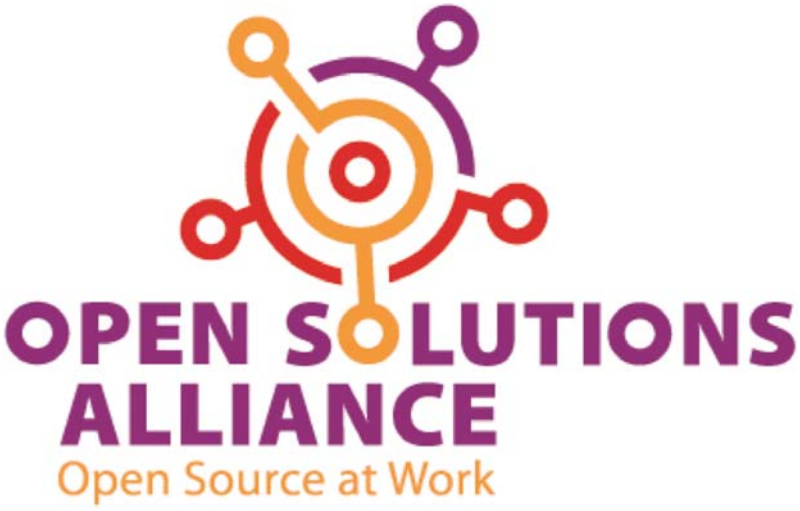

Approaches To Interoperability

An Open Solutions Alliance White Paper

The Case for Open Source Interoperability

Interoperability has been a major challenge facing the business software consumer for decades. Software applications, each designed with distinct requirements in mind and built on technology stacks by engineering teams working largely in isolation, were simply not designed to work together. The result has been customers enduring the time and expense of getting everything to work together.

The industry has attempted successive waves of standards with limited success. Many of the resulting standards are extremely complex, leading to poor understanding of their meaning and resulting in incompatible implementations that claim to be compliant. Another problem is the proliferation of standards, with multiple competing, incompatible standards often sponsored by competing proprietary vendors. This complexity and poor interoperability has been a barrier to adoption in most small to medium businesses, which frequently need a smaller solution that "just works".

The last five years has seen the emergence of technologies and development methodologies that can radically improve the interoperability landscape. The emergence of the web and its related technologies such as AJAX and Web Services make it much easier to assemble software pieces together into a new application. At the same time open source has emerged as a new way to develop software, with a strong emphasis on modularity and doing one thing extremely well. The result of all these trends is rich set of powerful building blocks, but not ready to use solutions.

The Open Solutions Alliance

The Open Solutions Alliance (OSA) has been formed expressly to help speed the creation and adoption of integrated, interoperable business applications based on open source. The OSA approach to interoperability is pragmatic – focusing on solutions, not standards. By raising awareness of interoperability problems and promoting guidelines for interoperability, the OSA hopes to sponsor the ongoing evolution of interoperability practices. We think that open source's spirit of openness and collaboration is the ideal way to develop and define these practices. One of the goals of the OSA is to develop, via community participation, a set of interoperability guidelines, eventually leading to a certification program.

The OSA plans to address the major interoperability issues faced by small and midsize businesses. We will develop a series of guidelines and best practices that detail how open source applications should support interoperability. The initial interoperability topics include compatible infrastructures, shared services, common standards, support models, the user experience and business models. Because open source projects are written in a variety of languages and support numerous operating systems each recommendation will clearly identify where they are applicable, such as "Java web applications" or "C++ applications on Linux".

All guidelines developed by the OSA will be the result of a public discussion on the OSA forums, where anybody can participate. When a new recommendation is drafted it will

be freely available as a public draft for 60 days before it can be approved by the OSA members.

Interoperability Issues

Open source projects focus on doing one thing, and doing it well. A business-ready solution consists of multiple open source applications and projects combined together into a single system typically called a "stack". In this paper a set of projects combined together will be referred to as a "solution". The people doing the integration will depend on the circumstances - it may be by an OSA member, another software vendor, a systems integrator, or even by an IT professional at the end user's company.

During the course of this paper we will use a hypothetical example of a customer relationship management (CRM) solution that consist of a CRM application, an analytics and reporting module, a document management system to hold contracts and other legal documents, a VOIP system for calling customers and perhaps a database. The various components of the solution and their relationship to the interoperability issues discussed in this paper are show in Illustration 1.

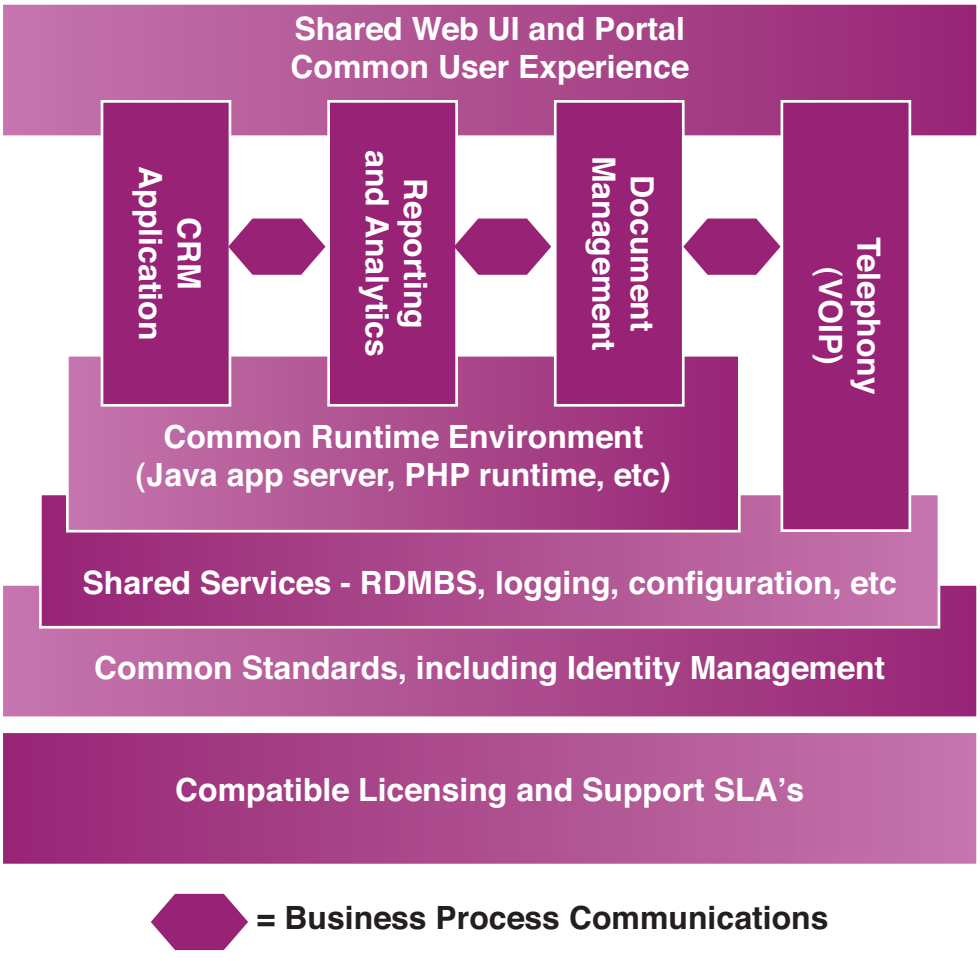

Figure 1 Hypothetical CRM Solution

Combining multiple projects together like this into a solution brings a myriad of issues up. What if a potential customer of this solution already has an Internet based PBX in place and wants to use it in place of the VOIP system? The amount of work required to replace the VOIP system and the usability of the end result directly depends on how much emphasis on interoperability went into the components of the solution and into the integration code itself.

The OSA interoperability guidelines are intended to make both the initial integration and this type of replacement of a component much easier than it traditionally is. There are seven categories of interoperability that will be the initial focus of the working group.

Standards

Software developers often joke that the best thing about standards is that there are so many of them to choose from. This is actually a problem for the industry, as it is difficult to choose between competing standards. Worse, competing standards are often incompatible with each other, forcing a developer to either decide which camp to support, or spending time duplicating work to have a version of their product that support each standard.

The Open Solutions Alliance does not intend to draft new standards but supports the use of existing standards, both formal and ad hoc, where it makes sense. One common type of application integration is "presentation integration", where the applications are actually separate but combine their presentation layer in a single web page via a portal. The OSA will host a community discussion of the various portlet standards such as JSR-168 from the Java Community Process and OASIS's Web Services for Remote Portlets. We will use the community's input to recommend all OSA member companies support a specific standard.

A community member may nominate any standard, with the exception of standards only supported by one vendor where that implementation is not available under an open source license. De facto or ad hoc standards (such as Hibernate for Java object relational mapping) are eligible.

Common User Experience

A single business application built from disparate applications needs to integrate the presentation layer of each component application into a seamless experience for end users. In each case, end users should be able to interact within one window. In our example of a CRM solution the user should be able to go from a customer information screen to a report on all contacts with that customer without switching applications.

The user interface is critical to presenting a seamless integration. The end user rapidly becomes disorientated and frustrated when the various screens or tabs in an application have a distinct look. To minimize this the look and feel of solution components needs to be customizable, while allowing vendors to preserve their branding.

Another aspect of the user experience that the OSA plans to address is recommending a common search mechanism. In our CRM example a search should be able to produce any matching records from the CRM database and any documents from the document management system.

Identity management and Single Sign On

One of the challenges most IT departments face is the task of provisioning new users. Typically a new user is added to a central authentication server, and then to multiple applications that cannot talk to the authentication server. Many applications that can use an external authentication server still require local roles and permissions to be set up for a user, and keeping all this information consistent can be both a major headache and a source of security breaches.

Another problem when different applications use their own authentication and security system is that they force the user to log in to each application. Users rapidly get annoyed when what appears to be a single application prompts them for their password multiple times.

The OSA interoperability guidelines will recommend that all applications support directories of users and roles. We will also host a discussion on single sign on methodologies and select one to recommend.

Infrastructure

In an ideal world, the varied applications packaged together into a solution would run against the same database, use the same servlet engine (for Java applications), and so forth. In the real world, this is difficult. "One size fits all" infrastructure stacks have been attempted by the industry, and grow quite large depending on how many of the myriad applications' dependencies they attempt to tackle. Similarly, having to run two different databases or two different portals places a needless stain on your computing system.

The OSA believes a better approach is to minimize dependencies on specific pieces of the infrastructure stack. If you can avoid using a specific dialect of SQL, do so. If you can't, consider using a database abstraction layer that eases the task of supporting multiple dialects. We will publish a series of best practices on how to avoid dependencies on a specific application server, operating system, portal or database. We will also work with the community to identify and recommend specific projects in areas where you have no choice but to choose just one solution, such as an application framework or persistence layer.

Management

Application interoperability goes beyond the integration and end user issues discussed so far. When a business attempts to roll out a solution to its employees an entirely new set

of concerns need to be addressed. The various applications that make up the solution need a uniform approach to the day-to-day care and feeding of the deployed solution.

The key to a successful deployment is having all the components appear as a single deployable application, not as a thin layer over disparate systems. The combined products should have a single installer and should have a single point of configuration. Nobody wants to set up an application that involves running seven installers and adjusting six different configuration files.

Other areas that are important include monitoring the deployed solution, having a single log file for all output, a common backup and recover mechanism, and an easy way to upgrade the entire suite at once.

Application Compatibility

Moving beyond purely technical issues, there is another layer of interoperability that face organizations trying to deploy multiple open source applications. The applications have to be use compatible licenses, need to offer similar support levels, subscription terms, etc. The combination of multiple applications is only effective if the weakest offering is acceptable. In our CRM example, having 24 by 7 support from the CRM vendor is useless if the VOIP system fails and only has email support with a 24-hour response time. Other important areas include the supported locales of each application – if you want to roll out the solution in Asia but only two of the five components support multi-byte characters sets the deployment will fail.

Our goals for application compatibility are to identify the non-technical areas that can prevent applications from working together, and define a categorization scheme that allows a user to quickly determine if two applications are compatible in ways that matter to that user. It is not our intention to make specific recommendations of what license members should use or what their support model should be, only to raise awareness of the issues.

Business Processes

In many cases integrating applications into the same infrastructure or a common user experience does not solve the problem. There may be a sequence of activities that require the applications to communicate with each other, they may need to share a data model, or both. Suppose a business wants to integrate the CRM solution with their shipping system, so when an opportunity is close a bill of material is sent to the shipping system so the order is automatically fulfilled. At a minimum the CRM solution needs to be able to tell the shipping system about the order, and preferably the two systems will share a common definition of the customer – the CRM system will use the contact information and the shipping system the ship-to address.

Applications to application communications such as these are frequently custom in nature. Standard business processes exist in some industries, but even then customers augment and extend the processes to fit the way their business works. In short, these processes are nearly impossible to build into a general solution.

The OSA recommends that each vendor or project lead carefully think about which functions in their application are going to need to be triggered by other applications, and ensure that they are exposed in a loosely coupled way so customers and integrators can take care of implementing the process. These functions should be exposed as a service and should be implementation language neutral, so for example a PHP application can invoke a feature in a Java application

The OSA does not intend to define data models for use by applications. We hope that our community of vendors and users will collaborate on common data models on our forums, be we will not bless or endorse any specific models.

Summary

Interoperability between independent software projects is a significant challenge, and there is no "one size fits all" solution. The selection of interoperability technologies and methodologies depends on the type of projects being offered. The OSA's interoperability working group plans to draft a series of recommendations that can be used by developers to help make their applications easier to integrate into larger solutions, and can be used by businesses when evaluating projects to adopt.

For all the interoperability topics the common theme is awareness. As mentioned earlier, open source communities are ideal for sharing this kind of information. Experiences can come directly from customers and integrators to a public forum, where community developers can respond, ask questions, and if so inclined, build a means of integrating, which is then subsequently iteratively improved by those same customers, integrators and other developers. The OSA's mission is to foster this kind of interaction and dialog, for the benefit of all open source projects relevant to business solutions, the vendors and integrators that support them, and ultimately the end customers willing to adopt them. Together, as a broad community, we can gradually and iteratively succeed where multiple attempts at "grand architecture" have failed.