and modular platform for the design, development and deployment of Composite Applications connected to Enterprise Resources:

- The Legacy Integrator generates Web services and widgets from legacy mainframe applications (3270, 5250, VT, etc.)
- The Web Integrator generates Web services and widgets from legacy Web applications (3270, 5250, VT, etc.)
- The Legacy Publisher dynamically generates fully functional HTML, Ajax or even Flex applications from existing legacy mainframe applications.
- The Web Clipper dynamically captures, transforms and aggregates regions of existing legacy Web applications, including their user interface, and renders them into modern visual components that can be later recombined into composite rich internet applications.
- The XML and SQL integrator allows combining results of Web services calls (either SOAP or REST) and SQL queries into with other sources into widgets and composite Web services. It can also be used to store mashup data into intermediate databases.
The Sequencer is able to aggregate and orchestrate several data sources and processes at the server-side. It is also used to format XML output of mashups according to company business models (compliance with already defined XSD), to store mashup data into intermediate databases (through the SQL integrator module).

The Studio is a visual integration tool, used to produce and deploy Convertigo Web services and widgets.

The Mashup Composer is a Web-based tool to visually pick and mix widgets into mashups and composite applications and then deploy them.

The Admin console is a Web-based interface used to manage and monitor the Convertigo platform and the deployed Web services and widgets.

XML and SQL integration

Web services and relational databases are common resources that are often needed to integrate into composite applications. Using this module, Convertigo users can turn existing Web services and tables into new Web services and widgets.

Integrating Web services into composite applications is not something unique; all mashup platforms can do this. Integrating tables is not unique either. The SQL / XML integrator module is particularly helpful when used in conjunction with the Convertigo sequencer as it allows composing server-side mashups where, for instance, legacy information is enriched with additional information coming from modern data sources like Web services and tables.

Convertigo SQL integrator is not an object-relational mapping tool and developers must supply well-formed SQL statements. It is not a Data Services platform either, even if it can be used to quickly generate Web
services from existing relational data.

Another usage of the SQL integrator is to write collected data to a local storage, in order to reuse it efficiently (e.g. without submitting a subsequent request to the original data source). This powerful technique is known as data harvesting. When several data sources are consolidated into such a local database it becomes a data mashup. SQL Integrator is not an ETL at all and has not been designed for that purpose, rather it is a convenient solution when persisting data collected by mashups is required.

API-less legacy mainframe and Web applications

As we have seen previously, a real Information System environment includes applications developed with different architectures, language and frameworks. Most recent applications expose structured data models (like SQL result sets and XML for instance) through well-defined APIs (like SQL and SOAP for instance) and are easy to integrate. Legacy applications are a different story.

Most legacy applications do not expose any structured data and APIs. They were not designed with integration in mind. Many software architects and project leaders, expert in new technologies, are unaware of this shortcoming and expect all enterprise resources to be service-enabled. The reality is almost the opposite, most legacy mainframe and Web applications cannot be directly integrated. There are a number of technical and financial reasons why this will not change soon:

› Companies are not willing to spend money to rewrite systems that are still working quite well – they prefer to invest in new business functionality

› Changing anything on mainframes is difficult and costly

› System administrators are reluctant to deploy new services on mainframes

› There is significant risk that rewritten mainframe applications will not work more efficiently on new distributed systems

› Sometimes legacy Web applications are external and cannot be modified

Occasionally, the database on a mainframe or underlying a Web application can be accessed. Even when it is possible from a technical point of view (an extremely rare case on mainframes), it is generally not a good practice, as it bypasses all of the business logic defined in the presentation and business layers.

C-EMS allows access to legacy mainframe and Web applications when they have no APIs and only expose unstructured data in screens. Convertigo can generate SOAP or REST Web services or even widgets from these screens. By accessing the application from its most external interface, e.g. the user interface, Convertigo operates via existing business logic across all its layers.
Convertigo is able to manipulate existing screens, even when complex navigation is required prior to accessing these screens. Convertigo interacts with existing systems exactly as users would. Everything that can be done by real users can be done through Convertigo. Convertigo is not limited to read operations, and therefore can input, validate and submit data just as users would.

Convertigo Web Integrator and Convertigo Legacy Integrator can create Web Services and widgets from existing Legacy assets.

Convertigo Web Clipper can not only capture data and process layers but also elements of the presentation layer of a legacy Web application. It is a very efficient solution when composing new applications keeping most of the original user interface look and feel.

Convertigo Legacy Publisher can automate the capture of all screens and navigation elements of a legacy mainframe or AS400 application. This feature relies on hierarchical capture rules and customizable generation rules. The final Web rendering can be adapted to particular needs using style sheets (CSS). Convertigo Legacy Publisher will add new output formats, like Flex and HTML 5 for instance, in a near future. It is still possible to manually adapt some of the capture and generation rules, following the 80/20 pattern: 80% of screens are automatically converted, allowing developers to spend more time on the most interesting screens. Using Convertigo Legacy Publisher, development teams can save significant time when modernizing existing mainframe applications.

Accessing both legacy mainframe and Web applications in addition to Web services and RDBMS is a unique differentiator of Convertigo Enterprise Mashup Server.

**Mashup Sequencer**

In the Convertigo terminology, the connections, navigations, extractions and generations required to dynamically instantiate a Web service or a widget from business functionality is called a “transaction”.

---

**fig. 4 - Convertigo reuse existing business logic**

User Interface (Surface controls)
Applications (business services)
Database (stored procedures)
Business Logic
A Convertigo “sequence” is an ordered set of calls to multiple transactions. A sequence is typically used to integrate results from multiple (possibly heterogeneous) transactions into a single result set. It is important to note that this integration is done at the server side; this is why this feature is often called “server-side mashup”. The steps of a sequence are visually created in Convertigo Studio; they orchestrate the different transactions, a little bit like in a workflow. There is a real flow of data between the steps, the output of one step becoming the input of the next.

A client-side mashup can perfectly mixes widgets created from Convertigo transactions with widgets created from Convertigo sequences. As a result, Convertigo literally supports a two-level composition mechanism. Composing at the server-side improves performance and maintainability, while composing at the client-side improves flexibility and reusability.

Another benefit of Convertigo sequences is the ability to map the final result output with an XML business grammar defined by customers themselves (conversely, output of Convertigo transactions cannot be tailored). Convertigo sequences can also rely on Convertigo SQL integrator to store collected and aggregated data into local databases.

The Convertigo Sequencer is another unique feature of C-EMS.

Yet Another Scraping Tool?

Convertigo offers much more than old screen scraping technologies:

› Nothing is installed at the client-side – all capture is done at the server-side.

› With a single tool and API, Convertigo can integrate legacy data coming from multiple legacy resources like IBM mainframes, BULL mainframes, IBM midrange servers (AS400), Unix VT, and Web applications (including applications with JavaScript), databases, XML and Web services.

› Convertigo can access, manipulate, filter, transform and aggregate data and processes from multiple systems at the server-side. This ability relies on a powerful sequencer technology. The basic use case is to combine and mix heterogeneous data and processes at server-side, but it can also be used for more complex orchestration scenarios.

› Convertigo integration is transactional.

› Convertigo can dynamically generate any output format (XML, REST, SOAP) on-the-fly as required by multiple client applications. In particular, Convertigo can generate portable SOAP formats compatible with different platforms and frameworks (Flex, Java, .NET). Convertigo can also generate XML business models defined by customers (XSD).

› Convertigo can generate Web services that can be consumed by any composite rich internet
Convertigo can generate both Web services and visual components (Ajax, Flex) than can be combined in any composition platform and portal, including Adobe’s Mosaic.

Convertigo can integrate legacy mainframe and Web applications at various levels: data, process and presentation.

Owing to the Legacy Publisher, Convertigo can generate a fully-functional Flex application from an existing legacy mainframe application, without Flex programming.

Using the Web Clipper, Convertigo can extract all aspects of one existing Web application, including the presentation layer, and can adapt it to mobile devices displays, without native device programming.

Convertigo visual tool allows for capturing data and processes without any programming.

Convertigo is able to cope with basic changes in the source applications, without having to change captures and redeploy them.

Convertigo server supports multiple performance optimization features (cache, connection pooling).

Convertigo is easy to deploy – either on-precises or in the Cloud.

Convertigo is compliant with modern development standards (REST, SOAP, XML, XPATH, JavaScript, HTML 5, Ajax, J2E) and tools (Eclipse, Tomcat, Flex, Mosaic).

Enterprise Composite Application development

An Enterprise Composite Application development cycle would typically include the following steps:

- Develop Convertigo transactions from legacy mainframe and Web applications (unstructured data) using Convertigo Legacy and Convertigo Web modules in Convertigo Studio.
- Develop Convertigo transactions connected to applications with structured data formats and APIs using Convertigo XML / SQL module in Convertigo Studio.
- Optionally compose transactions into sequences.
- Publish widgets and/or Web services out of Convertigo transactions and sequences.
- Assemble widgets and Web services into Enterprise composite applications using Convertigo Mashup Composer.
Convertigo Studio

The generation of Convertigo transactions, sequences, widgets and Web services is a strategic activity for a company. Building a repository of shared and reusable widgets and Web services is a key for reusability and requires project management skills and knowledge of the Enterprise Information System landscape.

Published widgets must be efficient, scalable and secured and should not impact existing systems in production. They must enforce IT governance rules. They can be used either in composite applications or in any other SOA application.

The combination of widgets and Web Services into composite applications is more situational and tactical as mashups have to evolve as quickly as the business. Power users can easily customize, recombine and redeploy composite applications by themselves, assembling widgets for the task at hand.
Convertigo Studio is built on the Eclipse platform. However the use of Studio does not require any Java programming skills. Convertigo Studio developers visually connect to legacy applications and data sources, capture zones, define generation rules and deploy projects. This is all accomplished by pointing and clicking. Convertigo Studio generates all of the XPATH statements from screen elements dragged and dropped in the Studio wizards.

Once transactions and sequences are designed and tested they can be deployed in Convertigo servers from Convertigo Studio. The same way widgets and Web services can be made available to the Convertigo Mashup Composer.
Convertigo Studio users are typically Web developers knowing HTML (and CSS), Ajax, XML in general (XPATH, XSL, XSD) and Web Services (SOAP, WSDL, REST…).

**Mashup Composer**

Convertigo Mashup Composer (CMC) is the Web-based composition environment. CMC applications are designed as sets of widgets, able to communicate together. Widgets can be a mix of HTML, Ajax or Flex components.

CMC projects are very simple to develop and do not require any programming skills. Applications and screens are composed by picking existing widgets and linking them together through interactions (a little bit like in a workflow). Interactions define how data flows between widgets and how widget actions are triggered. Interactions are also visually defined, there is no programming involved at that stage.

Once defined, applications can be stored in a shared repository, tested and executed.
Enterprise Composite Application deployment

Convertigo Administration

The Convertigo platform includes a Web-based administration console. This console allows deploying, starting and stopping Convertigo projects.

NB: projects can also be directly deployed from Convertigo Studio.

Optimizing Convertigo projects represents a significant part of Convertigo administration. Therefore the console includes a full monitoring tool. Convertigo monitoring is based on analysis of log files, and the tool supports advanced filters. At deployment time it is possible to select logging modules and their detail level.

Readers specifically interested in the performance of Convertigo applications should read the document entitled “C-EMS Performance”, available from the Convertigo Developer Network under the Articles section.

Convertigo Cloud

Convertigo projects can be deployed either on-premises or in the Convertigo Cloud. It is also possible to mix both deployment styles.

Convertigo Cloud is based on Amazon EC2 and complements it with its own monitoring layer, providing additional features for automated load-balancing and fault tolerance.
The Convertigo Shared Cloud is a public space in the Cloud that can be used to test new composite applications. It is free but has no SLA and can be reset at any time.

⚠️ It should not be used for benchmarking or for applications in production.

The Convertigo Private Clouds are secured and isolated spaces with SLA reserved to customers.

Convertigo Cloud is a convenient solution for customers not willing to provision, acquire, install and maintain servers for their new composite applications. There is no training requirement prior to Convertigo Cloud deployment. Subscribers can adjust the IT resources to actual projects requirements and business changes.

**C-EMS use cases**

The following use cases illustrate the power and flexibility of the Convertigo Enterprise Mashup Server. They show how customers in the Financial Services and Insurance industries use Convertigo to build and deploy composite applications to solve business challenges. These are just examples, as the possibilities offered by the platform are virtually endless.

**Smartphone-ization**

Banks willing to port some of their applications to smartphones face a difficult choice: which platform should they choose? Applications developed with native smartphone programming languages get the genuine look and feel and behavior of the underlying platform – but at the cost of portability. It also involves a huge development effort, in order to duplicate the business logic of existing applications, or at least expose specific Web services when possible. Modern smartphones all have a powerful Web browser, but traditional Web applications designed for desktop computers and laptops will not seamlessly run on these platforms, essentially because smartphone displays are smaller and require specific user interaction. Convertigo Web Clipper is the perfect tool to quickly capture essential portions of existing Web applications and recompose them on a dedicated user interface.
that can run on smartphones. In that case, Convertigo captures the data, business and presentation layers to produce optimized and portable Web applications for smartphones

Consolidate partial business views

For historical reasons, most banks and insurance companies use multiple systems to manage customer and service information. It is a common requirement to reconcile these complementary, scattered views of the same data in order to get comprehensive information – allowing better business decisions. In many cases, in-house legacy applications also contain critical data that must be matched with packaged applications. Convertigo Web Integrator and Convertigo Legacy Integrator modules are the right solution for that scenario.

New desktop

One of Convertigo’s customers, a large financial institution, decided to create a new, efficient integrated desktop application for their call center. The decision had been made after an acquisition and a joint-venture between different banks, creating synergies between financial offers, but resulting in multiple applications to be used at the same time. This call center is the main access point used by customers to manage their accounts and assets. Thousands of agents handle millions of calls per month. As a result of the acquisition and the joint venture, agents were forced to use up to 30 different applications in order to address customers’ requests. Most of these applications were either legacy mainframe applications or legacy Web applications (first generation Web applications without Web Services) coming from three different information systems.

Having to switch between so many applications was stressful for operators. It was taking too long to answer basic requests and it was almost impossible to complete more complex ones. In addition, security policies were preventing operators from performing as efficiently as they could.

As a result of these challenges, the customer decided to develop a new call center main application as an enterprise composite application, using Convertigo Legacy and Web Integrators to access legacy applications. Widgets connected to the legacy mainframe and Web applications were combined into business mashups that could be customized later. Operators now have a complete view of customers within a single application, with only the relevant data extracted from multiple systems being displayed – clearly showing which products and services should be proposed to customers. The user interface is unified, reducing training requirement and errors. Operators can now serve many more customer requests per day, and in complex situations, they can store work in progress, work on parallel requests and restart a case when necessary. The new composite application is managing SSO and all user credentials, enforcing data governance policies and reducing security risks.

Leverage existing applications

Another typical use case for Convertigo – very useful when having to deal with large legacy mainframe
applications – is to wrap these applications with a fully-functional composite application with a rich internet user interface. This could be done very efficiently when the legacy application has been designed and generated with a Case tool, because it is possible to use the hierarchic screen detection and user interface generation rules of Convertigo Legacy Publisher to automate up to 95% of the revamping process. With Legacy Publisher, mainframe applications can be exposed as rich internet applications in weeks instead of months or even years with traditional legacy scraping tools.

Conclusion

Composite applications are one of the most powerful development solutions to reintroduce user efficiency as the design center of new business applications. Composite applications provide a way to quickly deliver business solutions by composing visual components and services together into tactical enterprise mashups. In real-world, highly distributed and heterogeneous Information Systems environments, new composite applications must be connected to all enterprise resources, including the vast majority of legacy applications without an API. Convertigo’s Enterprise Mashup platform provides a comprehensive, integrated and unrivalled solution for development of dynamic Enterprise Composite Applications. Convertigo hides the complexity and difficulty of gaining easy and fast access to both legacy and Web-based data.

Composite applications present information in a new and agile way to key stakeholders, allowing them to consume it in an easy and rapid manner. The same presentation interface that end-users work with everyday on the internet and expect to see in the enterprise environment, coupled a seamless and effortless way of utilizing all of the different back-end data and systems together, is now possible.

Customers in the banking and financial services industries are already using this solution to quickly develop and deploy advanced business applications accessing critical legacy resources.
About Convertigo

Convertigo is a global software company that improves data sharing through dynamic composite applications. We improve the business agility needed to innovate, transact, streamline processes, and improve customer satisfaction for enterprise and SMB customers. Founded in 1994, over 300 customers rely on Convertigo to solve critical data access and delivery challenges.

Contact Us

**United States**
San Francisco, California  
+1 415 315-9543  
salesus@convertigo.com

**France**
8 boulevard Dubreuil 91400 Orsay  
+33 1 69 18 79 00  
info@convertigo.com

**Belgium**
Antwerp  
+32 (0)475 74 38 21  
luc.hanssens@convertigo.com

For more information about Enterprise Composite RIAs and Convertigo, please visit us on the web at [www.convertigo.com](http://www.convertigo.com)