Interchange Introduction

Table of Contents

1. INTRODUCTION TO INTERCHANGE

Akopia's Interchange is the industry's most widely distributed and implemented open source e−commerce platform. The software is equipped with a wide range of features that make it the most comprehensive, reliable, and cost−effective option available today. This introduction provides a background on Interchange, an overview of its capabilities, a description of how it works, and answers a number of frequently asked questions.

1.1. Background

Interchange is a descendent of Vend, an e−commerce solution originally developed by Andrew Wilcox in 1995. In the five years that followed, Mike Heins expanded and enhanced Vend, creating a more powerful and versatile tool, MiniVend. MiniVend grew to support thousands of businesses and their e−commerce sites. However, despite the fact that MiniVend was feature−rich, it lacked an intuitive administrative interface. A similar product, called Tallyman, had the features that MiniVend lacked. It was intuitive and had better content management features, but lacked some of MiniVend's transactional capabilities.

During the same period of time, an experienced e−commerce development team founded Akopia. Their goal was to create a sophisticated open source e−commerce platform that was both feature−rich and easy to use. Akopia acquired both MiniVend and Tallyman and integrated them to create Interchange.

1.2. Overview

Based on years of evolution and experience in the open source community, Interchange offers an extensive feature set usually only associated with costly enterprise applications.

Interchange is a high−end, fully customizable software system with complete database functionality. It provides flexible page display, search, and order entry capability. Interchange can support catalogs of over a million items with excellent performance. Interchange also has the capability be used for many applications besides shopping carts, such as a complete database−oriented content display system.

Interchange can work with a web server that supports Secure Sockets Layer (SSL), allowing encrypted transmission of sensitive customer data. This capability makes entering credit card numbers practical and secure. In addition, it supports online payment systems and advanced encryption.

A single Interchange server can support a large number of independent Interchange catalogs, allowing an Internet Service Provider (ISP) to serve many different customers from one or just a few Interchange server processes. As many as 2,000 Interchange catalogs have been run on one machine from the same server process.

2. HOW INTERCHANGE WORKS

This section describes how Interchange works.

2.1. The Vend Concept

The basic concept of implementing Interchange remains unchanged from Vend. Interchange maintains a unique set of pages, outside of regular HTML space, which contain special tags that are interpreted by the program. The tags provide access to special Interchange functions within the Interchange server.

These Interchange tags, which are in `[square brackets]` format, have a wide variety of functions. Some examples are:

User Form Input

Interchange recalls input by a user so that it can be easily passed from form to form. The value of any form variable is "remembered" and inserted upon finding a `[value input_field]` tag. The `input_field` is a normal HTML form field.

Database Contents

Interchange can support an unlimited number of attached databases, either in one of its own internal formats or one of a number of popular SQL databases, and in any database with an ODBC interface. The contents of a database can be referenced with tags, i.e., `[data table=products column=name key=334-12]` or `[query sql="select * from products where category = 'Clothing'"]`.

Session Parameters

Session parameters, as the name suggests, include information specific to the particular user and his or her current interaction with Interchange. The parameters include the location where the user originally found the catalog (`[data session referer]`), the domain they are from (`[data session host]`), the source of the hit in a partner program (`[data session source]`), the time of their last access (`[data session time]`), and so on.

Embedded Perl Versus ASP

Interchange has a powerful object system that allows direct access to Perl and to external programs. An ASP-like syntax can be used, or the traditional Interchange tag approach can be employed.

File Contents or Program Output

The contents of an outboard file can be inserted with `[file directory/file]` or `[include directory/file]`. It is also possible to include the output from an arbitrary program.

Searches of Files

Interchange supports a variety of search engines, including *Glimpse*, a popular open source search engine. It can also process the output from custom SQL database queries.

There are over 80 distinct Interchange tags that support hundreds of functions. In addition, Interchange allows user−defined tags to be created. User−defined tags are as powerful as Interchange's standard tags.

2.2. A Typical User Session

The customer discovers a merchant site using the Interchange catalog with a search engine, a link from another page, or a click−through from a banner ad. He or she clicks the link, which is a URL pointing to the Interchange CGI link program (generically called VLINK or TLINK). The link calls the Interchange server through a socket. The Interchange server detects the path information from the link, and loads the corresponding page.

The displayed page contains links to either locate or order items from the merchant's catalog. When the customer clicks a link to purchase an item, Interchange searches the product database, finds the item, and places it in the customer's shopping cart. (Each user has a separate shopping cart, which is attached to their session.)

Once the customer decides to purchase the items in their shopping cart, he or she checks out by completing a form with their name, address, payment information, shipping information, and any other information that may be required. Interchange has the capacity to compute sales taxes and shipping cost. The order is then submitted. The customer's payment can be taken online by real−time electronic payment, or their order information may be transmitted using encrypted email or FAX to a processing center.

The customer's order is saved to a file or to a database table as backup. In the case of fully automated systems, it can be sent directly to an order entry program or database link.

All of these operations are fully configurable. The Interchange demo program includes a sample store called Construct Something. Some users have actually customized the text and images inside Construct Something's sample catalogs, changed the database entries, and opened their store. Most users, however, have built their catalogs using their own distinctive look and feel.

2.3. How Interchange Manages Sessions

Normally, each request for a web page stands on its own. While the server is usually able to identify the particular machine that was the source of a request, it may not know if the next request comes from the same browser or even from the same user on that machine. Interchange automatically keeps track of each user session by one of two methods:

Cookies

Interchange can issue cookies that contain the user session ID. If the user returns the cookie, then the user can be presented pages without accompanying session information.

URL Re−Writing

If the user doesn't want to use cookies, Interchange will include a session ID in the URL. This is a random−looking piece of text that is unique for each customer browsing the catalog. The text allows Interchange to sort information presented to the system from simultaneous users.

2.4. Page Delivery in CGI−BIN

Pages in the catalog served by Interchange running as a CGI−bin program generate a special URL for every link. For example:

http://www.mystore.com/cgi−bin/simple/browse1?id=WehUkATn&pc=122&ar=99−102

The following is a description of each part of the URL link:

www.mystore.com

The Internet address of the server hosting the Interchange catalog.

cgi−bin

Informs server that the requested page will be generated by a program. This can be HTTP server−specific.

simple

The name of the program to be run. This is Interchange's VLINK or TLINK. The link program is a small compiled program that connects the web server to the Interchange server. The Interchange server remains running in the background, fully initialized, and able to quickly process requests.

browse

The page of the catalog to display.

id=WehUkATn

The session ID. This is used if cookies are not enabled.

&

Separates the session ID from other parameters (normal HTTP).

mv_arg=99−102

An argument usable by Interchange to select page display options.

&

Separates an argument from other parameters (normal HTTP).

mv_pc=122

A unique integer (or source code, if it contains a letter) that prevents caching servers from caching the URL.

2.5. Page Delivery in HTML

Interchange pages are written in regular HTML with extensions (the Interchange tags) to support catalog ordering. Interchange extensions look like this:

<A HREF="[href specials]">See our specials!</A>

Pages are delivered through the following steps:

1. The HTTPD server (Apache, Netscape, or NCSA are examples of HTTP servers) receives a request for an Interchange page.
2. The server is already running as a daemon, and the request calls a small compiled C program (source is vlink.c or tlink.c) that is named according to which catalog is being called. This program communicates with the Interchange program using a UNIX− or INET−domain socket.
3. Interchange reads the source page from the Interchange pages directory and interprets the Interchange tags in the file. If the page doesn't exist and corresponds to a part number in the database, it is dynamically built using a template page. In the process, it can read or modify any number of database tables. If the user's browser doesn't accept cookies, any links generated on the page will contain the session ID, which is needed to ensure the user's session is retained.
4. The page, which is now entirely in regular HTML, is delivered to the HTTP server, which returns it to the browser.

3. TECHNICAL REQUIREMENTS

This section describes the technical requirements needed to run Interchange.

3.1. Perl

Interchange 4.6 requires Perl version 5.005 or higher. Perl 5 can be downloaded from any CPAN (Comprehensive Perl Archive Network) site. See the following site:

http://www.perl.com/CPAN/

In addition, on systems that do not have GDBM or DB_File installed, SQL is recommended because large, resident catalog databases will use large amounts of memory.

3.2. Perl Modules

The core functions of Interchange can run with a stock Perl, but some features of Interchange (like the administrative interface) require additional modules. If using a UNIX server, use the CPAN module and run the following command:

```
perl −MCPAN −e 'install Bundle::Interchange'
```

This command should install all of the necessary modules, except DBM/DBI. The following modules are strongly recommended:

```
Digest::MD5
MIME::Base64
SQL::Statement
URI::URL
Safe::Hole
Bundle::LWP (contains MIME::Base64 and URI::URL)
GDBM or DB_File (comes with most i386 Perls)
```

The following modules are required:

```
Storable
Business::UPS (comes with Interchange)
```

The following modules are not essential:

```
Term::ReadLine::Perl
Term::ReadKey
```

The DBI module is required if using SQL, along with the appropriate DBD module for your database.

3.3. Operating System

The UNIX Operating System is highly recommended. Mac OS/Finder and Microsoft Windows platforms are not supported. Interchange will run on Mac OS/X.

3.4. Machine

A 400MHz Pentium or equivalent computer with 128MB of RAM is recommended. This machine can serve many catalogs, if that is all it does.

If a site is located on a machine with hundreds of domains, as sometimes happens with low−cost hosting operations, expect some problems. It is difficult to maintain a stable environment with numerous users.

3.5. If Problems Occur

Akopia is interested in making Interchange as reliable and trouble−free as possible and provides support for Interchange. Forward bug reports, questions and comments to the bug reporting system, Bugzilla, found at http://developer.akopia.com.

Interchange offers fee−based technical support through it's Technical Services Group. Help can also be found through the community of developers subscribing to the user group on the Akopia Developer Resource site. Akopia makes full source code of the current version of Interchange available on the Akopia web site :

```
http://www.akopia.com
```

Frequently asked questions and documentation are also available on the Akopia web site. There is a search feature that encompasses the documentation, FAQs, and an archived mailing list.

4. FREQUENTLY ASKED QUESTIONS

4.1. Can you tell me more about MiniVend, Interchange's ancestor?

MiniVend was originally based on Vend created by Andrew Wilcox. Vend was written in the early part of 1995, and the first released version was 0.2. The first version of MiniVend was based on that. It added searching and DBM catalog storage. Subsequent versions took parts from Vend 0.3, especially the VLINK and Server.pm modules, which were adapted to run with MiniVend.

The first release of MiniVend (0.2m7) was on December 28, 1995, and quickly proved to be a more powerful and versatile tool than Vend. Thousands of businesses depend on MiniVend for their e−commerce sites.

4.2. How does Interchange compare with other shopping carts?

Interchange is a comprehensive e−commerce solution that includes shopping cart functionality capable of maintaining databases containing hundreds of thousands of items. Interchange is database−based and can be customized to meet the most complex shopping cart requirements.

Interchange is recommended when a site has:

1. 100 or more items in its catalog.
2. many different catalogs maintained by the same organization.
3. a product offering that will frequently change.
4. a need for programmable product display.
5. a need for complex ordering interaction.
6. soft goods delivery after real−time charge of credit cards.
7. a need for flexible searching and categorization options.

Interchange may not be the right choice when a site has:

1. only a few items.
2. items that do not change frequently.

Consider a commercial product when a site has:

1. a need for power, but users who will interact only through a site building system like MS FrontPage or Netobjects Fusion.
2. a need to interact only through a GUI like Windows and not edit files.

4.3. What type of program is Interchange?

Interchange is not just a script. It is a combination of many programs, Perl modules, and links to other subsystems such as SQL databases, CyberCash, PGP, and the Glimpse search engine.

Interchange is a complete database access and retrieval application. It uses no more memory than a large

database server. It is optimized for catalogs of more than a hundred items and catalogs that expect to change and grow over time.

4.4. Can I use Interchange with my existing static catalog pages?

Yes, but it is recommended that you convert to the data−driven model. Interchange is designed to build pages based on templates from a database. If it is used only as a shopping cart, use a simpler program. It is not difficult to convert existing static pages to Interchange, but maintaining them can be difficult.

To place an order link on a page, use the following:

<A HREF="/cgi−bin/simple/order?mv_order_item=SKU_OF_ITEM">Order!</A>

Replace `/cgi−bin/simple` with the path to your Interchange link.

4.5. Will Interchange run on an ISP?

The majority of ISPs provide some CGI service, and an increasing number run systems that are compatible with Interchange. The catalog configurator for Interchange is designed to figure out many ISP directory setups.

Interchange requires a stable platform. Many ISP servers are heavily loaded, especially low−cost ones. If Interchange is run on a server that is constantly running out of memory and file descriptors, the results will be unsatisfactory.

Virtual servers that do not provide shell access are not usable for Interchange without direct support from the ISP. It can be done on some virtual servers.

4.6. Why is Interchange a daemon?

A daemon is a program that always is running in the background on the system. Interchange runs as a daemon in the normal course of events. Some examples of programs that run as daemons (in most cases) are:

1. HTTP server (Apache, etc.).
2. MySQL, mSQL, Oracle, and other SQL servers.
3. Sendmail.

Interchange takes time to compile and load even on the fastest systems. It has many configuration options, and can serve hundreds of catalogs. If it were to be loaded every time a user accessed it, it would be unusable. The daemon approach allows a rich set of features to be accessed fast. The actual CGI program (VLINK or TLINK) is a small program written in C that communicates with the Interchange daemon.

When a configuration file is modified, Interchange must be informed so that it can reload the information and reconfigure its operation as needed. This is done by either restarting the server or using the *reconfig* script to reconfigure an individual catalog.

4.7. Is Interchange secure?

Interchange uses the Perl Safe.pm module for user−embedded Perl subroutines and conditionals. However, there are several potential problems with credit card number security that can be avoided:

1. Unencrypted credit card numbers stored on disk.
   If Interchange's capability for encrypting credit card numbers or the real−time payment (CyberCash, etc.) interface is not used, there will be unencrypted credit card numbers present in session database files. If the system is the target of a break−in, these numbers would be available to any user ID that can read the session files. This is the reason Interchange defaults to read/write permission for the Interchange user only.
2. Unencrypted credit card numbers sent via email.
   The same things apply for orders sent from email. If it is not encrypted with PGP, it is at risk. The default demo also stores the orders in the file etc/tracking.asc, so check there as well if scrubbing existing credit card numbers from a disk.
3. Running in INET mode from another machine.
   When using INET mode, and the transmission is going from the network to another machine (i.e., not localhost), be concerned about which wires the SSL−encrypted data is traveling through. The server should be behind a firewall, firewall router, or at least some sort of spoofing−protected filter.
4. Running SQL databases without `WRITE_CONTROL` or other permission blocks.
   It is possible to enter arbitrary SQL in some search definitions. Though Interchange tries to block non−select calls, this is not guaranteed. It is recommended that certain tables be read−only for Interchange, i.e., no insert or update permission. For tables that must be updated, i.e., userdb, transactions, and orderline, use the `WRITE_CONTROL` capability and the `NoSearch` directive to protect them.
5. Running with AllowGlobal set.
   This allows the user programming the catalog to do anything the Interchange user ID can, and also disables the Safe.pm checking for embedded Perl code. It is strongly recommended that global UserTag routines be written instead of using `AllowGlobal`.

4.8. What HTML editors work with Interchange?

None, though by accepting a performance penalty many Interchange tags can be embedded inside of regular HTML. For example:

```
[pragma no_html_parse 0]
 <A HREF="[href minivend_page]">Link</A>
 <SCRIPT LANGUAGE=MV MV="set Action">
     mv_todo=return
     mv_nextpage=your_page
 </SCRIPT>
```

Many Interchange pages will have to be edited by hand. Some HTML editors have a tag like <NOTOUCH></NOTOUCH> that defines regions which should not be wrapped or reformatted by the editor. It is recommended that these tags be used.

In addition, if uploading a page from a Mac or PC, make sure it is uploaded in ASCII (non−image) format. If it is not in ASCII, the Interchange catalog can break.

4.9. How will Akopia support Interchange?

Akopia will:

1. Quickly follow up on cogent bug reports. A bug is a demonstrated fault in the program logic that can be duplicated. A cogent bug report is detailed and concise, and includes HTML/code snippets that demonstrate the problem. Bugs can be entered on Bugzilla at http://developer.akopia.com
2. Take note of faults in the demonstrations. Any fixes will be discussed on the mailing list and may be fixed in the next release version of Interchange.
3. Take note of faults in the documentation and update the next release version of Interchange. Edited replacement text is appreciated. The documentation source is available to see how it is maintained.
4. Respond to some of the well−presented questions that appear on the mailing list.
5. Try to constantly and incrementally improve the FAQ and other supporting information.

4.10. I want to partner with Akopia, or exchange links, etc.

Akopia receives many partnering requests. Keep in mind:

Payment Gateways

There are literally dozens of different companies doing e−commerce payment services on the Web. Akopia has received inquiries from many of them over time. The only way Akopia will consider putting in support for a specific payment gateway is if it is a funded consulting project. This usually costs thousands of dollars and is prohibitive.
Users have integrated Interchange with many different payment gateways, such as CyberCash, Signio, and Authorize.net. Akopia supports CyberCash and Signio directly in Interchange because they are market leaders. Interchange easily integrates user gateways.

Directory Listings

Interchange may be listed in software directories but not on mail lists. Akopia will not maintain the listing.

CD−ROM Distributions

Interchange may be distributed unchanged on a CD−ROM. Akopia requests that they be sent a copy of the CD−ROM.

Reselling

Interchange is Gnu GPL licensed and may not be resold. Interchange is freely downloaded. See the <www.gnu.org> Web site for more information.

Akopia Services

While Interchange has many appealing features, the true value of the Akopia solution lies in the comprehensive support provided throughout all stages of an e−business integration. Regardless of the extent of service you require, Akopia's experienced staff is dedicated to ensuring the success of an e−commerce implementation.

Akopia Professional Services

Akopia Professional Services are designed to build the most successful e−business architecture possible. Professional services encompass complete site development and integration, from needs assessment and planning to testing and deployment.
Akopia's Professional Services team is experienced and trained to maximize the power and flexibility of the Interchange platform. Akopia offers professional services for projects of all sizes and scopes, from smaller projects to traditional full lifecycle engagements.
Akopia professional services include:

- Strategic consulting
- Site architecture
- Web development
- Systems integration
- User Interface Design
- Deployment services

Akopia Managed Services

Many businesses would rather not commit valuable time and resources to the challenges associated with operating 24x7 customer−facing systems. With Akopia Managed Services, you can focus on running your business while our expert team ensures your site is performing optimally and your applications are continuously monitored.
Akopia Managed Services include:

- Site hosting
- Application monitoring and support
- Database backup and maintenance
- Service Level Agreements

Akopia Technical Support Services

Akopia provides access to our expert consultants by both telephone and electronic support offerings. A variety of support packages are available, and can be purchased according to your specific needs, such as frequency and response time.

Akopia Training Services

Akopia offers Interchange training options for technical and business professionals. We will customize a training program that is appropriate to your business needs.

- Akopia and Interchange are registered trademarks of Akopia, Inc. All other product names are trademarks or registered trademarks of their respective manufacturers. This version of the document supersedes any and all previous versions.