

# D44

## Life of a Buildmaster – Team Programming using WSAD, CVS, Ant

Peter Wall – Kaiser Permanente

Bryon Kataoka – Commerce Solutions



Las Vegas, NV

Nov. 11 - 15, 2002

# Stating the Problem

---

- Team Programming
- Source Code Control
- Build, Integrate, Test, Deploy
- Context and Scope of this talk...
  - Software Development Process
  - Our starting point: source code
  - Our ending point: deployed application



# Outline and Goals

---

- Introduce Tools: WSAD, CVS, Ant
- Motivation
- How to use Ant
- Theory – use Ant well
- Advanced Ant
  - Opening the source code
  - Customization



# Terminology and Tools

---

- FAQ -what I expect you to ask
- BP -how to realize potential
  
- WSAD (WebSphere Studio Application Developer)
- CVS (Concurrent Versions System)
- Ant (Another Neat Tool)



# Introduction to WSAD

---

- IDE aimed at J2EE programmers
  - EJB 1.1, Unit Test for J2EE, Deployment
- Perspectives
  - Try to show you what you want to see
- Views
  - editor, outline, tasks, console, etc
- Similar to Visual Age for Java??
  - No built-in Repository

# Introduction to CVS

---

- Repository for files
- Multiple versions of each file
- Using CVS
  - checkout, edit, release, tag, branch, etc...
- Files not locked on checkout
- CVS command line:

```
cvs checkout myProject
```



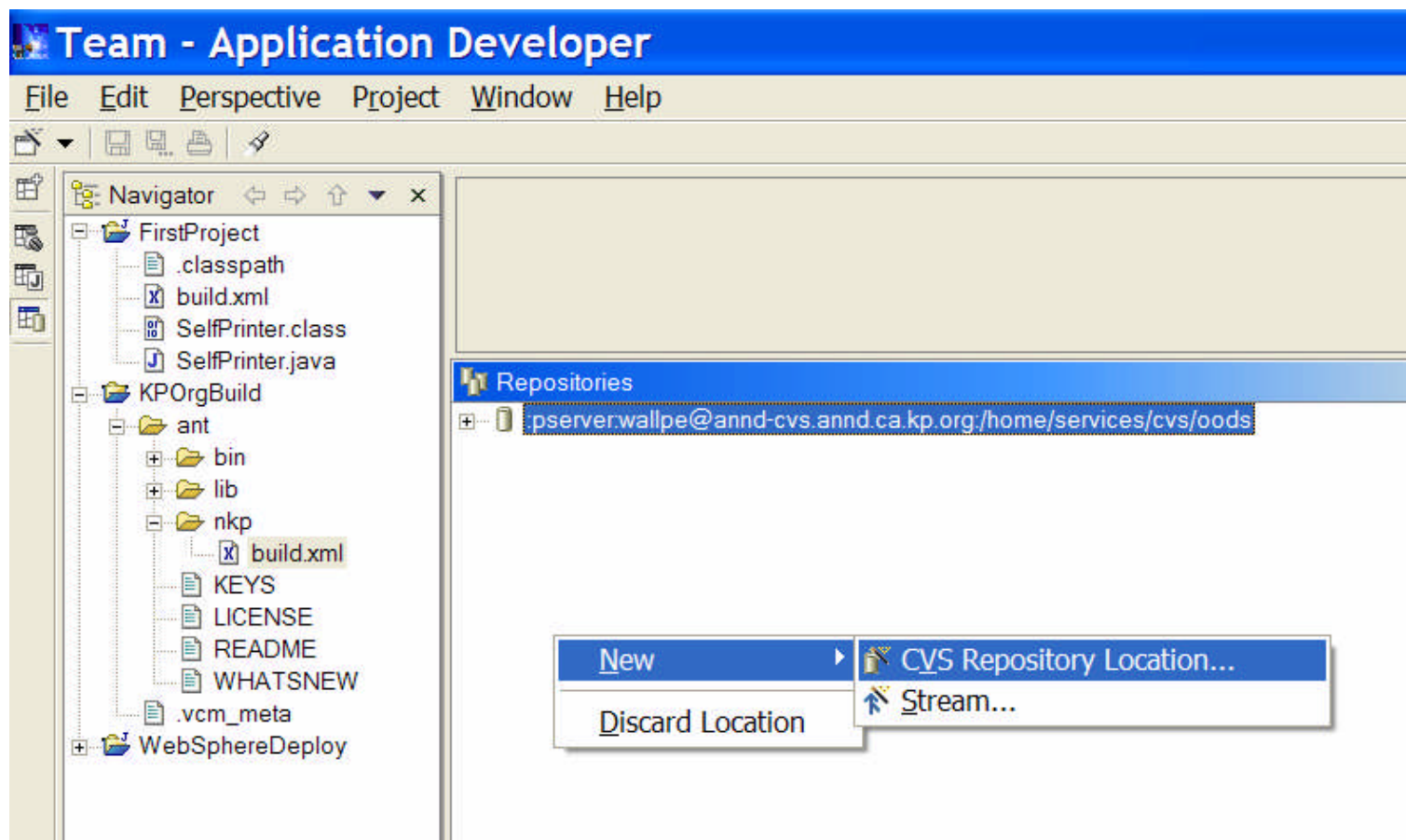
# Connect WSAD to CVS

---

- Team Perspective – Repository View
- New Repository Location
- Connect to CVS – “Add to Workspace”
- Synchronize with Stream
- Release changes CAREFULLY
- Graphical merge
- Source code in multiple projects



# New CVS Repository



# New CVS Repository(2)

**New**

**CVS Repository Location**  
Remember the location of an existing CVS repository.

Connection type: pserver

User name:

Host name:

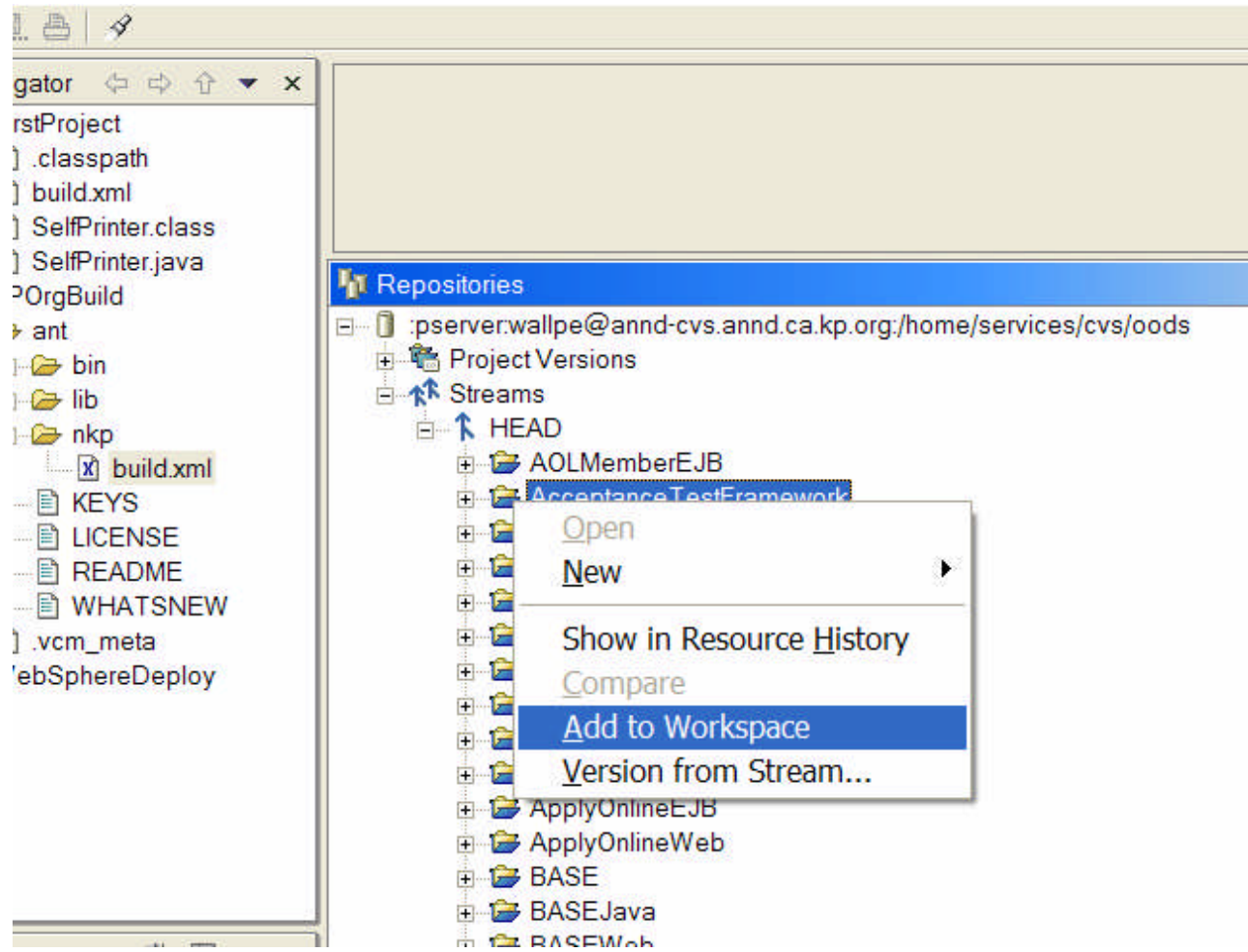
Repository path:

CVS location: :pserver:@:

Validate location on finish

Finish Cancel

# Add to Workspace



# Synchronize with Stream

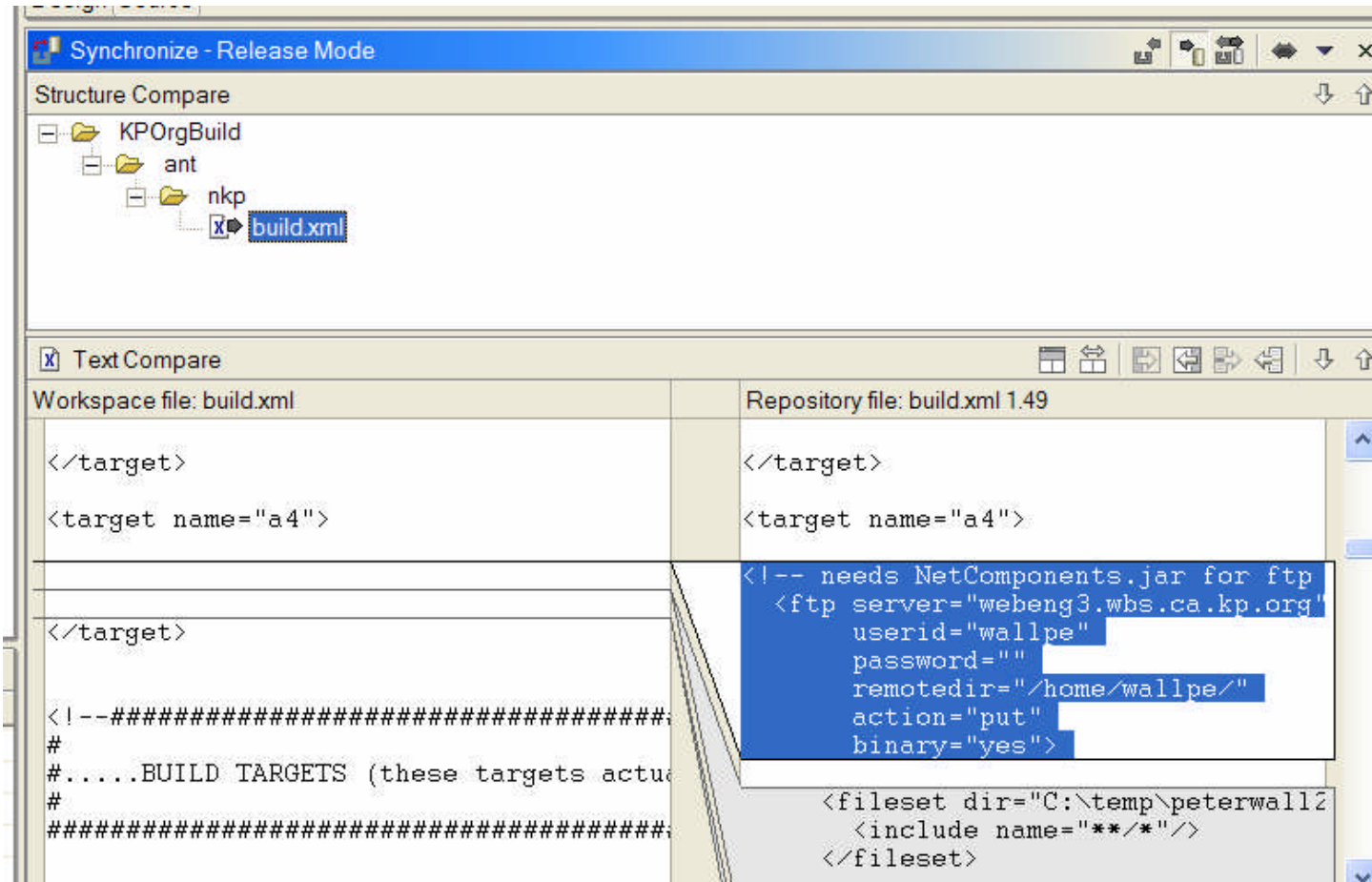
The screenshot shows the IBM WebSphere IDE interface. On the left, a project tree displays the structure of 'KPOrgBuild', including folders like 'ant', 'bin', 'lib', and 'nkp', and files like 'build.xml', 'KEYS', 'LICENSE', 'README', 'WHATSN...', and '.vcm\_meta'. The 'build.xml' file is selected, and a context menu is open over it. The menu items include 'New', 'Go To', 'Open', 'Open With', 'Copy', 'Move', 'Rename', 'Delete', 'Add Bookmark', 'Refresh From Local', 'Run Ant...', 'Run on Server', 'Validate XML File', 'Team', 'Compare With', 'Replace With', 'Apply XSL', and 'Properties'. The 'Team' menu item is expanded, showing a sub-menu with 'Synchronize with Stream' and 'Show in Resource History'. The 'Synchronize with Stream' option is highlighted. Below the project tree, a 'Properties' window shows the following details for 'build.xml':

Property	Value
edita...	true
last ...	9/14/02 6:29 ...
name	build.xml
path	/KPOrgBuild...
size	82339

At the bottom of the IDE, the status bar shows the path 'KPOrgBuild/ant/nkp/build.x'.



# Release, Merge



# What is Ant?

---

- “A Java-based build tool”
- FAQ: Is it like Make? (wait a few slides...)
- Building an application
- Build tools automate:
  - SCCS, compile, package, test, deploy, file i/o...
- Leverage benefits of Java
  - Cross-platform, scalable, extensible, etc.



# “Hello World” in Ant

---

- Develop Ant = write build.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
```

```
<project name="hello" default="say.hello" basedir=".">
```

```
  <target name="say.hello">
```

```
    <echo message="Hello World"/>
```

```
  </target>
```

```
</project>
```



# So far...

---

- WSAD to write code
- Hook it up to CVS where code lives
- Use Ant as the build tool
  - Give it raw material (code from CVS)
  - Tell it what to do (with build.xml)
- Next: A Build Process? Using these tools?

# Motivation - Build Process

Your experience with a repeatable process?

System Architect  
Team Lead  
Project Manager  
Build Engineer



Developer  
UI design  
HR Manager  
CEO

- More to come – Theory

# Motivation - CVS

---

- Standard – other software expects CVS
- Ease of use
  - As a client
  - As an administrator
- Reasonable price
- FAQ: But my SCCS can do X, Y, Z ...
  - Answer: Yes Ant can do it too!



# Motivation - WSAD

- JAVAPro Readers' Choice 2002 –  
    “Best Java Deployment Tool” - WINNER  
    “Most Valuable Product” - WINNER
- Supports J2EE:  
    EJB, JSP, HTML, servlet, web services,  
    XML from DTD, WAR, EAR, code  
    generation, local&remote testing...
- Team Development



# Motivation - Ant

"If you know Make, don't tell anyone."

- Assume we need a Build Process
- Other build tools
  - Make (Jam, Cons)
  - Shell based, not cross-platform
- James Duncan Davidson vs. The Makefile
- Easy to learn / extend Ant with Java classes
- Ant can execute any shell command/script
- FAQ: Can Ant do X, Y, Z ... ?

# Progress – 2/5

---

- ~~Introduce Tools: WSAD, CVS, Ant~~
- ~~Motivation~~
- How to use Ant ←
- Theory – Best Practice
- Advanced Ant

# Using Ant – Getting Started

---

- <http://jakarta.apache.org/ant/index.html>
- lib – core jar files
- bin – ant.sh, ant.bat
- set JAVA\_HOME and ANT\_HOME
- PATH = \$PATH + \$ANT\_HOME/bin
- **ant -version**

# Project, Target, Task

- "Hello World" revisited...

## An Analogy

*.java
Java Class
Method
Objects
Java

build.xml
Ant Project
Target
Tasks
Ant

# Running Ant

---

- BP: Run from command line
  - Simulates other environments
- `ant [options] [target [target2 [target3]...]]`
- Default Target
- You CAN pass in parameters
  - **`-Dproperty=value`**

# Core Tasks, Optional Tasks

---

- Core tasks for free
- Add .jars to Ant/lib to “import” Optional
- Point a browser to Ant docs
- Optional Tasks: They tell you what to do

# Ant Properties

---

- Remember about passing parameters?
- Property ~ String Variable (can't modify)
- Where set? xml, command, .property

```
<property name="build.home"  
          value="C:\Documents and Settings\peterw  
<echo message="working in ${build.home}"/>
```

- BP: Use <property>

# Ant Datatypes

---

- Patternset
- Fileset
- Path
  
- Numerous other types so:  
BP: Know Ant Datatypes (skim at least)

# Ant Datatypes - Patternset

- Use patterns to include/exclude
- Similar to regular expressions
  - \* matches 0 or more characters
  - ? Matches 1 character
- Examples: `**/CVS/*`      `**/test/**`
- Default Excludes: `**/*~`      `**/CVS/**`

# Ant Datatypes - Fileset

```
<fileset dir="{myproject.src}"  
  casesensitive="no" >  
  <patternset id="non.dummy.sources" >  
    <include name="**/*.java"/>  
    <exclude name="**/*dummy*" />  
  </patternset>  
</fileset>
```

- FAQ: How did you know the structure?

# Ant Datatypes - Path

```
<path id="my.compile.classpath">  
  <pathelement path="{was.lib}/j2ee.jar"/>  
  <pathelement path="{another.classpath}"/>  
  <pathelement location="{classes}"/>  
  <fileset dir="lib">  
    <include name="**/*.jar"/>  
  </fileset>  
</path>
```



# Flow of Control

---

```
<target name="delegate"  
  depends="get.source.files">  
  <antcall target="chmod.src">  
  <ant dir="${build.subproject}"/>  
</target>
```

- Execute {if | unless} property has been set

# Tasks: jar, war, ear, zip

- Package an application – J2EE Aware

```
<ear earfile="${build.dir}/app.ear"  
  appxml="${metadata}/application.xml">  
  <fileset dir="${build.dir}"  
    includes="*.jar,*.war"/>  
</ear>
```



# Tasks: java, exec...

---

- If Ant can't do what you want...
- Execute a java class - `<java>`
  - Specify args, classpath, new jvm, maxmemory
- Execute a shell command - `<exec>`
- Extend Ant...

Copy, delete, mkdir, javadoc, unjar, ftp, telnet

Email, validate xml, ejb deployment desc., ...

# JUnit

---

- What is JUnit?
- `<junit>` is an Optional Task
- Put `junit.jar` in `Ant/lib`
  
- Too much xml?
- What does JUnit task look like?
- More important: Why should we use it?



# Progress – 3/5

---

- ~~Introduce Tools: WSAD, CVS, Ant~~
- ~~Motivation~~
- ~~How to use Ant~~
- Theory – Best Practice ←
- Advanced Ant

# Theory – Best Practice

---

- XP - JUnit
- Continuous Integration
- Ant - best practice



# eXtreme Programming

---

- Kent Beck – Object Oriented Design
- Lightweight Process
- Planning
- Pair Programming
- Testing (JUnit) – write tests first?
  - complete tests, simple code, intent of code
- Continuous Integration



# Continuous Integration

---

- Martin Fowler & Matthew Foemmel
- Find bugs as fast as possible
- Build often! or else Exponential Effort
- Define a successful build
- Automate build/test/deploy
- Master build for project



# General Ant Best Practice

---

- Implement cont. integration (or get close)
- Automate – don't hack (config files...)
- Control your builds, execution
- Label & Monitor builds – send emails
- Non-Software, infrastructure, resources,  
share resources, restarting server, etc
- Decisions (next), then Customize

# Ant Best Practice - Decisions

---

- Who develops your build process?
- Make the build.xml readable?
- Multiple build.xml ?
- Non-Software - higher level build process
- Use Ant for 100% ?
  
- BP: Knowledge leads to Quality

# Progress – 4/5

---

- ~~Introduce Tools: WSAD, CVS, Ant~~
- ~~Motivation~~
- ~~How to use Ant~~
- ~~Theory – Best Practice~~
- Advanced Ant ←

# Advanced Ant

---

- Custom Tasks
- How Ant works – open the source
- Deployment – an example



# Custom Tasks

---

- Write 1 java class – EASY!
- Extend `org.apache.tools.ant.Task`
- xml attribute = java instance variable
- **public void setAttribute()**
- **public void execute()**  
**throws BuildException{**

# Custom – SimpleTask.java

---

```
public class SimpleTask extends Task {  
    private String msg;  
  
    public void execute() throws BuildException  
    {  
        System.out.println(msg);  
    }  
  
    public void setMessage(String msg) {  
        this.msg = msg;  
    }  
}
```



# Custom Task - XML

---

```
<taskdef name="mytask"  
         classname="com.SimpleTask"/>
```

```
<target name="main">  
    <mytask message="Hello World"/>  
</target>
```



# How does Ant do it?

---

- Parse build.xml
- Create Java objects
  - org.apache.tools.ant.Project
  - org.apache.tools.ant.Target
  - org.apache.tools.ant.Task
  - and so on...
- Java Introspection

# Example: Deployment

---

- Ant tasks
  - telnet, ftp, copy, exec
- Optional Tasks
  - Weblogic
- WebSphere Application Server
  - more difficult...
  - console vs. wscp



# WebSphere Control Program

- Wscp
- Command-line admin tool (for admin db)
- define, configure, manage
- Based on Tcl, JACL
- Custom Task builds string for  
`com.ibm.ejs.sm.ejscp.wscpcommand.WscpCommand`  
`com.ibm.ejs.sm.ejscp.wscpcommand.WscpResult`



# Review

---

- WSAD, CVS, Ant
- Motivation
- Usage
- Theory
  - XP, Continuous Integration, Repeatable Process
- Customization
- QUIZ: Does Ant have a DTD?

# Resources – Keep Learning!

---

- <http://jakarta.apache.org/ant/manual/index.html>
- ant-user mailing list
  
- "Java Development with Ant"  
Hatcher, Loughran
- "WebSphere Application Server Bible"  
Kataoka, Ramirez, Sit



# Contact Info

---

- Peter Wall  
Kaiser Permanente  
Peter.V.Wall@kp.org  
510-627-2632
- Bryon Kataoka  
Commerce Solutions  
bkataoka@commercesolutions.com  
707-773-1198

