



## The OpenSpan Platform. Enabling the New Enterprise Desktop.

### Contents:

1. Welcome to the New Enterprise Desktop .....	PAGE 1
2. How OpenSpan Works .....	PAGE 2
3. OpenSpan Platform Components .....	PAGE 3
4. How OpenSpan Compares to Alternative Technologies .....	PAGE 4
5. Uniqueness of OpenSpan .....	PAGE 5
6. Use Cases .....	PAGE 6
7. Benefits .....	PAGE 9
8. Summary .....	PAGE 9
9. About OpenSpan Inc. ....	PAGE 10
10. Getting Started .....	PAGE 10



## 1. Welcome to the New Enterprise Desktop.

Let's face it, the multitude of applications used within your organization today were just not designed to do what you need them to do. Supporting new business opportunities, adhering to new compliance policies, improving user productivity; these are the challenges you must tackle. But the multitudes of applications still in use within your enterprise are inflexible, not easily integrated, and incapable of being extended without a significant upgrade or re-write. Plus your staff is already fully utilized, your budget is strained, and you don't have enough of the necessary skills to even attempt to modify these applications. Heck, your customer service agents still have to use a 25 year old mainframe application to resolve customer problems.

You have listened to integration pitches from many different vendors over the years; all promising a grand solution to your problems. But guess what? You still have the same problems you've had all along. Spending a significant portion of your budget to bring in a team of professional service consultants to build custom solutions that will leave you in exactly the same place five years from now is not your idea of a solution. Replacing or significantly re-writing your applications isn't either. Your core legacy applications may not be pretty or loved by users but they still access the information necessary to run significant aspects of your business and you aren't about to abandon them.

Sound familiar? Every customer of OpenSpan has dealt with these exact same issues. And every one came to the conclusion that there had to be a better way. There had to be a better way for applications to interact with its users and with other applications. There had to be a better way to simply and effectively build automations, to add new functionality to existing applications, to integrate functionality between applications, and even to build new applications from relevant pieces of other applications.

"There was an instant wow factor with OpenSpan. We had tried to solve our integration challenges several times in the past but couldn't do so without having to re-write or replace many of our applications, each of which represented a significant investment for the company. OpenSpan allowed us to very quickly and easily build integrations that automated many of our agent's tasks, resulting in dramatic improvements to our agent productivity, ability to resolve customer issues more quickly, and thus customer satisfaction ratings."

— Mike Garner, Vice President Call Center Services, Afni Inc.

## 2. How OpenSpan Works

At its core, every application running on the desktop, regardless of when it was built or in what programming language includes a set of objects that interact with the underlying Windows® operating system and provide a basic set of functionality. These objects were originally constructed by the application developers and assembled together with a user interface designed to create a positive user experience. Some applications offer Application Programming Interfaces (APIs), which describe the objects within the application that can be accessed and in what manner. But many applications do not offer API's at all. Even those that do offer API's do not provide a flexible way to extend the functionality to allow for customization. In other words, you are limited to the data integration points that the original developers thought were important, not what is necessarily important for your business today.

The good news is that each of the objects within an application interacts directly with the underlying operating system. Every mouse click, every link, every submit button involves a series of communications between the objects and the operating system. The OpenSpan approach to integration is to leverage the on-going communication between objects and their supporting operating systems in order to perform the prescribed integrations and automations. More specifically, OpenSpan utilizes an event-driven approach in which code is injected into participating applications at runtime with the purpose of performing a given integration or automation once a pre-defined event occurs.

OpenSpan takes a unique approach to integration by delivering a desktop integration platform that enables any object from any application to be exposed, normalized, accessed, and ultimately integrated with any other object in a VERY SIMPLE AND QUICK manner. Effectively, OpenSpan breaks down your applications' walls and allows you to use the core functional blocks (objects) in ways well beyond the applications' original intentions. Now you can connect objects between applications and add logic to form new automations, add new functionality to existing applications, or even to build brand new composite applications that consist of functionality (objects) from multiple applications in a single unified view that dramatically improves productivity.

And OpenSpan allows you to do all of this in a very iterative way, ranging from simple automations that deliver almost immediate benefit to desktop users to more robust integrations between older legacy applications and newer solutions such as Web applications, Web services, Rich Internet Applications, and Mashups that deliver significant long-term benefit to your enterprise.

## 3. OpenSpan Platform Components

There are three primary components to the OpenSpan Platform:

**1. OPENSAN STUDIO:** The OpenSpan Studio is a Rapid IDE environment based on the Microsoft.NET Designer stack that allows you to quickly and graphically build integrations and automations, which combine integrations with logic to perform a desired workflow. It enables you to interrogate your applications to expose the underlying objects, assigns a digital “fingerprint” to uniquely identify each object, build automations and integrations from these objects, test them prior to deploying, and easily administer changes on an on-going basis.

**INTERROGATION:** The OpenSpan Studio allows you to easily interrogate applications to expose all of its key objects. Simply drag the interrogation icon over the important functionality within the application (fields, buttons, grids, etc...) and the OpenSpan Studio loads the object specifics including all of its properties, events, and methods. It is important to note that virtually any application can be interrogated including mainframe/green screen, Java, Windows (WIN32), and Web applications; even Web services. The extensibility of the OpenSpan Studio allows for any .NET component to be imported as well as standard components such as Microsoft Office. In this scenario, all properties, methods, and events are exposed without having to perform the interrogation step.

**OBJECT LIBRARY:** Upon interrogation, each object is assigned a digital “fingerprint” that includes XML meta-data that describes where an object resides and how it can be accessed. This allows the OpenSpan Integrator (desktop runtime environment described later) to properly access and interface with the object at runtime. It also normalizes all objects, enabling them to be treated as re-usable components that are stored within the OpenSpan Studio library. An object, regardless of the application in which it resides, becomes simply a building block for integrations and automations now or in the future.

**VISUAL MODELING ENVIRONMENT:** A highly powerful and simple-to-use visual modeling environment is the most popular component within the OpenSpan Studio. Similar in functionality to modern business process management / workflow visual modeling environments, OpenSpan Studio allows developers, or even sophisticated business analysts, to quickly build new integrations and automations by dragging-and-dropping objects and easily connecting them together all without coding. It also provides a very effective mechanism for building new composite applications from a set of objects spanning multiple applications and includes a highly intuitive User Interface design toolset.

**TESTING:** All integrations, automations, and composite applications can be run within the OpenSpan Studio environment to ensure proper functionality before deploying.

**ADMINISTRATION AND CHANGE MANAGEMENT:** The OpenSpan Studio creates an XML file (solution) that is hosted on a server and deployed across your desktop environment automatically through the OpenSpan Platform. When the Integrator (runtime) is run, it performs a brief version check to confirm that each of the defined integration solutions is current with the version deployed on the server. Changing or updating integrations are as simple as updating the solution files hosted on the server, which are then downloaded and run by the Integrator runtime within each desktop environment. This ensures that all users' desktops deploy the most recent integration versions.

**2. OPENSAN INTEGRATOR:** The OpenSpan Integrator runtime is built on and extends the Microsoft .NET Framework and acts as a lightweight desktop deployment runtime. After an integration, automation, or composite application is built and tested within the OpenSpan Studio, it is deployed seamlessly across users' Windows® desktop environments. The Integrator utilizes an event-driven approach in which code is injected into an application upon starting (as defined by the integration solution file) that then executes pre-defined integrations and automations based upon certain events occurring.

**3. OPENSAN SOA MODULE:** An optional plug-in, the SOA Module exposes methods and properties of any Web Service for consuming new or existing Web Services within the OpenSpan Platform. The SOA Module provides the 'last mile' for SOA by enabling Web Services and Rich Internet Applications (RIA) to be easily and immediately integrated with all other applications that can be integrated with the OpenSpan Platform including older legacy applications, new applications as they are created, third-party (packaged) applications, custom-built applications, Microsoft Office® applications, data sources, etc...

## 4. How OpenSpan Compares to Alternative Technologies:

There have been a number of technology approaches for application integration over time and new integration niches seem to appear every year. Virtually all technology approaches access application data in one of two ways:

**1. VIA BACK-END ADAPTERS, APIS, OR WEB SERVICES.** Technologies such as Enterprise Application Integration (EAI), Service-Oriented Architectures (SOA), Enterprise Service Buses (ESB), Enterprise Information Integration (EII), and even new Mashups access data from applications (including Web applications) by accessing data through new or existing 'tunnels'. These tunnels include any existing application API's (Application Programming Interfaces), third party-created adapters, or increasingly thru Web services, which expose underlying objects in a standards-based manner for purposes of simplifying integration. Virtually all of these technologies offer a number of advantages because they are server-side based. This includes ease of deployment, updating, and management along with scalability, security, and reliability. The key challenges associated with back-end integrations are that they are very often highly complex to build because they require a deep understanding of the original application's programming language and/or business logic, and may require access to the underlying source code. The result is that most back-end integration projects are very costly, require a significant amount of development time, and often require an 'all or nothing' approach that can be highly risky to the enterprise. Another key challenge is that adapters, API's, and Web services are not available at all for many applications, thus limiting the scope of many integration projects or requiring such applications to be re-written or replaced.

**2. VIA AN APPLICATION'S USER INTERFACE.** This technology mimics the actions of a user; automating the steps that would normally be taken manually by a user of the application. As such, GDI-Hooking, commonly known as Screen Scraping, remains a cost-effective way to build simple integrations and automations between applications. The primary challenge of this approach is that it is not a robust, scalable, or stable solution; requiring object fields to remain in the same geographic location within an application in order to function properly. Integrations break easily when applications are upgraded or when customizations are made to the user interface. Additionally, unlike the OpenSpan approach that enables applications involved in an integration or automation to be hidden from view in order to remove complexity for the user, screen scraping requires applications involved in an automation to be visible.

OpenSpan takes an altogether new approach to application integration. The magic of OpenSpan is that it interfaces “directly” with the objects within an application; residing between the application’s objects and the underlying Windows® operating system(s) that deliver the functionality of the object to the user. Because of this approach, OpenSpan allows developers to build integrations and automations:

- Between any objects “quickly and easily” within any application including mainframe/green screen, custom-built applications, desktop applications, Java or Windows-based applications, new Web applications, even Web services
- Without requiring programming-specific language skills
- Without requiring access to the underlying source code
- Without requiring re-compilation of the application’s code
- That can be easily and quickly updated if changes to the application or its user interface occur. In fact, often times minor changes to the application do not require any modifications to integrations and automations at all

The other important aspect of OpenSpan is that it is very complementary with server-side integration technologies, including Web services. The OpenSpan Platform enables legacy applications including custom-built applications and older applications without an available API or adapter to participate in a Service-Oriented Architecture. Additionally, the OpenSpan Integrator run-time environment can be hosted on a Citrix environment and also in a mixed desktop/Citrix session environment, depending upon your specific requirements. You can also build integrations consisting of objects hosted both in a Citrix environment and on the desktop.

## 5. Uniqueness of OpenSpan

OpenSpan customers continually highlight two areas for which OpenSpan technology is truly unique:

**1. THE SPEED AND SIMPLICITY IN WHICH ROBUST INTEGRATIONS AND AUTOMATIONS CAN BE BUILT AND MAINTAINED.** The OpenSpan Platform includes the OpenSpan Studio, a visual programming environment that allows you to quickly interrogate applications and within seconds expose the critical objects that make up those applications. All objects, regardless of their parent application, are assigned a digital “fingerprint” that normalizes them thus allowing any object to be treated as a standard building block within the OpenSpan Studio. The visual programming capabilities allow a developer to rapidly assemble these objects to build integrations; accomplishing in hours what normally requires weeks or months with alternative integration techniques. Additionally, integrations and automations can be easily modified using the same visual paradigm to account for application changes or upgrades.

**2. THE ABILITY TO EASILY INTEGRATE ANY OBJECT WITHIN ANY TYPE OF APPLICATION.** Although a common claim among integration software vendors, only OpenSpan allows you to quickly and easily integrate any application; including home-grown applications, older legacy applications such as mainframe applications, and applications that reside on your user’s desktops. These applications can now be easily extended; for example by adding new regulatory compliance process to older applications without even needing to have access to the underlying source code or any application programming-specific skills. OpenSpan also enables the ability to integrate Web services with these older applications; effectively delivering the last mile of SOA by leveraging newly created services to positively impact real-world desktop user experience. In addition, OpenSpan allows any object within an application to be interrogated and thus integrated – unlike alternative technologies that are limited to functionality exposed through user interfaces or the application’s API.

## 6. Use Cases

There are a seemingly endless number of use cases that can apply to the OpenSpan Platform although the use cases below represent the most common successes to date. One of the major advantages of the OpenSpan approach is that it allows enterprises to employ an iterative approach to achieving ROI from application integration projects. Simple automations enjoy almost instant productivity gains and a very quick ROI while more strategic integration projects such as Enterprise Mashups and SOA re-architectures can employ OpenSpan integrations to extend their functionality to legacy applications including custom-built and other desktop applications.

**SIMPLE AUTOMATION:** One of the quickest ways to impact the business is to build simple automations that reduce repetitive tasks needed by employees. Specific examples include:

**1. UPDATING CUSTOMER INFORMATION.** The process of updating customer information can be a time-consuming challenge for contact center agents. Sometimes referred to as 'Swivel Chair' or manual integration, copying-and-pasting customer info to multiple applications is a very time-consuming task that restricts the overall ability of an agent to solve real customer problems. Simple automations can be created with OpenSpan to mimic the actions of the agent, thus eliminating the need for manual copying-and-pasting and freeing the agent to focus on resolving customer issues. For example, if an agent changes a customer's contact information in their core application, the same change can be made automatically to the company's billing application(s), shipping application(s), CRM application(s), or any other application.

**2. INCREASING UP-SELLING OPPORTUNITIES.** Although clearly beneficial to the bottom line, identifying up-selling and cross-selling opportunities often requires data to be obtained from more than one source and presented to an agent or sales representative at the right time. OpenSpan can automate the business processes necessary to determine the available up-selling or cross-selling opportunities and present them to the customer service agent at the appropriate time during the call. Call or customer feedback can also be logged automatically based on criteria defined by the business (such as customer type or specific product being sold) and distributed to multiple applications or managers.

**3. MANAGING BUSINESS PROCESSES.** Having all users follow defined business processes all of the time can often be difficult or near-impossible, particularly when doing so means having to learn the inner workings of multiple disparate applications and the associated workflows. This becomes even more challenging when new products or new processes are introduced. With OpenSpan, business logic that spans multiple applications can be added to ensure that users fully complete one part of the process (e.g. entering a valid email address or reading a particular script) before moving onto next steps.

**4. AUTOMATING MULTIPLE LOGINS.** Similar to the example above, customer service agents often waste a lot of time each day logging into the multiple applications, often multiple times during the day. This generally extends customer calls and lengthens average hold times for contact center agents. OpenSpan automations can be built that enables an agent to login to a central application at the beginning of their shift and then be automatically logged into all relevant applications continually throughout the shift.

**COMPOSITE APPLICATIONS:** Many OpenSpan customers choose to build composite applications or dashboards that allow a new unified view to replace the need for employees to interface with multiple applications. This often delivers tremendous ROI to the business in terms of improved productivity, improved performance, and reduced training costs. Composite applications also provide a way to limit access of certain applications or functionality based on user type. Specific examples include:

**1. CONSOLIDATION OF APPLICATIONS, INFORMATION, AND WORKFLOW.** A simple composite application can be created to streamline business processes that typically span multiple applications and/or data sources. Instead of accessing or processing information across a number of different application user interfaces, a single unified view into all applications and data sources can be utilized by an employee. Remaining with the contact center industry examples, the processes followed by a customer service agent could be dramatically streamlined by allowing them to interact with relevant customer information in a single dashboard view that interfaces behind the scenes with the CRM, shipping, billing, and other applications. In addition to streamlining agent productivity, composite applications can dramatically reduce training times for new agents.

**2. LIMITING APPLICATION ACCESS BASED ON ROLE.** Often times, users of today's applications are not the original intended audience for an application. For example, most financial applications were written for accounts and other finance employees, not customer service or collections agents. While it's often important for agents to have access to select functionality or data within an application, it may not make sense for these roles to have access to all functionality or data within the entire application. Creating a composite application with OpenSpan can solve this problem by providing access to only well defined objects within an application. Not only does this provide enhanced security and regulatory compliance, it can also dramatically simplify the training necessary and reduce the number of applications in which an agent must interface.

**EXTEND LEGACY APPLICATIONS:** Because the OpenSpan Platform provides you control over your applications' objects, a very powerful capability of the OpenSpan Platform becomes the ability to add new functionality or business logic to an existing legacy application. This is an especially important capability for enterprises that have significant investments in legacy applications which are considered 'closed' (no available API), where an enterprise has little understanding of the underlying logic of the application, or where they lack the necessary skills for the underlying programming language. Specific examples include:

**1. ADDING COMPLIANCE TO AN EXISTING APPLICATION.** New regulatory compliance issues such as Sarbanes-Oxley create new challenges for enterprises and require changes to existing applications and business processes. This presents extremely difficult challenges for enterprises that have a significant investment in legacy applications that are not easily modified or extended. OpenSpan actually allows new logic to be added to existing applications easily, resolving this issue. For example, customer interaction logs can be added to existing applications for compliance auditing purposes. Similarly, an automation can be deployed that provides customer service representatives explicit messaging that must be discussed with a customer or prospect dependent upon their State of Residency, in order to comply with State-specific regulations.

**2. ADDING VALIDATION TO AN EXISTING APPLICATION.** Similar to above, automations can be deployed that ensure that data is validated prior to submitting into an application. For example, an automation for a billing application could be created such that a credit or wire transfer could not be executed for an amount that exceeds a defined limit amount for a given customer or customer group. Similarly, an automation could be deployed that ensures that the data entered in a given field matches the desired data structure such as requiring a Social Security number to be 9 digits in length.

**3. LEVERAGING WEB SERVICES TO EXTEND APPLICATION FUNCTIONALITY.** Web services can deliver new functionality in a very accessible manner, but integrating such functionality into an existing legacy application can be complex and extremely time-consuming. OpenSpan enables Web services to be consumed in the same manner as any application object and thus allows Web services functionality to be easily added to legacy applications. For example, a financial institution could build an automation that accesses existing web services from UPS, FedEx, and the US Postal Service to determine the least expensive shipping option for a particular customer shipment and present that data to a customer service agent in real-time.

**4. LEVERAGING DATA FROM THIRD-PARTY WEB APPLICATIONS.** Similar to the example above, data from third-party Web applications can be obtained via an automation and posted to an internal legacy application. For example, an automation for a banking institution application could be deployed whereby competitive Savings or Certificate of Deposit banking rates are obtained from publicly-available Web applications or Websites and presented to an agent in the form of a pop-up window. This would allow the customer service agent to know in real-time what competitive pressures exist without requiring that agent to manually lookup the information.

**ENABLING LEGACY APPLICATIONS TO PARTICIPATE IN SOA ENVIRONMENT:** Not only can the OpenSpan Platform consume Web services as described in the above examples, integrations and automations created by the OpenSpan Platform can be consumed as part of an enterprise's next generation SOA environment. Specific examples include:

**1. ENTERPRISE MASHUPS, PORTALS, AND RICH INTERNET APPLICATIONS.** Many enterprises are building these next-generation applications that mix and match data from multiple internal and external applications and services to create a single customizable and dynamic interface. OpenSpan makes it much simpler for existing information assets from legacy or desktop applications to be included in these solutions. For example, a line-of-business mashup might be extended to include data from a 'closed' legacy application that does not expose data via an API or a Web service and automatically create reports, e-mails or alerts via Microsoft Office®.

**2. SOA 'LAST MILE':** Enterprises moving towards a Services-Oriented Architecture will benefit from creating standards-based business processes formed across a spectrum of applications and data sources. While Web Services play a key role in enabling SOA, exposing all existing applications as Web services, especially older legacy applications such as mainframe/green screen applications or custom-built applications, will be difficult and/or impossible given budget, risk, time, and skill mismatches. Because OpenSpan enables the objects within any application to be exposed and integrated, new application deployment becomes significantly easier since the Web services and Rich Internet Applications providing new business logic can be rolled out and operated successfully within the existing environment; thus making OpenSpan the true last mile of SOA.

## 7. Benefits

Hopefully the section above demonstrates how OpenSpan can deliver value in many different scenarios. The uniqueness of the OpenSpan Platform is that it enables objects from “any” application (from legacy mainframe or custom-built applications to Web applications and Web services) to be integrated together and to do so in a “very” simple and quick manner compared to competitive technologies. Specific benefits include:

### **GENERIC BENEFITS:**

- ❑ Iterative approach for more rapidly realizing ROI benefits of application integration projects
- ❑ Dramatically simplifies and expedites the application integration process, thus reducing the risk for new integration projects
- ❑ Reduce operational costs by streamlining existing business processes
- ❑ Extend existing applications to comply with new business requirements (e.g. regulatory compliance)
- ❑ Restrict access to certain application functionality for particular roles
- ❑ Extend legacy applications to participate in next-generation portals, mashups, and rich Internet applications

### **ADDITIONAL CONTACT CENTER-SPECIFIC BENEFITS:**

- ❑ Improve effectiveness of up-selling and cross-selling campaigns by providing timely and accurate access to customer, competitor, and promotional data
- ❑ Providing a single unified view to streamline contact center agent processes
- ❑ Reduce average handling times thus improving agent call volumes
- ❑ Improve customer satisfaction by resolving issues more quickly with fewer hops
- ❑ Reduce training costs for new and existing employees
- ❑ Comply with Federal and State regulatory compliances

## 8. Summary

OpenSpan allows you to quickly and easily build, deploy, and administer integrations and automations and thus begin realizing the associated ROI benefits more quickly and with less risk than previous technologies. The OpenSpan approach to application integration allows you to expose the underlying objects within your applications (including desktop and custom-built applications), treat them as standardized building blocks, build powerful and robust integrations and automations via a visual programming environment, and deploy and administer them seamlessly across your desktop environment. And it enables you to do so in a fraction of the time compared to alternative technologies.

With OpenSpan, your applications can now better support your business and perform as you need; beyond even how the original designers intended. OpenSpan allows you to tear down the walls between applications to build integrations and automations that solve today’s challenges while also better preparing you to support the challenges of tomorrow by providing the link between server-side Service-Oriented Architectures and the applications, data, and user activity taking place on the desktop.

## 9. About OpenSpan Inc.

OpenSpan is a three year old software company founded by a group of technologists with extensive experience building integrations for Fortune 100 enterprises; designed specifically to assist contact centers simplify various computing processes for their customer service agents. The core product offering, the OpenSpan Integration Platform dramatically simplifies and expedites the application integration process within contact centers and across other industries as well. Companies such as Alltel Wireless, JC Penney, AAA, Charter Communications, Aon, and Afni have deployed the OpenSpan Platform to over 24,000 of the most demanding desktops environments around the world. Recognizing the unique approach to the difficult challenge of application integration, Gartner awarded OpenSpan with a 2006 Cool Vendor Award.

## 10. Getting Started

If you would like to learn more about the OpenSpan Platform, we invite you to contact us directly via phone at +1 (678) 527-5416 or e-mail at [sales@openspan.com](mailto:sales@openspan.com). Our team of integration professionals will be glad to assist you in determining if OpenSpan would be of value to your organization. They can also arrange an online or in-person demonstration of the OpenSpan Platform if so desired.

Additional resources are also available on the OpenSpan website at [www.openspan.com](http://www.openspan.com).