Part 1

Summary

This report describes an automated build system implemented for the Eclipse ECF project. The system uses cruisecontrol and ant; it runs on an OSU machine. It checks out code from the Eclipse repository, performs a headless build, and uploads the files to the ECF downloads directory on dev.eclipse.org. In addition, extra builds are performed from a repository located on another OSU machine and made available from a web site hosted by that machine. The report also describes how the automated build system can be remotely administered and suggests future work.

Contents

Summary	1
The Goal	
Defining the Goal	3
The Zeroth Stage	3
The First Stage	4
Download and install CruiseControl.	5
Download and Install Ant	5
Testing that Ant Works	
Testing that Cruisecontrol Works	
Getting the Build under Cruisecontrol	6
Starting Up CruiseControl	7
The ECF Builds	
Build Intervals and Quiet Periods	7
The ECF Daily Build.	9
<u>cc-build.xml</u>	9
Sending Email	
Uploading Files	14
Passphraseless Key Authentication.	
Making and Using the Key-Pair on Linux	

Using the Key-Pair on Windows	15
Uploading Daily Files	18
Making Files Available on eclipse.org	19
Keeping a Week's Worth of Dailies	21
More Detail on the Daily Builds	
Second Stage	
Setting Up the Cruisecontrol Web Reporting Tool	
Using the Web Reporting Tool Remotely	22
Getting a Remote Desktop	
Getting a Remote Browser.	
Third Stage	31
Fourth Stage	

The Goal

Our goal is to have an automated build system for the Eclipse ECF project.

Defining the Goal

What do we mean by an automated build system? I'm not going to try to describe our entire vision and then show it in operation. It has too much plumage for that.

Rather I'll describe our automated build system in stages. This reflects reality more closely because that's how we implemented it. In fact, some of the later stages have not been implemented.

The Zeroth Stage

In this stage we just wanted to understand the key terms.

PDE Builds. Just some background information about PDE builds. We're building ECF, which is an Eclipse pug-in. This is a called a PDE build.

The most straightforward way is to build the plug-in from within the Eclipse IDE. But we want to be able to build from the command line. We don't want the Eclipse IDE to pop up. Once we can build from the command line, we can put the build in a script and then automate the script.

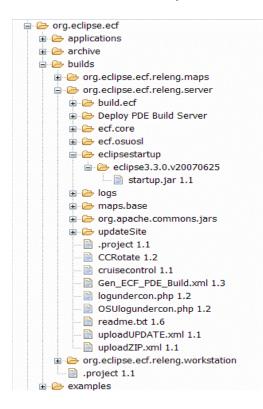
Headless Builds. Building from the command line is called a *headless build*. There is an important caveat. A headless build requires the file startup.jar which is no longer part of the Eclipse SDK distribution. So we grabbed an older one and put it in our own section of the repository.

We are using the startup.jar from Eclipse 3.3.0.

Server vs. Workstation Builds. Notice that our repository makes a distinction between server and workstation builds. This is a distinction that has ceased to be important; we use the server build exclusively now.

The workstation build was one that we could start and run from the command line. You would provide arguments on the command line; but it turns out that you can do the very same thing with the workstation build.

Here's a figure that shows how our repository is structured. The repository is /cvsroot/technology/org.eclipse.ecf.



The First Stage

Here are the characteristics of our first stage.

- We do our builds on a machine in the opensource lab at OSU in Corvallis, Oregon called **ecf2**. It runs Suse 10.2 Linux.
- Our source code is in the Eclipse repository.
- We run cruisecontrol on ecf2. This cruisecontrol checks for changes in the repository every 30 minutes. If there is a change it performs a build. We call this the *auto build*. It sends out email to the list ecf-build. We do not save the auto builds.
- Cruisecontrol also performs a daily build if it detects a change in the repository. We upload the daily build to dev.eclipse.org, and it is available from our ECF web page. (Later we decided to perform a daily build whether or not there was a change in the repository.)

We chose cruisecontrol for no special reason. It's the warhorse of automated builds, but it is showing its age. We are certainly open to trying out other newer systems. We are dependent on cruisecontrol at this time, and so if we try out other methods, we'd have to do it as a prototype and make the switch after we determine it is working the way we want.

Here is how we implemented the first stage.

Download and install CruiseControl.

2.7 was the latest release when we started. There is now a 2.7.1, but we have not upgraded. Our first installation used 2.6.2.

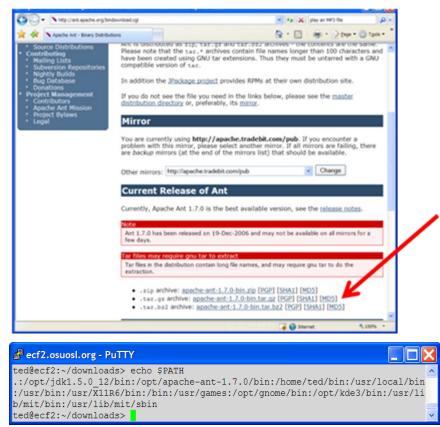
Go to <u>http://cruisecontrol.sourceforge.net/download.html</u> and download the file cruisecontrol-src-2.7.zip. Unzip it to /opt. Then, go into the directory /opt/cruisecontrol-2.t/main and run the script build.sh.

You get a lot of screen output that lasts for a couple of minutes. A dist directory is created under main. To run cruisecontrol, run the script cruisecontrol.sh under main/bin.

Download and Install Ant

The ant that came with Suse 10.2 (downloaded with YaST) did not work for me. Most of it did, but the **scp** authentication part did not. However, the ant I downloaded from the ant site did work. It was a newer version too.

Go to <u>http://ant.apache.org</u>. Click on Download->Binary Distributions.. Then, untar the file into /opt, and put /opt/apache-ant-1.7/bin in the path.



You can test by creating an antfile and running it. I liked to test ant by trying out a sample project from the book *Pragmatic Project Automation* by Mike Clark. I'm going to skip the details of that test here, but it's worth saying something. So here goes.

Testing that Ant Works

You make an antfile; I called it cc-build.xml like Mike Clark. You start it up as

```
ant -buildfile cc-build.xml
```

Note that -f and -file perform the same operation as -buildfile. The file cc-build.xml can be very simple. Note that the example below has one project that runs the target called build. It deletes a directory called dms (this stands for Document Management System and comes from Mike Clark's book). This checks out (from CVS) the project dms. It uses the repository defined in the cvsroot environment variable.

```
<project name="cc-build" default="build" basedir="/home/ted/builds/checkout">
        <target name="build">
            <delete dir="dms" />
            <cvs command="co dms" />
            <ant antfile="build.xml" dir="dms" target="test" />
        </target>
</project>
```

The file cc-build.xml calls the target test in another antfile called build.xml. I'm not going to reproduce the details of build.xml here, but the target test depends on the target compile-tests, which depends on the target compile. And the target compile calls the ant task <javac/> which does the compilation.

Testing that Cruisecontrol Works

That last build was run under ant, not cruisecontrol. Let's just see what's necessary to get it under cruisecontrol.

Getting the Build under Cruisecontrol

You need a file called **config.xml**. Well, that's the default name, and we might as well use it. Here's a minimal **config.xml**.

```
<cruisecontrol>
  <project name="dms" buildafterfailed="false">
        <listeners>
            <currentbuildstatuslistener
            file="logs/dms/DMScurrentbuildstatus.txt" />
        </listeners>
        <modificationset quietperiod="30">
            <cvs localworkingcopy="checkout/dms" />
            </modificationset>
        <schedule interval="300">
            <ant buildfile="cc-build.xml" target="build" />
        </schedule>
        </project>
        </pro>
```

Note the <listeners/> element. This replaces the obsolete

```
<bootstrapper>
    <currentbuildstatusbootstrapper/>
</bootstrapper>
and
<publisher>
    <currentbuildstatuspublisher/>
</ publisher >.
```

Starting Up CruiseControl

Start up cruisecontrol by just running the cruisecontrol.sh script. Give it options to identify the config file, specify the port for the JMX console, and specify the port for remote method invocation (rmi). The values below are the default values, but I like to list them anyway. More on these ports later.

```
/opt/cruisecontrol-2.7/main/bin/cruisecontrol.sh -configfile config.xml -port
8000 -rmiport 1099'
```

The ECF Builds

We do an auto build and a daily build. Each is a separate cruisecontrol project.

Each of these builds consists of three zip files More about what these zip files are and how to make them later. Right now, think about how to configure the scheduling.

Build Intervals and Quiet Periods

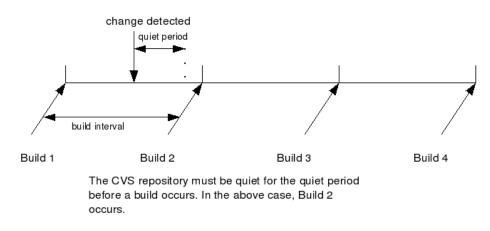
We want to set cruisecontrol to consider an auto build every 30 minutes. Note the sample config.xml listed above and the elements <modificationset/> and <schedule/>.

The <modificationset/> is just what you think it is. It's the location of the checked-out files that you want to build. Cruisecontrol monitors these files to see if they change in the CVS repository. The <schedule/> element specifies how often cruisecontrol considers doing a build. The units are seconds ... 30 minutes are 1800 seconds, so for auto builds, set the interval to 1800.

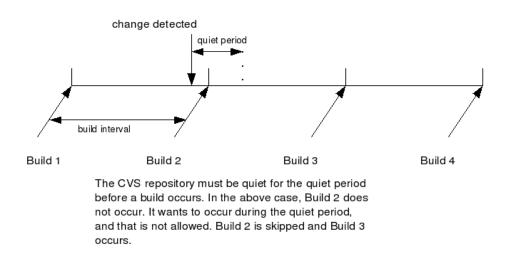
```
<modificationset quietperiod="300">
<cvs localworkingcopy="${localcopy}" />
</modificationset>
<schedule interval="1800">
```

What about the quietperiod? The CVS repository must be quiet for the quietperiod before a build is attempted. If it is not, then the next scheduled build is skipped. The next two figures illustrate what I mean.

Consider a series of builds ... Build 1, Build 2, Build 3, Build 4, each separated by the build interval. In the first figure, the quiet period starts after a CVS change is detected and before the build interval between Build 1 and Build 2 ends. Hence, Build 2 occurs.



In the next figure, the build interval between Build 1 and Build 2 ends before the quiet period terminates. Hence, Build 2 does not occur.



The ECF Daily Build

The last section showed a snippet from **config.xml** for the ECF auto build. Here are the equivalent lines for the Daily Build.

```
<modificationset quietperiod="300">
  <filesystem folder="/opt/ECFBuilds/daily_kick" />
  <cvs localworkingcopy="${localcopy}" />
  </modificationset>

<schedule>
  <ant buildfile="cc-build.xml"
      time="1500"
      target="ecf.copy" >
      <property name="mapVTag" value="HEAD" />
      <property name="feature" value="ecf.core" />
      <property name="feature" value="DAILY" />
      <property name="buildIdentifier" value="DAILY" />
      <property name="genFVSuffix" value="false" />
      </ant>
</ant>
```

Notice that there isn't an interval attribute for the <ant/> element. Rather there is a time attribute. It specifies the time based on a 24-hour clock; 1500 is 3PM.

Note the properties. These pass values to the antfile cc-build.xml. If you wanted to run the build from the command line (not through cruisecontrol), use the -D option on the ant command line to pass these values.

```
ant -DmapVTag=HEAD -Dfeature=ecf.core -DbuildIdentifier=DAILY -DbuildType=D
-DgenFVSuffix=false -buildfile=cc-build.xml ecf.copy
```

Why is the target ecf.copy and not ecf.core? I'll explain that later. Let's look at cc-build.xml first.

cc-build.xml

The file cc-build.xml executes the <java/> task that performs the headless build. You know, you don't have to perform the headless build through ant. You could invoke java right from the command line and supply the appropriate options.

Here are the key lines in cc-build.xml.

```
<java classname="org.eclipse.core.launcher.Main"
    classpath="${eclipse.home}/startup.jar"
    fork="true" resultproperty="buildResult">
    <arg line ="-application org.eclipse.ant.core.antRunner
        -buildfile build.ecf.xml -logfile logs/${logfile}"/>
    <sysproperty key='feature' value='${feature}'/>
    <sysproperty key='mapVersionTag' value='${mapVersionTag}'/>
    <sysproperty key='buildType' value='${buildType}'/>
    <sysproperty key='buildId' value='${buildId}'/>
    <sysproperty key='buildLabel' value='${buildType}-${timestamp}'/>
    <!-- Zip build folder -->
    <sysproperty key='baseLocation' value='${eclipse.home}'/>
```

```
<sysproperty key='timestamp' value='${timestamp}'/>
<sysproperty key='generateFeatureVersionSuffix' value='${genFVSuffix}'/>
<sysproperty key='forceContextQualifier'
    value='${forceContextQualifier}'/>
</java>
```

What's happening here? Well, you're running the eclipse launcher from the command line. You are launching the application antRunner (this is the ant that's part of Eclipse) and specifying the antfile build.ecf.xml.

Remember, when you did a headless build, you invoked java from the command line or more likely put this java invocation in a script. What we're doing is instead of that script we use the antfile cc-build.xml and then arrange that cruisecontrol call this antfile. So the flow of control looks like the following.

```
cruisecontrol →
config.xml →
cc-build.xml →
java, eclipse, antrunner, build.ecf.xml →
build.xml
```

Sending Email

Cruisecontrol takes care of sending email in case of success or failure. This is controlled through config.xml.

But first we set up a mailing list called **ecf-build**. To subscribe to this mailing list, go to <u>https://</u><u>dev.eclipse.org/mailman/listinfo/ecf-build</u>. Click on **ECF Build Archives** to see an archive of mail messages.

🖉 ecf-build In	o Page - Windows Internet Explorer	
00- D	ttps://dev.eclipse.org/mailman/listinfo/ecf-build	🔹 🐓 🗙 headless builds eclipse 🛛 🔎 🗸
🚖 🚓 😥	🍘 Main Page - CodeSprint 🛛 🕵 Safely Storing Business 🎑 Build and Test Automati 🙋 ecf-build Info Page 🛛 🗙	🟠 • 🚮 • 🖶 • 🔂 Bage • 🕥 Tools • 🎽
eclipse	contact legal	Search: Go
	HOME COMMUNITY MEMBERSHIP DOWNLOADS PROJECTS ABOUT US	
Navigation	¥ Mailing list: ecf-build	ecf-build
<u>Committers</u> Newsgroups	ECF build results list	ecf-build list run by mailman-owner
▶ <u>Bugs</u>	About ecf-build	at eclipse.org ecf-build administrative interface
<u>Articles</u>	ECF build results list	(requires authorization) <u>Overview of all eclipse org mailing</u> lists
	To access a web archive of this list, visit the <u>ecf-build Archives</u> .	
	Or you can subscribe to this list's RSS feed 🗟	
	Using ecf-build	
	To post a message to all the list members, send email to <u>ecf-build@eclipse.org</u> .	
	You can subscribe to the list, or change your existing subscription, in the sections below.	
	Subscribing to ecf-build	
	Subscribe to ecf-build by filing out the following form. You will be sent email requesting confirmation, to prevent others from gratulously subscribing you. If you have SPAM filters, or if e-mail sent to you needs to be whitelisted or approved, your subscription may fail. Please disable these tools until your subscription is completed. This is a hidden list, which means that the ist of members as available only to the ist administrator.	
	Your email address:	
	Your name (optional):	
	You may enter a privacy password below. This provides only mild security, but should prevent others from messing with your subscription. Do not use a valuable password as it will occasionally be emailed back to you in cleartext.	
	17 1 11 1 1 11 11 11 11 11 11 11 11 11 1	×
		🏹 🎱 Internet 🔍 100% 🔹

Here is a sample mail message. If you click on the link in the message, right now you just get a link to the ECF downloads page (where the Daily Builds are available). Our intent is to provide

more build statistics on this page in the future.

Ø ecf-build Archive (Date View) - Windows Inte	ernet Explorer 📃 📃 🔀
The second seco	👻 🗲 🗙 headless builds eclipse 🛛 🔎 🗸
💢 🞲 🗄 🔻 « 🕵 Saf 🤙 Buil 🖉 e X »	🟠 • 🔝 • 🖶 • 📝 <u>P</u> age • 🎯 T <u>o</u> ols •
ecf-build Archive (Date View)	
<u>Thread Index</u>	
[First Page] [Prev Page][Next Page] [Last Page]	
October 30, 07	
21:39 [ecf-build] osu build.90 Build Successful	tedkubaska
21:31 [ecf-build] ecf build.157 Build Successful	tedkubaska
20:56 [ecf-build] ecf build.156 Build Successful	tedkubaska
20:47 [ecf-build] osu build.89 Build Successful	tedkubaska
October 26, 07	
19:18 [ecf-build] osuDaily build.32 Build Success	s <mark>ful</mark> tedkubaska
18:18 [ecf-build] ecfDaily build.63 Build Successf	ful tedkubaska
11:15 [ecf-build] osu build.88 Build Successful	tedkubaska
11:08 [ecf-build] ecf build.155 Build Successful	tedkubaska
03:24 [ecf-build] ecf build.154 Build Successful	tedkubaska
03:15 [ecf-build] osu build.87 Build Successful	tedkubaska
01:35 [ecf-build] osu build.86 Build Successful	tedkubaska
01:06 [ecf-build] ecf build.153 Build Successful	tedkubaska
00:34 [ecf-build] osu build.85 Build Successful	tedkubaska
00:27 [ecf-build] ecf build.152 Build Successful	tedkubaska
October 25, 07	
23:41 [ecf-build] ecf build.151 Build Successful	tedkubaska
23:34 [ecf-build] osu build.84 Build Successful	tedkubaska
22:59 [ecf-build] ecf build.150 Build Successful	tedkubaska
22:43 [ecf-build] osu build.83 Build Successful	tedkubaska
05:18 [ecf-build] osuDaily build.31 Build Fixed	tedkubaska
October 23, 07	todkubacka
16:53 [ecf-build] osuDaily Build Failed 16:13 [ecf-build] osuDaily build.30 Build Success	tedkubaska
To To Tect-Dulid) Osubally Dulid.30 Dulid Success	
Done	🗃 🌍 Internet 🔍 100% 👻

A typical mail message looks like the following:

Wessage Developer A rad Rey	□ □ □ □ ▲ ♥ =	[ecf-build] ecf build.157 Build Successful - Message (HTML)	_ = ×
Repty Repty <td< th=""><th>Message Developer</th><th></th><th>0</th></td<>	Message Developer		0
To: ed-build@edpse.org Cc Subject: [ed-build]edbuils75 Build Successful Message] [D_ATT00327.bt (2418) View results here -> http://ecf2.osuosl.org/logundercon.php?log=log20071030181453Lbuild.157 BULD COMPLETE - build.157 Date of build: 10/30/2007 18:14:53 Time to build: 4 minutes 43 seconds Last changed: 10/30/2007 10:45:30 Last log entry: changed javadoc generation markup Errors/Warnings: (11) Click on the results link to get copies of the latest Daily builds 	to All Folder * Rule A	Cher Block ⊇ Not Junk Categorize Follow Mark as Up - Unread 2 kelet - kelet	
cc Subject: [cd+besage] Classing Classing Wiew results here -> http://ccf2.osuosl.org/logundercon.php?log=log20071030181453Lbuild.157 Date of build: 10/30/2007 18:14:53 Time to build: 4 minutes 43 seconds Last log entry: changed javadoc generation markup Errors/Warnings: (1) Click on the results link to get copies of the latest Daily builds	From: ecf-build-bounces@eclipse.org on behalf o	f tedkubaska@comcast.net Sent:	Tue 10/30/2007 6:20 PM
Subject [ed+build] ef build[37 Build Successful IMMessage ①ATT00327.bt (241 B) View results here -> http://ecf2.osuosl.org/logundercon.php?log=log20071030181453Lbuild.157 BUILD COMPLETE - build.157 Date of build: 10/30/2007 18:14533 Time to build: 4 minutes 43 seconds Last changed: 10/30/2007 10:45:30 Last log entry: changed javadoc generation markup Errors/Warnings: (11) Click on the results link to get copies of the latest baily builds 			
View results here -> http://cdf2.osuosl.org/logundercon.php?log=log20071030181453Lbuild.157 BUILD COMPLETE - build.157 Date of build: 10/30/2007 18:14:53 Time to build: 4 minutes 43 seconds Last changed: 10/30/2007 14:530 Last log entry: changed javadoc generation markup Errors/Warnings: (11) Click on the results link to get copies of the latest Daily builds			
BUILD COMPLETE - build.157 Date of build: 10/30/2007 10:14:53 Time to build: 4 minutes 43 seconds Last changed: 10/30/2007 10:45:30 Last og entry: changed javadoc generation markup Errors/Warnings: (11) Click on the results link to get copies of the latest Daily builds	Message ATT00327.txt (241 B)		
BUILD COMPLETE - build.157 Date of build: 10/30/2007 10:14:53 Time to build: 4 minutes 43 seconds Last changed: 10/30/2007 10:45:30 Last og entry: changed javadoc generation markup Errors/Warnings: (11) Click on the results link to get copies of the latest Daily builds	View results here > http://ecf2.osur	sl.org/logundercon.php2log=log20071030181/531.build 157	-
Time to build: 4 minutes 43 seconds Last changed: 10/30/2007 10:45:30 Last log entry: changed javadoc generation markup Errors/Warnings: (11) Click on the results link to get copies of the latest Daily builds	view results here -> <u>http://eci2.osuc</u>		
Last changed: 10/30/2007 10:45:30 Last log entry: changed javadoc generation markup Errors/Warnings: (11) Click on the results link to get copies of the latest Daily builds		10/30/2007 18:14:53	
Last log entry: changed javadoc generation markup Froms/Warnings: (11) Click on the results link to get copies of the latest Daily builds			
Errors/Warnings: (11) Click on the results link to get copies of the latest Daily builds			
Click on the results link to get copies of the latest Daily builds 		changed Javadoc generation markup	
<pre></pre>		t copies of the latest Daily builds	
build.environment.properties: buildingOSGi = true build.environment.properties: celipse.home = /opt/celipse cc-build.ml: logfile = ecf.core-A20071030-1815.log buildIdentifier: false buildIdrentifier: false buildIdrentifier: false buildIdrentifier: false buildIdrentifier: false buildIdrentifier: buildIdrentifier cc-build.ml: buildResult = 0 Unit Tests: (0) No Tests Run This project doesn't have any tests Modifications since last successful build: (1)			
build.environment.properties: eclipse_home = /opt/eclipse cc-build.xml: logfile = ecf.core-A20071030-1815.log buildidentifier: false buildidentifier: core.A20071030-1815.log Unit Tests: buildResult = 0 Unit Tests: (0) No Tests Run This project doesn't have any tests Modifications since last successful build: (1)			
cc-build.xml: logfile = cf.core-A20071030-1815.log buildfdentifier: false buildfd: 20071030-1815 buildfd: 20071030-1815 buildfd: 20071030-1815 buildfd: 20071030-1815 buildfd: 20071030-1815 buildfd: Comparison cc-build.xml: buildResult = 0 Unit Tests: (0) No Tests Run This project doesn't have any tests Modifications since last successful build: (1)			
buildId: 20071030-1815 buildType: A cc-build.xml: buildResult = 0 Unit Tests: (0) No Tests Run This project doesn't have any tests Modifications since last successful build: (1)			
buildType: A cc-build.xml: buildResult = 0 Unit Tests: (0) No Tests Run This project doesn't have any tests Modifications since last successful build: (1)			
cc-build.xml: buildResult = 0 Unit Tests: (0) No Tests Run This project doesn't have any tests Modifications since last successful build: (1)			_
Unit Tests: (0) No Tests Run This project doesn't have any tests Modifications since last successful build: (1)	buildrype:	A	
No Tests Run This project doesn't have any tests Modifications since last successful build: (1)	cc-build.xml:	buildResult = 0	
	No Tests Run This project doesn't have any tests Modifications since last successfu	l build: (1)	

If you click on the link next to "View results here," you get the following:



To set up cruisecontrol to send mail to our mailing list, add the following to config.xml.

```
<!-- PROJECT ECF This is the Auto build -->
<project name="ecf" buildafterfailed="false">
```

```
<listeners>
       .
     </listeners>
     <modificationset ...>
     </modificationset>
     <schedule ...>
     </schedule>
     <publishers>
        <htmlemail mailhost="${smtpserver}"
              returnaddress="${mailsender}"
              skipusers="true"
              buildresultsurl="${resultswebpage}"
              css="/opt/cruisecontrol-2.7/docs/cruisecontrol.css"
              xsldir="/opt/cruisecontrol-2.7/reporting/jsp/webcontent/xsl">
           <map alias="ecf-build" address="ecf-build@eclipse.org"/>
           <always address="ecf-build" />
        </htmlemail>
    </publishers>
</project>
```

A couple of comments about \${smtpserver} and \${mailsender} are in order. On ecf2, we define \${smptserver} to be smtp.osuosl.org, which is in fact OSU's smtp server.

On ecf2, what we put in for \${mailsender} is irrelevant; it doesn't even have to be a real mail address. However, when I run cruisecontrol on my own workstation and talk to my own ISP (comcast), I must provide my comcast email address for comcast to accept my mail for transfer. Both \${smtpserver} and \${mailsender} are defined as properties up at the top of config.xml.

```
<property name="smtpserver" value="smtp.osuosl.org" />
<property name="mailsender" value="tedkubaska@comcast.net" />
```

The css and xs1 files are defaults that come with cruisecontrol. Change them if you like.

skipusers is an important attribute. Cruisecontrol has an interesting feature in that it tries to send email to the user that made the cvs change. It constructs the address as

```
<the cvs user's name>@<the value of defaultsuffix>
```

defaultsuffix is an attribute of <htmlemail /> that I haven't shown and don't use. What skipusers does is cause cruisecontrol to ignore this feature. With skipusers set to true, mail is sent only to ecf-build and not to any of the cvs users.

Note that the mail is listed as **<always** />. This means that both success and failure messages are sent. Sometimes you want only failure messages, and you can arrange that as follows.

<failure address="ecf-build" reportWhenFixed="true"/>

When you do that, an email is sent only on a failure and then again when that failure is fixed.

Uploading Files

We want to make our Daily Builds available for download from

http://www.eclipse.org/ecf/downloads.php . We can make it available by uploading the files to a directory on dev.eclipse.org, namely downloads/technology/ecf. Downloads is a link under my home on dev.eclipse.org. Put a file there and in a few minutes, it starts appearing on the mirrors.

PassphraselessKey Authentication

When we first considered uploading files to dev.eclipse.org, we looked at the <ftp /> task. However, for security reasons we preferred to use scp with passphraseless key authentication. We wanted passphraseless because we did not want to embed a password in our scripts.

Making and Using the Key-Pair on Linux

Here's how I made both an rsa and a dsa key. In your home, create and enter the directory .ssh. Issue the command

ssh-keygen -t rsa or ssh-keygen -t dsa

and then enter a return for a password.

In the screenshots that follow I used my own Linux desktop called halibut. They keys I made are not unique to halibut. The keys are ASCII files, and you can look in the public key file and see ted@halibut in it, but that's OK. It still works with tkubaska@dev.eclipse.org.

🗖 ted@halibut: ~/.ssh	_ • ×
Eile Edit View Terminal Tabs Help	
tedêhalibut:-/.ssh\$ ssh-keygen -t dsa	
Generating public/private dsa key pair.	
Enter file in which to save the key (/home/ted/.ssh/id_dsa):	
Enter passphrase (empty for no passphrase):	
Enter same passphrase again:	
Your identification has been saved in /home/ted/.ssh/id_dsa. Your public key has been saved in /home/ted/.ssh/id dsa.pub.	
The key fingerprint is:	
c2:12:bc:73:c6:3e:5c:ef:50:c9:ec:7d:90:97:d7:29 ted0halibut	
ted@halibut:~/.ssh\$ ls	
authorized keys id dsa id dsa.pub known hosts	
ted@halibut:~/.ssh\$ ssh-keygen -t rsa	
Generating public/private rsa key pair.	100
Enter file in which to save the key (/home/ted/.ssh/id_rsa):	
Enter passphrase (empty for no passphrase):	
Enter same passphrase again:	
Your identification has been saved in /home/ted/.ssh/id_rsa.	
Your public key has been saved in /home/ted/.ssh/id_rsa.pub.	
The key fingerprint is:	
al:f3:77:0f:9e:ld:b0:8d:12:51:24:3c:f8:b1:6c:e7 ted@halibut	
ted@halibut:~/.ssh\$ sftp ted@ecf2.osuosl.org	-

You get two keys: a private key (id_dsa) and a public key (id_dsa.pub).

Keep the private key in your .ssh on your source machine, in this case ecf2. Make the private key -rw----- by you; you here is the user that cruisecontrol runs as.

Put the private key in a file called **authorized_keys** in .ssh in your home on the destination machine. Make the file **authorized_keys** -rw----- by you.

With passphraseless key authentication, you can then log onto the destination machine without using a password as follows.

ted@halibut: ~/.ssh	_ - X
<u>File Edit View Terminal Tabs Help</u>	
<pre>ted@ecf2:-/.ssh> ssh tkubaska@dev.eclipse.org Last login: Wed Oct 31 12:22:57 2007 from c-67-189-66-199.hsdl.or.comcast.net tkubaska@node4:-> logout Connection to dev.eclipse.org closed. ted@ecf2:-/.ssh></pre>	

You can also use **scp** to transfer files from **ecf2** to **dev.eclipse.org** without having to supply a password.

Using the Key-Pair on Windows

If you are using Windows with PuTTY, you have to load this key (which on Linux was made as an OpenSSH key) and convert it into PuTTY format. Run puttygen.exe and click on Load.

🚰 PuTTY Key Generator		? 🗙
<u>F</u> ile <u>K</u> ey Con <u>v</u> ersions <u>H</u> elp		
Key No key.		
Actions Generate a public/private key pair		Generate
Load an existing private key file		Load
Save the generated key	Save p <u>u</u> blic key	Save private key
Parameters Type of key to generate: O SSH-1 (RSA)	⊖ssh	-2 <u>D</u> SA
Number of <u>b</u> its in a generated key:		1024

Load the private key id_dsa.

Load private ke	ey:					? 🔀
Look <u>i</u> n:	Dutty		*	G 💋	► 🔝 🏷	
Recent Desktop My Documents	hold id_dsa.ppk id_dsa.ppk id_dsa.pub id_rsa.pub id_rs	<pre> ② PUTTY,HLP</pre>				
	Cile normal	id_dsa			*	0
My Computer	File <u>n</u> ame:					<u>Open</u>
	Files of type:	All Files (*.*)			*	Cancel
		Open as <u>r</u> ead-only				.::

PuTTYgen Notice			
Ų.	Successfully imported foreign key (OpenSSH SSH-2 private key). To use this key with PuTTY, you need to use the "Save private key" command to save it in PuTTY's own format.		

Click OK. Then, click Save Private Key.

PuTTY Key Gene	rator		NAMES OF A STREET OF A STREET	? 🕑
e Key Conversions H	lelp			
Кеу				
Public key for pasting in	nto OpenSSH authoriz	ed_keys file:	<u> </u>	
Fijo3/Zq5FMksELHg6	bAyTgaF50OfHixArPE	oKUh3ImXasbFdfMicd3/+yx3 rP4jLL14aYjNKG6FTkjUeRG DyKMQbkZz6+81IK/vy6/lvHox	TSds3U5TllrxnqJC	
TZDEVkd7zGFII2yP71	lisyk/8G1HCP6Qdh5C	llat5z906Pcfj+s638H410pGGF	Hb6IC6S/CJRDxsIA	*
Key fingerprint	ssh-dss 2048 c2:12:	bc:73:c6:3e:5c:ef:50:c9:ec:7d:	90:97:d7:29	
Key <u>c</u> omment	imported-openssh-k	сеу		
Key p <u>a</u> ssphrase:				
C <u>o</u> nfirm passphrase:				
Actions				
Generate a public/priva	ate key pair		<u>G</u> enerate	
Load an existing privat	e key file		Load	
Save the generated ke	у	Save p <u>u</u> blic key	Save private ke	у
Parameters				
Type of key to generat	e: () SSH-2 <u>R</u>	SA OSSH	-2 <u>D</u> SA	
Number of <u>b</u> its in a gen	erated key:		1024	

Click Yes



Save the file ensuring that you save it with a **ppk** extension. Then, when you use PuTTY to connect, you must first run **pageant.exe**. When you execute pageant, it appears in your toolbar. Click on it icon and Add the private key you made. You must add this private key every time you bring up Windows, but it will stay active during your Windows session.

Uploading Daily Files

When we first started making Daily Builds, we didn't save more than the latest one. So every Daily Build that we uploaded just overwrote the previous one. What we did was add an <onsuccess /> task underneath <publishers />, and in that <onsuccess /> task call the antfile antscp.xml.

```
<!-- PROJECT ECF This is the Daily build -->
```

```
<project name="ecfDaily" buildafterfailed="false">
   <listeners>
    </listeners>
    <modificationset ...>
    </modificationset>
    <schedule ...>
    </schedule>
    <publishers>
       <htmlemail ...>
       </htmlemail>
      <onsuccess>
         <antpublisher
             antscript="/opt/apache-ant-1.7.0/bin/ant"
             antWorkingDir="/opt/build.ecf"
             buildfile="antscp.xml"
             target="deploy" />
      </onsuccess>
    </publishers>
 </project>
```

The file antscp.xml looks as follows. We define source and destination directories. The source directory is on ecf2; the destination directory is on dev.eclipse.org. We also specify the location of the private key.

```
<project name="upload files" default="deploy">
<property name="deploysrc" location="/opt/build.ecf/ecf.output" />
<property name="deploydest"
value="tkubaska@dev.eclipse.org:/home/data/users/tkubaska/downloads/technolo
gy/ecf/dailies" />
<property name="keyloc" location="/home/ted/.ssh/id_dsa" />
<target name="deploy">
<scp todir="${deploydest}"
keyfile="${deploydest}"
cscp todir="${deploydest}"
cfileset dir="${deploysrc}">
<fileset dir="${deploysrc}">
<fileset dir="${deploysrc}">
</fileset dir="${deploysrc}">
</fileset>
</scp>
```

Making Files Available on eclipse.org.

These files are available from http://www.eclipse.org/ecf/downloads.php . I edited

download.php to have an ECF Dailies link.

🖉 ECF Downloads - Windo	ws Internet Explorer		
Http://www.eclips	se.org/ecf/downloads.php	🗸 😽 🗙 ssh-ke	eygen windows microsoft 🔎 🗸
🔶 🕸 🔡 🖌 🏉 Microsoft Win	dows HO 🖉 ECF Downloads 🛛 🗙	🙆 • 🔊 - 🖶	• \therefore Page • \bigcirc Tools • *
	This zip file contains ECF core plugins.		
		Build Date	
	org.eclipse.ecf.core-1.2.0.v20071019-1300.zip 2	007-10-19	
	This zip file contains ECF examples plugins.		
	File	Build Date	
	org.eclipse.ecf.examples-1.2.0.v20071019-1300.zip	p 2007-10-19	
	The following files are the latest ECF Daily builds. ECF code, a daily build is performed, and the zip f download. These zips are more recent than the re thoroughly tested.	files placed here for	
	Daily Builds		
	For daily downloads go to ECF Dailies		
	nstall from zip files:		E
	Close Eclipse (if running) and unzip into your Eclipse C:\eclipse)		
	Launch Eclipse		
	Problems? Check the FAQ and file a bug report.		
ECI	F Source Code		
For	source code access to all ECF plugins, see the ECF c	developer resources page	
ECI	F Extras Plugins at OSU Open Source Lab		
	F 'Extras' plugins are available via our site at the Or urce Lab.	regon State University Open	
		🏹 😜 Internet	🔍 100% 👻 🔡

Click on ECF Dailies and see the following.



Note that on dailies.php, I show several daily builds. When we first started, we had just three Daily zips. So the three zips we had were the sdk, examples, and core zips. The filenames did not include the date and time (those numbers just before the DAILY). And the filenames now have the version number, which is not shown in the old figure above. A typical zip name is org.eclipse.ecf.sdk-2.0.0.v20080620-1600-DAILY.zip.

Keeping a Week's Worth of Dailies

Here's a problem. Our ECF web page needs to show the names of the daily builds and these names depend on what is currently being offered for download and what is being offered changes depending on whether a Daily Build is performed or not. It depends on what is in the downloads directory on dev.eclipse.org.

So when a user accesses the downloads page and clicks on ECF Dailies, we don't want our PHP to have to go and do an ls on dev.eclipse.org. This might result in an inordinate number of hits on dev.eclipse.org and, well, that's just inappropriate.

So what to do? The solution we have is not lilely the best one, but in the interest of true collaboration, I'm going to describe what I did, knowing full well that it's a kludge.

More Detail on the Daily Builds

To modify how our web site presents daily builds for download, you have to first look at how the resulting zip files are moved around.

Remember in config.xml the target for the daily build was ecf.copy. If you look in ccbuild.xml at the target ecf.copy, you see that it depends on ecf.build. So ecf.build gets done first and then ecf.copy happens.

The first thing ecf.copy does is copy the output to a directory called ecf.output on ecf2. It copies the updatesite as well as the zips. Then, it puts the daily builds in ecf.dailies. Then it does an ls -1 on ecf.dailies and puts the output in filelist.txt. Note that this is an ls -1 (one not el). The file filelist.txt is in the workspace area that eclipse uses for its checkouts. Then, cc-build.xml calls the antfile antfilelist.xml.

The antifilelist.xml checks filelist.txt into www/ecf under the repository dev.eclipse.org:/cvsroot/org.eclipse, which is where our web site is stored.

Then, remember that the link ECF Dailies actually brings up dailies.php, which has in it the following lines.

```
<?php
$files=file("filelist.txt");
rsort($files);
foreach ($files as $file) {
  echo '<tr> 
  <a href="http://www.eclipse.org/downloads/download.php? ' .
    'file=/technology/ecf/dailies/' . $file .'">' . $file .
    '</a>
}
```

Now as time marches on, the list of available daily downloads gets larger and larger. There's lots of room on dev.eclipse.org, but it's not good practice to let something be so open-ended. The Eclipse Fundation does not allow its users to set up cron jobs (rightly so, it's a security leak). But you can request that Eclipse System Administration set one up for you, and if you got a good reason, they do it. We have a cron that looks as follows. It gets run once a week.

```
find /home/data/users/tkubaska/downloads/technology/ecf/dailies -mtime +7
    -name '*.zip' -type f -exec rm -rf {} \;
```

Second Stage

We want to be able to force builds remotely. And we want to do that from the cruisecontrol web reporting tool. The cruisecontrol site makes a distinction between the *web reporting tool* and the *cruisecontrol GUI*. The web reporting tool is a war file that runs in tomcat's webapps folder (or under your favourite servlet engine), and the cruisecontrol GUI is a jnlp file.

Setting Up the Cruisecontrol Web Reporting Tool

How do you make the war file?

As root, go into the directory /opt/cruisecontrol-2.7/reporting/jsp. Issue the command sh build.sh war

You get lots of output on the screen. You get asked for a logs directory; I chose /opt/build.ecf/logs/. Now I don't know if this is absolutely reproducible, but I think I needed that trailing slash (/). You also get asked for a status file; I chose ECFcurrentbuildstatus.txt. You get asked for an artifacts directory; I chose /opt/build.ecf/logs/. I'm not really using artifacts (an artifact is something like a junit report).

The result is a file called cruisecontrol.war in the directory /opt/cruisescontrol-2.7/reporting/jsp/dist. As root, copy this file to tomcat's

webapps folder which on ecf2 is /srv/www/tomcat5/base/webapps.

Then, as **root**, start up tomcat.

/usr/share/tomcat5/bin/startup.sh

Using the Web Reporting Tool Remotely

From Portland, we want to be able to log onto **ecf2** in Corvallis and force either an auto or a daily build. We do this with *ssh passphraseless key authentication* and *ssh port forwarding*.

We do our remote administration from the command line, from a port-forwarded browser, and from a port-forwarded desktop. From the command line, we can edit the confiuration files, start/stop cruisecontrol, and look at log files. We use the port-forwarded browser to force builds. We use the port-forwarded browser to configure Eclipse and sometimes force builds.

Getting a Remote Desktop

I'll show this from Windows, but it works also from Linux.

From Windows, I use PuTTY. Set ecf2.osuosl.org as the host name and port 22. Click on Load, then Open.

PuTTY Configuration	? 🗙
Category:	
 Session Logging Terminal Keyboard Bell Features Window Appearance Behaviour Translation Selection Connection Data Proxy Telnet Rlogin SSH Serial 	Basic options for your PuTTY session Specify the destination you want to connect to Host Name (or IP address) Port ecf2.osuosl.org Connection type: Raw Telnet Rlogin Save or delete a stored session Saved Sessions ecf2 Default Settings dev.eclipse.org ecf2 halibut101C Delete Close window on exit Always Never Only on clean exit
About <u>H</u> elp	<u>O</u> pen <u>C</u> ancel

Get a login screen. Login and start the vncserver. If this is the first time you are running **vncserver**, you get to set a password. Remember it.

Now as a sidenote, realize that the window you get by default is going to be twm. If you like twm, that's fine. I prefer gnome. You can choose your window manager by putting the appropriate **xstartup** file in .vnc in your home. Note the **passwd** file. It stores your vnc password as a data file.

ed@ecf2:~/.vnc> ls otal 212	5 -1				
otal 212 rw-rw-r 1 ted ci	magtorg 110021	2007 10 21	12.00	ogf2.4 log	
rw-rw-r 1 ted ci rw-rr 1 ted us					
rw-rw-r 1 ted ci					
rw-rr 1 ted us		2007-10-18			
rw-rw-r 1 ted ci	masters 32655	2007-10-18	21:48	ecf2:6.log	
rw-rw-r 1 ted ci	masters 6	2007-10-18	21:47	ecf2:6.pid	
rw-rr 1 ted us	sers 2909	2007-08-01	14:51	ecf2:9.log	
rw-rr 1 ted us	sers 4	2007-08-01	14:44	ecf2:9.pid	
rw 1 ted us	sers 8	2007-08-01	14:08	passwd	
rwxr-xr-x 1 ted us	10.05	2007-08-01	14.00		

Assuming you got the files in .vnc that you want, here's how the start for vncserver looks.



Then, open up another instance of PuTTY. Load ecf2. Then, choose SSH/Tunnels. Type in 5904 for the Source port. The 4 in 5904 comes from the 4 in the login window shown above. Also type in localhost:5904 for Destination.

😵 PuTTY Configurat	tion			? 🗙
Category:				
Session Logging Terminal Keyboard Bell Features Window	*	Port forwarding	ns controlling SSH po ccept connections fro s do the same (SSH-2 s:	om other hosts
Appearance Behaviour Translation Selection Colours Connection Data Proxy Telnet		Add new forward Source port Destination	led port 5904 localhost.5904	A <u>d</u> d
Rlogin SSH Auth TTY X11 Bugs	*	 Auto 		O IPv <u>6</u>
About I	<u>H</u> elp		<u>O</u> pen	<u>C</u> ancel

Click on Add.

😵 PuTTY Configurat	tion	? 🗙
Category:		
Session Logging Terminal Keyboard Bell Features Window Appearance Behaviour Translation Selection Colours Connection Data Proxy Telnet Rlogin SSH Kex Auth TTY		Options controlling SSH port forwarding Port forwarding Local ports accept connections from other hosts Remote ports do the same (SSH-2 only) Forwarded ports: Remove L5904 localhost5904 Add new forwarded port: Add Source port Add Destination Output Quotal Remote Dynamic Auto IPv4 IPv6
X11 Tunnels Bugs	*	
About	<u>-</u> lelp	<u>O</u> pen <u>C</u> ancel

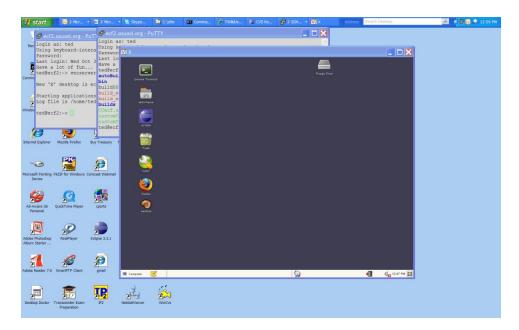
Then, click on Open and login. On your Windows box, open up RealVNC (you've already installed this, right?)

VNC View	er : Conn	ection Details	X
VQ	Server:	127.0.0.1:4	*
	Encryption:	Always Off	*
<u>A</u> bout	Opti	ons OK	Cancel

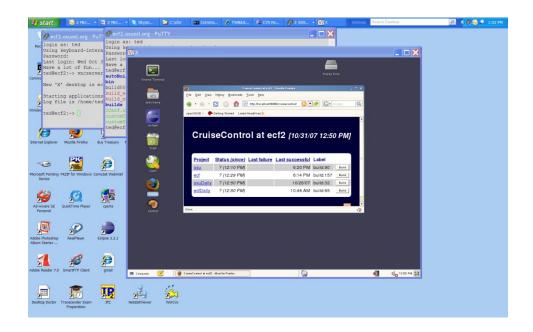
Click on OK.

VNC Viev	wer : Authe	ntication [No Encryption]
VP	Username:	ОК
<u> </u>	Password:	Cancel

Type in your password. This is not your user password on **ecf2** but rather the password you chose when you set up **vncserver** for the first time. Then you see the VNC window into **ecf2**.



Open up Firefox in the ecf2 window. Navigate to <u>http://localhost:8080/cruisecontrol</u>. The cruisecontrol GUI comes up. Notice that there are actually four projects listed. (Currently, we have nine projects.) osu and ecf are the Auto Builds for the osu and ecf projects (we haven't talked about osu builds yet). osuDaily and ecfDaily are the Daily Builds.



OK, let's start a build. First, if you want to monitor the build, open up an **xterm**. You're not going to get the **gnome** terminal launcher by default. I set one up previously. In the **xterm**, **cd** to

/opt/build.ecf and issue a tail -f nohup.out.

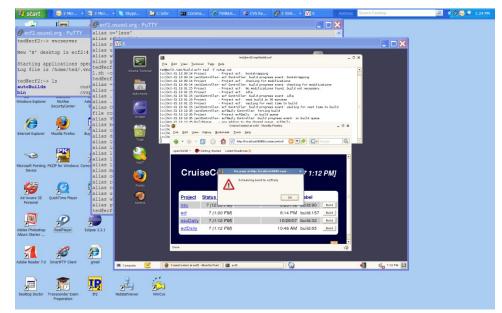
Cruisecontrol runs all the time. It does sometimes die or hang, but it's pretty reliable. But I started it with **nohup** so that I could log out of ecf2. I started it as follows.

```
nohup /opt/cruisecontrol-2.7/main/bin/cruisecontrol.sh -configfile config.xml
-port 8000 -rmiport 1099'
```

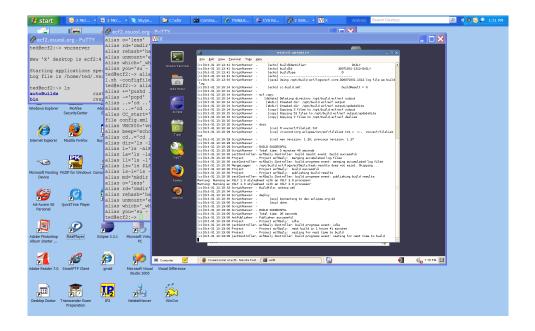
You can see its output by looking at nohup.out.

🛃 start 🔗 😡 2 Micr 🔹 🔯 2 Micr	💈 Skype 🏠 C:\xfer 🛛 🐼 Comma	. 💋 TWIKUS 😂 CVS Re	🛃 2 SSH ▼ 🔽 X	Address Search Desktop	🔎 🍕 😡 🔎 1:19 PM
A 1 Rect					
	as: ted		<u></u>		
ted@ecf2:~> vncserver Using				_ 0	X
New 'X' desktop is ecf2:4 Last	10	ted ide cf2:/cottboild.e/		_ U X	
Starting applications spec	cf File Edit View Territol				
Log file is /home/ted/.vnc bin	Grame Terrinal tedgecf2 <> cd /opt/build.ecf> 1	ail of nohup.out		A 1	
ted@ecf2:~> 1s build build		ntroller- ecf Controller: build progre	ss event: bootstrapping		
autoBuilds cust build bin cvst build	[cc]Oct-81 18:00:14 ject5	ntroller- ecf Controller: build progre	ss event: checking for modifications		
Windows Explorer McAfee Add CCecf	(cc)0ct-81 18:01:15 Project (cc)0ct-81 19:01:15 jectD	t - Project ecf: idle ntroller- ecf Controller: build progre	as event: idle		
SecurityCenter E custo	TT [cc]0ct-31 13:01:15 Project	t - Project ecf: waiting for ne	90 minutes mt time to build ss event: vaiting for next time to build	-	
custo tede		ntroller- ecf controller: build progre	ss event: valting for next time to build		
	(1) (N)	CruiseCentral at ect2 Manif	a Fordes	. C ×	
Internet Explorer Mozilla Firefox Buy Treasury	Bie Edit Name b	igtory Bookmada Tools Help			
	Test 🦛 - 🛞 - 😭	🛛 🙆 🕅 http://local.host.8080/cnu	isecontrol/ 🔯 💌 🔶 🔀 • Google		
	epenSUSE = 🌩 Ge	ming Started Latest Headlines 🚴			
	VasT				
Microsoft Pointing PKZIP for Windows Comcast Webme Device	al		0		
	Cruis 👔	econtrol at eci	2 [10/31/07 1:07 PM]		
	Finder				
Ad-Aware SE QuickTime Player oports	Project S	Status (since) Last failure Las	t euccaseful Label		
Personal	rastiles OSU	? (12:55 PM)	6:20 PM build.90 Build		
	ed ad	? (1:00 PM)	6:14 PM build 157 Build		
	osuDaily	? (1:07 PM)	10/26/07 build.32 Build		
Adobe Photoshop RealPlayer Eclipse 3.3.1	ecfDaily	? (1:07 PM)	10:48 AM build.65 Build		
Album Starter					
	Done		.383	4	
📶 🎜 🌮					
Adobe Reader 7.0 SmartFTP Client gmail	🗮 Computer 🗾 🔮 CraiseCornel at ect2 - M	ezilla Fiset	Q	📲 📢 🖬 1.07 PM	
	ata an				
Desktop Doctor Transcender Exam JP2	NetstatViewer WinCvs				
Preparation					

Click on the Build button for ecfDaily. Then, click on OK.

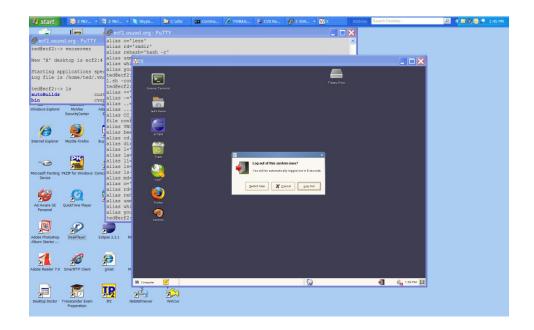


Here's the build output.



Notice from the build output that the build is successful (buildResult=0 ... first red arrow), that the checkin of filelist.txt occurs (the second red arrow), and that the scp transfer occurs (the third red arrow).

							ed@ecf2:/opt/build.ecf	- 8
<u>F</u> ile	<u>E</u> dit	⊻iew	Terminal	Ta <u>b</u> s	Hel	р		
			1 ScriptF				buildIdentifier:	DAILY
			1 ScriptF				buildId:	20071031-1312-DAILY
			1 ScriptF			[echo]	buildType:	D
cc]Oct	t-31	13:12:4	1 ScriptF	Runner	-	[echo]		
cc]Oct log.	t-31	13:12:5	3 ScriptF	Runner	-	[java]	Using /opt/build.ecf/log	gs/ecf.core-D20071031-1312.log file as buil
	+-31	13.18.1	4 ScriptF	Runner		[echo]	cc-build.xml:	buildResult = 0 🖌
			4 ScriptF			(acrio)	ce bored. Ante.	burtuneburt o
						ecf.copy:		
			4 ScriptF				Deleting directory /opt,	/build ecf/ecf output
			4 ScriptF				Created dir: /opt/build.	
			4 ScriptF					.ecf/ecf.output/updateSite
			4 ScriptF				Copying 3 files to /opt	
			5 ScriptF					t/build.ecf/ecf.output/updateSite
			5 ScriptF				Copying 3 files to /opt	
			5 ScriptF			(cob)	copying 5 files to /opt,	/build.ec//ecf.dailies
			5 ScriptF			lait.		
							M www/ecf/filelist.txt	—
			4 ScriptF					
	t-31	13:18:2	9 ScriptF	kunner	-	[cvs]	/cvsroot/org.eclipse/www	w/ecf/filelist.txt,v < www/ecf/filelist
xt								
			9 ScriptF		-	[cvs]	new revision: 1.28; pre-	vious revision: 1.27
			9 ScriptF		-			
						SUILD SUCCES		
							5 minutes 48 seconds	
							ntroller: build result e	
			9 Project				Daily: merging accumulat	
								event: merging accumulated log files
			9 MergeLo					esults does not exist. Skipping
			9 Project				Daily: build successful	11210
			9 Project				Daily: publishing build	
								event: publishing build results
							XSLT 2.0 processor	
							XSLT 2.0 processor	
					- E	Buildfile: :	antscp.xml	
			4 ScriptF		-			
			4 ScriptF					· · · · · · · · · · · · · · · · · · ·
			5 ScriptF				Connecting to dev.eclip:	se. org: 22
			3 ScriptF			[scp]	done.	
			9 ScriptF					
						BUILD SUCCES		
cc]Oct	t-31	13:19:0	9 ScriptF	Runner	- 1	fotal time:	26 seconds	
						ublisher su		
cc]Oct	t-31	13:19:0	9 Project	:	- F	Project ecfl	Daily: idle	
cc]Oct	t-31	13:19:0	9 jectCor	ntrolle	- e	cfDaily Con	ntroller: build progress	event: idle
			9 Project				Daily: next build in 1	
			9 Project				Daily: waiting for next	
								event: waiting for next time to build



Click on Log Out. Then, in one of the PuTTY windows, find the pid of the **vnc** process and kill it.

🔮 ecf2.osuosl.org - PuTTY	
ted@ecf2:~> ps ax grep vnc	-
26448 pts/0 S 0:16 Xvnc :4 -desktop X -httpd /usr/share/vnc/classes -auth /	ho
me/ted/.Xauthority -geometry 1024x768 -depth 24 -rfbwait 120000 -rfbauth /home/ted/	. v
nc/passwd -rfbport 5904 -fp /usr/share/fonts/misc:unscaled,/usr/share/fonts/local,/	us
r/share/fonts/75dpi:unscaled,/usr/share/fonts/100dpi:unscaled,/usr/share/fonts/Type	1,
/usr/share/fonts/URW,/usr/share/fonts/Speedo,/usr/share/fonts/truetype,/usr/share/f	ion
ts/uni,/usr/share/fonts/CID	
27695 pts/1 R+ 0:00 grep vnc	
ted@ecf2:~> kill 26448	
tedfecf2:~>	

Logout of any remaining PuTTY windows.

Getting a Remote Browser

We want to force builds without having to get a remote desktop. This involves forwarding ports 8000 and 8080. But just forwarding the ports is not enough.

At first that's what I did, and the JMX console on port 8000 is functional; the cruisecontrol GUI (that tomcat war file) was not. The cruisecontrol GUI came up, but the **Build** buttons didn't do anything.

You can force builds through the JMX console, but it's clunky. The JMX console is very functional, and what we want to do is very simple. It's easier to use the cruisecontrol GUI.

Here's what I had to do.

Ecf2 runs SuSE Linux, which wants to do a bunch of stuff by default that gets in the way. Before making the cruisecontrol.war file, I took out the default SuSE stuff by adding the following lines to .bashrc for both root and ted.

```
unset JDK_HOME
unset JAVA_BINDIR
unset JAVA_HOME
unset JRE_HOME
unset SDK_HOME
unset JAVA_ROOT
```

Then defined the JAVA HOME I wanted.

```
JAVA_HOME=/opt/jdk1.6.0_04
export JAVA_HOME
```

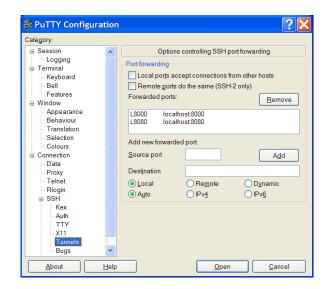
In root's .bashrc I defined

```
CATALINA_HOME=/opt/apache-tomcat-6.0.13
export CATALINA_HOME
```

Then, I edited the web.xml in /opt/apache-tomcat-6.0.13/webapps/cruisecontrol/WEB-INF. For some reason I do not understand, the IP address did not work. But I used localhost and that worked.

```
<context-param>
	<param-name>cruisecontrol.jmxhost</param-name>
	<param-value>localhost</param-value>
	<description>If your server doesn't know it's own proper IP address or it's name isn't
	resolvable by other machines on the network, set the IP address or resolvable name here. This
	will enable things like the "Force Builds" button to work.
	</description>
</context-param>
```

To use the CC GUI on ecf2, make a PuTTY connection as follows.



Then, point your browser to http://localhost:8080/cruisecontrol. You see the following.

) ▼ 🙋 http://localhost:808	30/cruisecontrol/		~	🍫 🗙 apache ant	
G-	💙 Go 🕂 🌍 🌄 👻 🕨	🛛 🗸 📩 Bookmari	ks 🛛 🥸 12 blocked 🤺	🦫 Check 👻 🔦 AutoLink	• » 🧯
🗧 🕶 Talki M Gn	nai 💠 Home 🔪 Apa	90 🍇 IEEEt 🎸	Cr ×	• <u> </u> • 🖶 • 🗟	' <u>P</u> age 🔻 🍥 1
C!		1 - 4			
Crui	seContro	of at eci	[2 [6/21/08	8 7:29 PMJ	
Project	Status (since)	<u>Last failure</u>	Last successfu	l Label	
ecfIntegratio	<u>on</u> ? (6/16/08)		6/16/08	8 build.50 Build	
<u>3.3ecf2.0</u>	? (1:00 PM)		6/18/08	8 build.17 Build	
<u>3.4ecf2.0</u>	? (2:00 PM)		6/18/08	8 build.27 Build	
<u>ecf2.1</u>	? (7:18 PM)		6/19/08	8 build.26 Build	
<u>ecf2.0</u>	? (7:21 PM)		6/19/08	8 build.26 Build	
3.4Daily2.1	? (7:29 PM)		6/20/08	8 build.20 Build	
3.4Daily2.0	? (7:29 PM)		6/20/08	8 build.20 Build	
3.3Daily2.1	? (7:29 PM)		6/20/08	8 build.21 Build	
3.3Daily2.0	? (7:29 PM)		6/20/08	8 build.26 Build	
<u>5.5Dany</u> 2.0					
<u>5.5Daily2.0</u>					

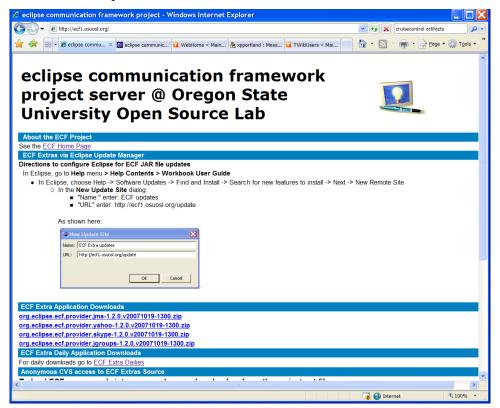
Third Stage

We want to incorporate the osu build into our automated build. The osu build consists of plugins that are in an early stage of development, so early that they have not yet passed the stringent IP requirements that Eclipse puts on its plug-ins. We have every expectation that they will, but

they haven't yet.

So the plug-ins belonging to the osu build are not kept in our eclipse repository. Rather they are kept in a repository on another machine at Oregon State called ecf1, and they are accessed through a web site hosted on ecf1.

I modified the ecf1 site to provide a link to ECF Extra Downloads.



Click on ECF Extra Dailies, and see the available osu downloads. Currently the osu plug-ins consist of the yahoo, skype, jms, and jgroups plug-ins.

- () (🖉 http://ecf1.osuosl.org/dailies.php 🔽 😽 🗙 cruisecontrol artifact	ts 🔎
-	88 🔻 🏉 e × 🥃 ecli » 👘 🔹 🔊 👘 🔹 🔂 🖓 Eage	▼
ECF I	Extra Daily Downloads	
org.eclip	se.ecf.provider.yahoo-20071031-1600-DAILY.zip	
org.eclip	se.ecf.provider.yahoo-20071026-1602-DAILY.zip	
org.eclip	se.ecf.provider.yahoo-20071025-0202-DAILY.zip	
org.eclip	se.ecf.provider.yahoo-20071023-1253-DAILY.zip	
org.eclip	se.ecf.provider.yahoo-20071023-1230-DAILY.zip	
org.eclip	se.ecf.provider.yahoo-20071023-1132-DAILY.zip	
org.eclip	se.ecf.provider.yahoo-20071019-1601-DAILY.zip	
org.eclip	se.ecf.provider.yahoo-20071018-1600-DAILY.zip	
org.eclip	se.ecf.provider.yahoo-20071017-1602-DAILY.zip	
org.eclip	se.ecf.provider.yahoo-20071016-1602-DAILY.zip	
org.eclip	se.ecf.provider.yahoo-20071015-1603-DAILY.zip	
org.eclip	se.ecf.provider.skype-20071031-1600-DAILY.zip	
org.eclip	se.ecf.provider.skype-20071026-1602-DAILY.zip	
org.eclip	se.ecf.provider.skype-20071025-0202-DAILY.zip	
org.eclip	se.ecf.provider.skype-20071023-1253-DAILY.zip	
org.eclip	se.ecf.provider.skype-20071023-1230-DAILY.zip	
org.eclip	se.ecf.provider.skype-20071023-1132-DAILY.zip	
org.eclip	se.ecf.provider.skype-20071019-1601-DAILY.zip	
org.eclip	se.ecf.provider.skype-20071018-1600-DAILY.zip	
org.eclip	se.ecf.provider.skype-20071017-1602-DAILY.zip	
org.eclip	se.ecf.provider.skype-20071016-1602-DAILY.zip	
org.eclip	se.ecf.provider.skype-20071015-1603-DAILY.zip	
org.eclip	se.ecf.provider.jms-20071031-1600-DAILY.zip	
org.eclip	se.ecf.provider.jms-20071026-1602-DAILY.zip	
org.eclip	se.ecf.provider.jms-20071025-0202-DAILY.zip	
org.eclip	se.ecf.provider.jms-20071023-1253-DAILY.zip	
org eclin	se.ecf.provider.jms-20071023-1230-DAILY.zip	

Originally, we ran all the projects on ecf2. We found, when we did that, that the Skype plugins were non-functional. The problem was that ecf2 is a 64-bit machine and that the Skype builds use some 32-bit plugins and fragments from the Eclipse plug-in directory. ecf2 requires a 64-bit version of Eclipse and a 64-bit version of Java.; the 32-bit versions do not work on ecf2. There should, I think, be some way of forcing the builds on ecf2 to use the 32-bit plugins and fragments, but I was not successful in doing this. Finally, I just set up cruisecontrol/ant/tomcat on ecf1 and run the osu builds on ecf1, and that works.

To add the osu build to cruisecontrol, edit config.xml. The new projects (called osu and osuDaily) are very similar to the ecf projects.

```
<project name="osu" buildafterfailed="false">
    <listeners>
    </listeners>
    <modificationset quietperiod="300">
      <cvs localworkingcopy="${localcopyEXAMPLE}" />
      <cvs localworkingcopy="${OSUlocalcopy}" />
    </modificationset>
    <schedule interval="2700">
      <ant buildfile="OSUcc-build.xml" target="ecf.osuosl" >
         <property name="mapVTag" value="HEAD" />
         <property name="feature" value="ecf.osuosl" />
         <property name="buildIdentifier" value="false" />
         <property name="buildType" value="A" />
         <property name="genFVSuffix" value="true" />
      </ant>
    </schedule>
    <publishers>
       <htmlemail ...>
       </htmlemail>
    </publishers>
</project>
```

Notice that the <modificationset /> is different because the cruisecontrol monitors a different set of files. Also, the antfile that is called in the <schedule /> task is different, and this antfile calls a different task called ecf.osuosl.

Inside osucc-build.xml, specific build information comes from the directory ecf.osuosl instead of ecf.core. This specific information consists of stuff like what map files to use, the location of the CVS repository, and the location of the Eclipse workspace.

Also, when the **osuDaily** build is performed the resulting zip files are not transferred to **dev.eclipse.org**. Rather they are transferred to **ecf1**.

The osu repository is on ecf1; the build is performed on ecf1 (used to be ecf2), and the resulting zip files stay on ecf1. Actually, they are transferred to ecf1 from ecf1, so that the code stays essentially the same.

How does that file transfer occur? Remember the file antscp.xml used in the ecf file transfer? There's a corresponding OSUantscp.xml. ecfl has a copy of the ecf2 public key, just like dev.eclipse.org. The files are transferred to the directory /home/ted/dailies on ecfl.

Now to have the files available on the ecf1 web site, they must reside in /var/www/localhost/ htdocs. We actually want them in a directory dailies in htdocs. The key authentication only works for an ordinary user (giving key authentication to root is a silly idea even if it is possible).

Nor do we want to open up htdocs to a group to which a user has write access. So what I did was make a link in htdocs to a directory in my home that contains the osu zips.



Then if you have the filenames in an array (called **\$file**), you present them for download as follows.

```
foreach ($files as $file) {
  echo '  <a href="http://ecf1.osuosl.org/dailies/' . $file . '">'
  . $file .'</a>';
}
```

How to get them into a **\$file** array. The following PHP function does that.

Then,

```
$files=directoryToArray("/var/www/localhost/htdocs/dailies",false);
```

Oh, and the same thing with ecf1 ... OSU won't let us write our own cron job; we must make a request.

Fourth Stage

I'm going to describe the fourth stage in another file because this one is getting too long. Here's what happened in our fourth stage.

- We added a number of new projects on **ecf2** for a total of nine. The extra projects have to do with using CVS branches.
- We have a 2.0 and a 2.1 branch. We want to continue work on 2.1 and provide 2.0 for the Eclipse Ganymede release.

- We have a branch used for integration with the Eclipse platform. The Eclipse platform itself uses six ECF plugins/fragments for 3.4. So we need a project that provides these plugins to the platform on a weekly basis.
- We want projects for the Eclipse 3.3 and 3.4 releases. These release builds must have their jars signed with Eclipse's Verisign certificate. The difference between ECF provided for 3.3 and ECF provided for 3.4 is that the 3.3 ECF includes the plugins/fragments that in 3.4 ECF are provided by the platform. So the 3.4 ECF must not include these plugins.