USE CASE	
NUMBER	USE CASE

Pre Production Build AND user wants to run version manager application

	Make sure the config.properties files looks like screenshot below. Depending on what the user wants the application will prompt if user wants to build							
Step 1	<pre>(release environment other parameters MUST be included in config.properties file) or version 1 default = false 2 nexus.url = http://gbvmapscend01.emea.win.ml.com:8085/nexus/content/repositories/releases/com/bofa/scp/ 3 4 current.path = C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\2.137\ER1\ 5 build = yes 6 control.path = C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\Core\2.136\ER1\ 7 dry.rum = no 8 update.from = nexus 9 !Please delimit lists of major and minor releases by semicolons-";" 10 major.releases = SCDPestrop 11 minor.releases = tradeviewer</pre>							
	12 13 release.env=prod							
	Run the version manager (java application). See command line examples below . In the command line the number of arguments specified determines the end result.							
	For example 0 command arguments would resemble (see below). In addition a developer would not include any arguments in command to verify updating of pom.xmls was completed correctly (only updating portion)							
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build_							
Step 2	For example, 1 command arguments would resemble (see below). In addition, only the first portion of project is build (JAR files-SCTDesktop) which would help develop understand in JARs were updated correctly and built correctly							
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build "mvn clean package ins tall"							
	For example, 2 command arguments would resemble (see below). Entire project (JAR and WAR files are built and updated), developer would use this pre- production build as evidence the process works							
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning≻java Build "mvn clean package ins tall" "mvn clean package"							
Step 3	Application prompts user to "direct build" and user inputs YES. Thus, control file path, build environment and other version manager prompts are BYPASSED. Defaults that will determine version numbers are obtained prompts (Control File, Environment, Compare to nexus or control file etc)							
Expected Results	A pre production run we only expect to see the command line outputs (indivudual results for command lines are explained in Step 2)							

Pre Production Build but user wants to SKIP version manager application by including pertinent information (release environment, region, etc..) in the config.properties file of his/her project

	Notice "default" is equal to meaning developing wants to SKIP version manager application prompts and version according to his/her configuraitons.property file (but note the developer DOES want to BUILD). Specify pertinent parameters in the config.properties file, for examples look at
	screenshot below:
	<pre>1 default = true 2 nexus.url = http://gbvmapscend01.emea.win.ml.com:8085/nexus/content/repositories/releases/com/bofa/scp/ 3</pre>
Step 1	<pre>4 current.path = C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\2.137\ER1\ 5 build = ves</pre>
	6 control.path = C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\Core\2.136\ER1\ e deserved
	/ dry.run ho 8 update.from = nexus
	9 !Please delimit lists of major and minor releases by semicolons-";" 10 major.releases = SCDDesktop
	11 minor.releases = tradeviewer
	13 release.env=prod
	Run the version manager (java application). See command line examples below . In the command line the number of arguments specified determines the
	end result.
	of pom.xmls was completed correctly (only updating portion)
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build_
Step 2	For example, 1 command arguments would resemble (see below). In addition, only the first portion of project is build (JAR files-SCTDesktop) which would help develop understand in JARs were updated correctly and built correctly
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build "mvn clean package ins tall"
	For example, 2 command arguments would resemble (see below). Entire project (JAR and WAR files are built and updated), developer would use this pre- production build as evidence the process works
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build "mvn clean package ins tall" "mvn clean package"
_	Application reads current project source path from the config.properties file shown above. In this example, the path is:
Step 3	<pre>4 current.path = C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\2.137\ER1\</pre>
	Application reads from the config.properties file shown above whether user to directly build the project (yes) or update versioning (no), in this example the
Step 4	answer is: NO. Then read the control file path from the file.
	<pre>control.path = C:\perforce\nbkgged scd\EMEA Credit\tactical\projects\SCD\Core\2.136\ER1\</pre>
	Application reads from the config.properties file shown above whether user wants to use Nexus's latest release or the control file as the baseline to update
	the pom file. (Different answer specified in the following steps)
	For example, it the latest version in Nexus for desktop module is 1.0.6 AND the input control file version for desktop module is 1.0.4.
Step 5	If indicate NEXUS , then the application will compare the desktop module with the nexus latest release and update the version number based on 1.0.6.
	8 update.from - nexus
	If indicate CONTROL , then the application will compare the desktop module with the nexus latest release and update the version number based on 1.0.4.
	8 update.from = control
Step 6	In contig.properties file, since "major.releases" and "minor.releases" have no values specified, the update type is "default release", which increments the
	OR 1.0.5 if "CONTROL" is chosen in Step 5.
Stop 7	Updated version number will show in the newly generated pom.xml.test files but the original pom.xml will stay as not modified. A pre production run we only
Step /	expect to see the command line outputs (indivudual results for command lines are explained in Step 2)
Expected Results	A pre production run we only expect to see the command line outputs based on a USER modified configuration.properties file

Pre production build and user wants to run

version manager application. In addition,

user wants to version according to patch release

	Make sure the config.properties files looks like screenshot below.					
	<pre>1 default = false 2 nexus.url = http://gbvmapscend01.emea.win.ml.com:8085/nexus/content/repositories/releases/com/bofa/s</pre>					
	<pre>3 4 ourrent.path = C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\2.137\ER1\</pre>					
Step 1	<pre>5 build = no 6 control.path = C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\Core\2.136\ER1\</pre>					
otop i	7 dry.run = no 8 update.from = nexus					
	9 !Please delimit lists of major and minor releases by semicolons-";" 10 major.releases = 10 major.releases =					
	11 minor.releases = 12 13 malasso coursed					
	Run the version manager (java application). See command line examples below. In the command line the number of arguments specified determines the					
	end result.					
	For example 0 command arguments would resemble (see below). In addition a developer would not include any arguments in command to verify updating of pom.xmls was completed correctly. The build uses conditions specified in the config.properties file. Since direct build is specified as "no", the pom.xml files are updated. Since no command line argument is specified, neither SCTDesktop nor webstart project is built. In all, the application does nothing to the SCD application.					
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build_					
Step 2	For example 1 command arguments would resemble (see below). The build uses conditions specified in the config.properties file. Since direct build is specified as "no", the pom.xml files will be updated. Since one command line argument is specified, the application builds the SCTDesktop project based on the command line argument but not the webstart project. In all, the WAR file is not built.					
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build "mvn clean package ins tall"					
	For example 2 command arguments would resemble (see below). The build uses conditions specified in the config.properties file. Since direct build is specified as "no", the pom.xml files will be updated. Two (or more) command line arguments are specified, SCTDesktop project is built based on the first command line argument and webstart project is built based on the second command line argument. Finally the built WAR file is renamed to be "SCTDesktop-env" where env is the value specified in the "release.env" field in config.properties file.					
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build "mvn clean package ins tall" "mvn clean package"					
Step 3	Application prompts user for the current project source path, for example, user can input: C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\Core\SCD\					
Step 4	Application prompts user to directly build the project (yes) or update versioning (no), user input: NO . Then the application prompts user for the control file path (the released version to be compared with), for example: C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\Core\2.136\ER1\					
	Application prompts user to use Nexus's latest release or the control file as the baseline to update the pom file. (Different answer specified in the following steps) Store example, if the latest version in Nexus for desktop module is 1.0.6 AND the input control file version for desktop module is 1.0.4					
Step 5	User input NEXUS , then the application will compare the desktop module with the nexus latest release and update the version number based on 1.0.6					
1	User input CONTROL , then the application will compare the desktop module with the nexus latest release and update the version number based on 1.0.4.					
	In configuremention file, gines, "major releases," and "minor releases," have no values specified, the undate type is "default releases", which increments the					
	patch version number only. In the above example, IF there's difference, version manager will update the version to 1.0.7 if "NEXUS" is chosen in Step 5 OR 1.0.5 if "CONTROL" is chosen in Step 5.					
Step 6	NOTE: the version number looks like					
	1 . 0 . 6					
ł	major-release-number.minor-release-number.patch-release-number					

Step 7	Application prompts user if this is a dry-run or not (dry-run creates new pom.xml.test files instead of overwriting the original pom.xml file). User input: YES.
Expected Results	Updated version number will show in the newly generated pom.xml.test files but the original pom.xml will stay as not modified. A pre production run we only expect to see the command line outputs (indivudual results for command lines are explained in Step 2)

Pre production build and user wants to BYPASS version manager application (user must specify pertinent information such as environment). In addition, user wants to version according to patch release

	Make sure the config.properties files looks like screenshot below.							
	<pre>1 default = true 2 nexus.utl = http://dbymapscend01.emea.win.ml.com:8085/nexus/content/repositories/releases/com/bofa/sop/</pre>							
	3							
	<pre>4 current.path = C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\Core\SCD</pre>							
- ·	<pre>6 control.path = C:\perforce\nbkgged scd\EMEA Credit\tactical\projects\SCD\Core\2.136\ER1\</pre>							
Step 1	7 dry.run = yes							
	8 update.from = control 9 UPlace delimit lists of major and minor releases by semicolons-":"							
	10 major releases =							
	11 minor.releases =							
	12 13 release.env=prod							
	Run the version manager (java application). See command line examples below. In the command line the number of arguments specified determines the							
	end result.							
	For example 0 command arguments would resemble (see below). In addition a developer would not include any arguments in command to verify updating							
	or pom.xmis was completed correctly. The build uses contained in the control properties tile. Since direct build is specified as no, the pom.xmi							
	These are updated. Since no command line argument is specified, neither SC i Desktop nor webstart project is built. In all, the application does nothing to the							
	SUD application.							
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build_							
	For example 1 command arguments would resemble (see below). The build uses conditions specified in the config.properties file. Since direct build is							
o , o	specified as "no", the pom.xml files will be updated. Since one command line argument is specified, the application builds the SCTDesktop project based							
Step 2	on the command line argument but not the webstart project. In all, the WAR file is not built.							
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build "mvn clean package ins tall"							
	For example 2 command arguments would resemble (see below). The build uses conditions specified in the config.properties file. Since direct build is							
	specified as "no", the pom.xml files will be updated. Two (or more) command line arguments are specified, SCTDesktop project is built based on the first							
	command line argument and webstart project is built based on the second command line argument. Finally the built WAR file is renamed to be							
	"SCTDesktop-env" where env is the value specified in the "release.env" field in config.properties file.							
	C:\us\scd\SCPRatches\SCDRuildex\src\Wersioning\jaua_Build_"mun_clean_package_ins							
	tall" "mun clean package"							
	Application reads current project source path from the config.properties file shown above. In this example, the path is:							
Step 3	4 ourwant math = C:\marfavna\mbkmmad and\EMEL Cradit\tartical\mvnarts\SCD\2 137\ED1\							
	• date is part = 0. (betalog impages joint interference) (applied (beb (210) (bit (
	Application reads from the config.properties file shown above whether user to directly build the project (yes) or update versioning (no), in this example the							
Step 4	answer is: NO. Then read the control file path from the file.							
	5 build = no							
	<pre>6 control.path = C:\perforce\nbkgged scd\EMEA Credit\tactical\projects\SCD\Core\2.136\ER1\</pre>							
	Application reads from the config.properties file shown above whether user wants to use Nexus's latest release or the control file as the baseline to update							
	the pom file. (Different answer specified in the following steps)							
	For example, if the latest version in Nexus for desktop module is 1.0.6 AND the input control file version for desktop module is 1.0.4.							

Step 5	If indicate NEXUS , then the application will compare the desktop module with the nexus latest release and update the version number based on 1.0.6.
	If indicate CONTROL , then the application will compare the desktop module with the nexus latest release and update the version number based on 1.0.4.
Step 6	In config.properties file, since "major.releases" and "minor.releases" have no values specified, the update type is "default release", which increments the patch version number only. In the above example, IF there's difference, version manager will update the version to 1.0.7 if "NEXUS" is chosen in Step 5 OR 1.0.5 if "CONTROL" is chosen in Step 5.
Step 7	Application prompts user if this is a dry-run or not (dry-run creates new pom.xml.test files instead of overwriting the original pom.xml file). User input: YES.
Expected Results	Updated version number will show in the newly generated pom.xml.test files but the original pom.xml will stay as not modified. A pre production run we only expect to see the command line outputs (indivudual results for command lines are explained in Step 2)

Pre production build in which the user runs the version manager application. In

- 5
- addition the user also specifies that versioning must be done to either or major and minor releases

	Make sure the config.properties files looks like screenshot below.
Step 1	<pre>1 default = false 2 nexus.url = http://gbvwapscend01.emea.win.ml.com:8085/nexus/content/repositories/releases/com/bofa/scp/ 3 4 current.path = C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\2.137\ER1\ 5 build = yes 6 control.path = C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\Core\2.136\ER1\ 7 dry.tum = yes 8 update.from = nexus 9 !Please delimit lists of major and minor releases by semicolons-";" 10 major.releases = StDbesktop 11 minor.releases = tradeviewer 12 13 release.env=prod</pre>
	Run the version manager (java application). See command line examples below. In the command line the number of arguments specified determines the For example 0 command arguments would resemble (see below). In addition a developer would not include any arguments in command to verify updating of pom.xmls was completed correctly. The build uses conditions specified in the config.properties file. Since direct build is specified as "no", the pom.xml files are updated. Since no command line argument is specified, neither SCTDesktop nor webstart project is built. In all, the application does nothing to the SCD application.
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build_
Step 2	For example 1 command arguments would resemble (see below). The build uses conditions specified in the config.properties file. Since direct build is specified as "no", the pom.xml files will be updated. Since one command line argument is specified, the application builds the SCTDesktop project based on the command line argument but not the webstart project. In all, the WAR file is not built.
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build "mvn clean package ins tall"
	For example 2 command arguments would resemble (see below). The build uses conditions specified in the config.properties file. Since direct build is specified as "no", the pom.xml files will be updated. Two (or more) command line arguments are specified, SCTDesktop project is built based on the first command line argument and webstart project is built based on the second command line argument. Finally the built WAR file is renamed to be "SCTDesktop-env" where env is the value specified in the "release.env" field in config.properties file.
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build "mvn clean package ins tall" "mvn clean package"
Step 3	Application prompts user for the current project source path, for example, user can input: C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\Core\SCD\
Step 4	Application prompts user to directly build the project (yes) or update versioning (no), user input: NO . Then the application prompts user for the control file path (the released version to be compared with), for example: C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\Core\2.136\ER1\

	Application prompts user to use Nexus's latest release or the control file as the baseline to update the pom file. (Different answer specified in the following steps) For example, if the latest version in Nexus for desktop module is 1.1.6 AND the input control file version for desktop module is 1.2.4.								
Step 5	User input NEXUS , then the ap	plication will compare	e the desktop module	with the nexus latest re	elease and update the	version number based on 1.1.6			
	User input CONTROL, then the	application will com	pare the desktop mod	ule with the nexus lates	st release and update	the version number based on 2.	2.4.		
	See versioning matrix below for	examples of increme	enting major/minor an	d patch releases					
	Nexus/Control Version Number	major.releases (specified in config.properties file)	minor.releases (specified in config.properties file)	type of release resulted in	default release number? (resulted in this column in the other sheet)	result			
	desktop module: 1.0.5	(blank)	(blank)	patch release	Yes	desktop module: 1.0.6			
	desktop module: 1.0.5	desktop	(blank)	major release	No	desktop module: 2.0.0			
	desktop module: 1.0.5	(blank)	desktop	minor release	No	desktop module: 1.0.7			
	desktop module: 1.0.5	desktop	desktop	major and minor release	No	desktop module: 2.1.0			
Step 6	desktop module: 1.0.5 BC module: 2.3.5	(blank)	(blank)	patch release	yes	desktop module: 1.0.6 BC module: 2.3.6			
	desktop module: 1.0.5					desktop module: 2.0.0			
	BC module: 2.3.5	desktop; BC	(blank)	major release	no	BC module: 3.0.0			
	desktop module: 1.0.5	(black)	dealers DO			desktop module: 1.1.0			
	BC module: 2.3.5	(blank)	desktop; BC	minor release	no	dealten medule: 2.1.0			
	BC module: 2.3.5	desktop: BC	desktop: BC	release	no	BC module: 3.1.0			
	desktop module: 1.0.5	000m0p, 20	doskop, bo	major release for desktop; minor		desktop module: 2.0.0			
	BC module: 2.3.5	desktop	BC	release for BC	no	BC module: 2.4.0			
	desktop module: 1.0.5	50		major release for BC; minor release for		desktop module: 1.1.0			
	BC module: 2.3.5	BC	desktop	desktop	no	BC module: 3.0.0			
	Application prompts user if this i	s a dry-run or not (d	ry-run creates new po	m.xml.test files instead	of overwriting the orig	ginal pom.xml file). User input: Y	ES.		
Step 7	7 dry.run = yes	-	-						
Expected Results	Updated version number will she expect to see the command line	ow in the newly gene outputs (indivudual	erated pom.xml.test file results for command	es but the original pom. ines are explained in S	xml will stay as not mo tep 2)	odified. A pre production run we	only		

Pre production build in which the user BYPASS the version manager application (and wants to version according to

6 parameters specified in the config.properties file). In addition the user also specifies that versioning must be done to either or major and minor releases

> User builds specifying an environment. User also executes the version manager

7 application. In addition the user also specifies that versioning must be done to either or major and minor releases

	Make sure t	he config.properties	files looks like scree	nshot below.					
	1 2	<pre>default = false nexus.url = http://gk</pre>	vmapscend01.emea.win.m	1.com:8085/nexus/conte	nt/repositories/releas	es/com/bofa/scp/			
	3 4	current.path = C:\per	force\nbkgged scd\EME&	. Credit\tactical\proje	cts\SCD\2.137\ER1\				
	5	5 build = no							
Step 1	7	dry.run = no	.iorce(ibkgged_scu(intx	_credit(cactical(proje	CCS(SCD)COLE(2.136)ERI	1			
	8	update.from = nexus !Please delimit lists	of major and minor re	leases by semicolons-"	;"				
	10	major.releases = SCTD minor.releases = trad	esktop leviewer						
	12								
	Pup the ver	release.env=prou	application) See com	mand line examples	bolow. In the comm	and line the number of	arguments specified date	orminos tho	
	end result.	Sion manager (Java a		inana ine examples	below . In the comm		arguments specified dete		
	For example	e 0 command argum	ents would resemble	(see below). In additi	ion a developer would	d not include any aroun	nents in command to veri	rifv updating	
	of pom.xmls	was completed corr	ectly. The build uses	conditions specified	in the config.properti	ies file. Since direct bui	ld is specified as "no", th	ne pom.xml	
	files are upo	ated. Since no com	mand line argument i	s specified, neither S	CTDesktop nor web	start project is built. In a	all, the application does n	nothing to the	
	SCD applica	ation.							
	C:\ws\sc	d\SCPBatches\	SCDBuilder\src	\Versioning>ja	wa Build_				
	For example	a 1 command argum	ents would recemble	(see below) The built	d uses conditions en	ecified in the configura	nerties file. Since direct l	build is	
Stop 2	specified as	"no", the pom.xml fi	les will be updated. S	Since one command li	ine argument is spec	ified, the application bu	ilds the SCTDesktop pro	oject based	
Siep 2	on the comr	mand line argument l	but not the webstart	project. In all, the WA	R file is not built.	, ,,			
	_								
	C:\ws\sc tall"	d\SCPBatches\	SCDBuilder\src	\Versioning>ja	iva Build "mvn	clean package i	ns		
	For example	For example 2 command arguments would resemble (see below). The build uses conditions specified in the config.properties file. Since direct build is							
	specified as	"no", the pom.xml fi	les will be updated. T	wo (or more) comma	ind line arguments ar	e specified, SCTDeskte	op project is built based o	on the first	
	command lin	ne argument and we	bstart project is built	based on the second	l command line argur	nent. Finally the built W	AR file is renamed to be	e	
	"SCTDeskto	op-env" where env is	s the value specified	in the "release.env" f	ield in config.properti	es file.			
	C:\ws\sc tall" "m	cd∖SCPBatches∖ nyn clean pack	SCDBuilder\src age"	∖Versioning>ja	wa Build "mvn	clean package i	ins		
	Application	prompts user for the	current project source	ce path, for example,	user can input:				
Step 3	C:\perforce\	nbkgged_scd\EMEA	A_Credit\tactical\proje	ects\SCD\Core\SCD\	١				
	Application	prompts user to direct	ctly build the project (yes) or update version	oning (no), user input	: NO. Then the applica	tion prompts user for the	e control file	
Step 4	path (the rel	path (the released version to be compared with), for example: C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\Core\2.136\ER1\							
	Application	prompts user to use	Nexus's latest release	e or the control file a	s the baseline to upd	ate the pom file. (Differ	ent answer specified in the	the following	
	steps)								
	For example	For example, if the latest version in Nexus for desktop module is 1.1.6 AND the input control file version for desktop module is 1.2.4.							
Step 5	User input NEXUS, then the application will compare the desktop module with the nexus latest release and update the version number based on 1.1.6					on 1.1.6			
	User input C	User input CONTROL , then the application will compare the desktop module with the nexus latest release and update the version number based on 2.2.4.						sed on 2.2.4.	
	See version	ing matrix below for	examples of increme	nting major/minor and	d patch releases				
	T		major.releases	minor.releases		default release			
		Nexus/Control	config.properties	config.properties	type of release	this column in the			
		Version Number desktop module: 1.0.5	file) (blank)	file) (blank)	patch release	other sheet) Yes	desktop module: 1.0.6		
		desktop module: 1.0.5	desktop	(blank)	major release	No	desktop module: 2.0.0		
		desktop module: 1.0.5	(blank)	desktop	minor release	No	desktop module: 1.0.7		
	1	desktop module: 1.0.5 desktop desktop desktop desktop release No desktop module: 2.1.0							
		desktop module: 1.0.5	desktop	desktop	release	No	desktop module: 2.1.0		

	1	desktop module: 1.0.5					desktop module: 1.0.6	
Step 6		BC module: 2.3.5	(blank)	(blank)	patch release	yes	BC module: 2.3.6	
		desktop module: 1.0.5					desktop module: 2.0.0	
		BC module: 2.3.5	desktop; BC	(blank)	major release	no	BC module: 3.0.0	
		desktop module: 1.0.5					desktop module: 1.1.0	
		BC module: 2.3.5	(blank)	desktop; BC	minor release	no	BC module: 2.4.0	
		desktop module: 1.0.5			major and minor		desktop module: 2.1.0	
		BC module: 2.3.5	desktop; BC	desktop; BC	release	no	BC module: 3.1.0	
		desktop module: 1.0.5			major release for		desktop module: 2.0.0	
					desktop; minor			
		BC module: 2.3.5	desktop	BC	release for BC	no	BC module: 2.4.0	
		desktop module: 1.0.5			major release for BC;		desktop module: 1.1.0	
		BC module: 2.3.5	BC	desktop	desktop	no	BC module: 3.0.0	
	Application	prompts user if this is	a dry-run or not (dry	-run creates new po	m.xml.test files instead	of overwriting the o	priginal pom.xml file). User in	nput: YE
Stop 7			, , ,			Ŭ	ö 1 ,	•
Step 7		7 dry.rur	n = no					
Expected Results	Updated ve	ersion number will sho	w in the newly gener	ated pom.xml.test file	es but the original pom.	xml will stay as not	modified. A pre production ru	run we c

User builds specifying an environment. BYPASS the version manager application (and wants to version according to

8 parameters specified in the

config.properties file). In addition the user also specifies that versioning must be done to either or major and minor releases

Step 1	Make sure the config.properties files looks like screenshot below.							
	<pre>1 default = true 2 nexus.url = http://gbvmapscend01.emea.win.ml.com:8085/nexus/content/repositories/releases/com/bofa/scp/ 3 current.path = C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\2.137\ER1\ 5 build = no</pre>							
	<pre>7 dry.rum = nd 8 update.from = nexus 9 iPlease delimit lists of major and minor releases by semicolons-";" 10 major.releases = SCTDesktop 11 minor.releases = tradeviewer 12 13 release.env=prod</pre>							
	Run the version manager (java application). See command line examples below . In the command line the number of arguments specified determines the end result.							
	For example 0 command arguments would resemble (see below). In addition a developer would not include any arguments in command to verify updating of pom.xmls was completed correctly. The build uses conditions specified in the config.properties file. Since direct build is specified as "no", the pom.xml files are updated. Since no command line argument is specified, neither SCTDesktop nor webstart project is built. In all, the application does nothing to the SCD application.							
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java_Build_							
Step 2	For example 1 command arguments would resemble (see below). The build uses conditions specified in the config.properties file. Since direct build is specified as "no", the pom.xml files will be updated. Since one command line argument is specified, the application builds the SCTDesktop project based on the command line argument but not the webstart project. In all, the WAR file is not built.							
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build "mvn clean package ins tall"							
	For example 2 command arguments would resemble (see below). The build uses conditions specified in the config.properties file. Since direct build is specified as "no", the pom.xml files will be updated. Two (or more) command line arguments are specified, SCTDesktop project is built based on the first command line argument and webstart project is built based on the second command line argument. Finally the built WAR file is renamed to be "SCTDesktop-env" where env is the value specified in the "release.env" field in config.properties file.							
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build "mvn clean package ins tall" "mvn clean package"							

	Application reads current proje	ct source path from t	he config.properties fi	e shown above. In this	example, the path is:		
Step 3	4 current.path = C:\perforce	e\nbkgged_scd\EMEA_Cre	dit\tactical\projects\	SCD\2.137\ER1\			
Step 4	Application reads from the con answer is: NO. Then read the 5 build = no 6 control.path = C:\perf	fig.properties file sho control file path from force\nbkgged_scd\EME#	wn above whether use the file. . Credit\tactical\proje	er to directly build the provide the provide the provide the provide the provide the provide the provided the	roject (yes) or update v	versioning (no), in this e	example the
	Application reads from the con the pom file. (Different answer For example, if the latest version	fig.properties file sho specified in the follow on in Nexus for deskt	wn above whether use ving steps) op module is 1.0.6 AN	er wants to use Nexus's	s latest release or the c version for desktop mo	control file as the baseli odule is 1.0.4	ne to update
	If indicate NEXUS, then the ap	oplication will compar	e the desktop module	with the nexus latest re	lease and update the v	version number based of	on 1.0.6.
Step 5	8 update.from = 1	lexus					
	If indicate CONTROL, then the	e application will com	pare the desktop mod	ule with the nexus lates	t release and update th	ne version number bas	ed on 1.0.4.
	8 update.from =	control					
	See versioning matrix below for	r examples of increm	enting major/minor an	d patch releases			
	Nexus/Control Version Number	major.releases (specified in config.properties file)	minor releases (specified in config.properties file)	type of release resulted in	default release number? (resulted in this column in the other sheet)	result	
	desktop module: 1.0.5	(blank)	(blank)	patch release	Yes	desktop module: 1.0.6	
	desktop module: 1.0.5	desktop	(blank)	major release	No	desktop module: 2.0.0	
	desktop module: 1.0.5	(blank)	desktop	minor release	NO	desktop module: 1.0.7	
	desktop module: 1.0.5	desktop	desktop	release	No	desktop module: 2.1.0	
Step 6	desktop module: 1.0.5 BC module: 2.3.5	(blank)	(blank)	patch release	yes	desktop module: 1.0.6 BC module: 2.3.6	
	desktop module: 1.0.5	de aldares DO	(black)			desktop module: 2.0.0	
	desktop module: 1.0.5	desktop; BC	(blank)	major release	no	desktop module: 1.1.0	
	BC module: 2.3.5	(blank)	desktop; BC	minor release	no	BC module: 2.4.0	
	desktop module: 1.0.5			major and minor		desktop module: 2.1.0	
	desktop module: 1.0.5	desktop; BC	desktop; BC	major release for	no	desktop module: 2.0.0	
	RC medule: 2.2.5	dealstein	P.C.	desktop; minor		PC medule: 2.4.0	
	desktop module: 1.0.5	desktop		major release for BC;	110	desktop module: 1.1.0	
	RC module: 2.2.5	BC	desiton	minor release for	20	RC modulo: 2.0.0	
	BC module: 2.3.5	BC	desktop	desktop	10	BC module: 3.0.0	
	Application prompts user if this	s is a dry-run or not (o	dry-run creates new po	m.xml.test files instead	l of overwriting the orig	inal pom.xml file). User	input: YES.
Step 7	7 dry.ru	n = no					
Expected Results	Updated version number will s expect to see the command lin	how in the newly gen e outputs (indivudua	erated pom.xml.test fil results for command	es but the original pom. ines are explained in S	xml will stay as not mo tep 2)	dified. A pre production	n run we only

User wants to execute version manager appliccation and build with specified

environment (prod, QA, str, etc). User also wants to version according to patch release number

	Make sure the config.properties files looks like screenshot below.
	1 default = false
	<pre>2 nexus.url = http://gbvmapscend01.emea.win.ml.com:8085/nexus/content/repositories/releases/com/bofa/scp/ 3</pre>
	4 current.path = C:\perforce\nbkgged_sod\EMEA_Credit\tastical\projects\SCD\Core\SCD
	<pre>5 build = no 6 control.path = C:\perforce\nbkggd sod\EMEA Credit\tactical\projects\SCD\Core\2.136\ER1\</pre>
Step 1	7 dry.run = no
	8 update.from = control 9 iPlease delimit lists of major and minor releases by semicolons=";"
	10 major.releases =
	11 minor.releases =
	13 release.env=prod
	Run the version manager (java application). See command line examples below. In the command line the number of arguments specified determines the
	end result.
	For example 0 command arguments would resemble (see below). In addition a developer would not include any arguments in command to verify undation
	To example o command againetism would estimate (see below), in addition a developer would not include any arguments in command to verify operating of nom ying was completed correctly. The build uses conditions specified in the confit morphetism file. Since direct build is specified as "not" the nom ying
	files are updated. Since no command line aroument is specified, neither SCTDesktop nor webstart project is built. In all, the application does nothing to the
	SCD application.
	C:\ws\sca\scrbatches\scbbuilder\sPc\versioning/java_build_
	For example 1 command arguments would resemble (see below). The build uses conditions specified in the config.properties file. Since direct build is
_	specified as "no", the pom.xml files will be updated. Since one command line argument is specified, the application builds the SCTDesktop project based
Step 2	on the command line argument but not the webstart project. In all, the WAR file is not built.
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build "mvn clean package ins
	For example 2 command arguments would resemble (see below). The build uses conditions specified in the config.properties file. Since direct build is
	specified as "no", the pom.xml files will be updated. Two (or more) command line arguments are specified, SCTDesktop project is built based on the first
	command line argument and webstart project is built based on the second command line argument. Finally the built WAR file is renamed to be
	"SCTDesktop-env" where env is the value specified in the "release.env" field in config.properties file.
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build "mvn clean package ins
	tall" "mun clean package"
Step 2	Application prompts user for the current project source path, for example, user can input:
Step 3	C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\Core\SCD\
	Application prompts user to directly build the project (yes) or update versioning (no), user input: NO. Then the application prompts user for the control file
Step 4	path (the released version to be compared with), for example: C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\Core\2.136\ER1\
	Application prompts user to use Nexus's latest release or the control file as the baseline to update the pom file. (Different answer specified in the following
	steps)
	For example, if the latest version in Nexus for desktop module is 1.0.6 AND the input control file version for desktop module is 1.0.4.
Step 5	User input NEXUS, then the application will compare the desktop module with the nexus latest release and update the version number based on 1.0.6
	User input CONTROL, then the application will compare the desktop module with the nexus latest release and update the version number based on 1.0.4.
	In config.properties file, since "major.releases" and "minor.releases" have no values specified, the update type is "default release", which increments the
	patch version number only. In the above example, IF there's difference, version manager will update the version to 1.0.7 if "NEXUS" is chosen in Step 5
	OR 1.0.5 if "CONTROL" is chosen in Step 5.
Step 6	
	NOTE: the version number looks like
	1 . 0 . 6
	major-release-number.minor-release-number.patch-release-number
Step 7	Application prompts user if this is a dry-run or not (dry-run creates new pom.xml.test files instead of overwriting the original pom.xml file). User input: NO to
Cich i	overwrite the original pom.xml files .

Expected Results Updated version number will show in the original pom.xml files (they are overwritten). An actual build run we only expect to see the command line outputs (indivudual results for command lines are explained in Step 2)

User wants to BYPASS version manager appliccation and build with specified (in config.properties file) environment (prod, QA, str, etc). User also wants to version according to patch release number

Step 1	<pre>Make sure the config.properties files looks like screenshot below. 1 default = true 2 nexus.url = http://gbvmapscend01.emea.win.ml.com:8085/nexus/content/repositories/releases/com/bofa/scp/ 3 4 current.path = C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\Core\SCD 5 build = no 6 control.path = C:\perforce\nbkgged_scd\EMEA_Credit\tactical\projects\SCD\Core\2.136\ER1\ 7 dry.run = no 8 update.from = control 9 !Please delimit lists of major and minor releases by semicolons-";" 10 major.releases = 11 minor.releases = 12 13 release.env=prod</pre>
	Run the version manager (java application). See command line examples below. In the command line the number of arguments specified determines the end result.
	of pom.xmls was completed correctly. The build uses conditions specified in the config.properties file. Since direct build is specified as "no", the pom.xml files are updated. Since no command line argument is specified, neither SCTDesktop nor webstart project is built. In all, the application does nothing to the SCD application.
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build_
Step 2	For example 1 command arguments would resemble (see below). The build uses conditions specified in the config.properties file. Since direct build is specified as "no", the pom.xml files will be updated. Since one command line argument is specified, the application builds the SCTDesktop project based on the command line argument but not the webstart project. In all, the WAR file is not built.
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build "mvn clean package ins tall"
	For example 2 command arguments would resemble (see below). The build uses conditions specified in the config.properties file. Since direct build is specified as "no", the pom.xml files will be updated. Two (or more) command line arguments are specified, SCTDesktop project is built based on the first command line argument and webstart project is built based on the second command line argument. Finally the built WAR file is renamed to be "SCTDesktop-env" where env is the value specified in the "release.env" field in config.properties file.
	C:\ws\scd\SCPBatches\SCDBuilder\src\Versioning>java Build "mvn clean package ins tall" "mvn clean package"
Step 3	Application reads current project source path from the config.properties file shown above. In this example, the path is:
Step 4	Application reads from the config.properties file shown above whether user to directly build the project (yes) or update versioning (no), in this example the answer is: NO. Then read the control file path from the file.
	Application reads from the config.properties file shown above whether user wants to use Nexus's latest release or the control file as the baseline to update the pom file. (Different answer specified in the following steps) For example, if the latest version in Nexus for desktop module is 1.0.6 AND the input control file version for desktop module is 1.0.4.

	If indicate NEXUS , then the application will compare the desktop module with the nexus latest release and update the version number based on 1.0.6.
Step 5	8 update.from = nexus
	If indicate CONTROL , then the application will compare the desktop module with the nexus latest release and update the version number based on 1.0.4.
	In config.properties file, since "major.releases" and "minor.releases" have no values specified, the update type is "default release", which increments the
Step 6	patch version number only. In the above example, IF there's difference, version manager will update the version to 1.0.7 if "NEXUS" is chosen in Step 5 OR 1.0.5 if "CONTROL" is chosen in Step 5.
Step 7	Application prompts user if this is a dry-run or not (dry-run creates new pom.xml.test files instead of overwriting the original pom.xml file). User input: NO to overwrite the original pom.xml files . 7 dry.run = no
Expected Results	Updated version number will show in the original pom.xml files (they are overwritten). An actual build run we only expect to see the command line outputs

Versioning Matrix

				default release	
	major.releases	minor.releases		number? (resulted in	
Nexus/Control Version	(specified in	(specified in	type of release resulted	this column in the other	
Number	config.properties file)	config.properties file)	in	sheet)	result
desktop module: 1.0.5	(blank)	(blank)	patch release	Yes	desktop module: 1.0.6
desktop module: 1.0.5	desktop	(blank)	major release	No	desktop module: 2.0.0
desktop module: 1.0.5	(blank)	desktop	minor release	No	desktop module: 1.0.7
desktop module: 1.0.5	desktop	desktop	major and minor release	No	desktop module: 2.1.0
desktop module: 1.0.5					desktop module: 1.0.6
BC module: 2.3.5	(blank)	(blank)	patch release	yes	BC module: 2.3.6
desktop module: 1.0.5					desktop module: 2.0.0
BC module: 2.3.5	desktop; BC	(blank)	major release	no	BC module: 3.0.0
desktop module: 1.0.5					desktop module: 1.1.0
BC module: 2.3.5	(blank)	desktop; BC	minor release	no	BC module: 2.4.0
desktop module: 1.0.5					desktop module: 2.1.0
BC module: 2.3.5	desktop; BC	desktop; BC	major and minor release	no	BC module: 3.1.0
desktop module: 1.0.5			major release for		desktop module: 2.0.0
BC module: 2.3.5	desktop	BC	desktop; minor release	no	BC module: 2.4.0
desktop module: 1.0.5			major release for BC;		desktop module: 1.1.0
BC module: 2.3.5	BC	desktop	minor release for desktop	no	BC module: 3.0.0

Note For multiple modules that are specified in the major.release or minor.release key field in config.properties file, separate them by ";"

	desktop
	Ivol
	BaseCorrelation
	BasketManager
	reporting framework
Modulos con ho specified	reportviewer
in the major release and	RFL
minor rologoo fieldo:	SCK
minor.release neids.	tradeviewer
	SCTDesktop
	ConfigurationsUat
	ConfigurationsProd
	ConfigurationsUatDualServer
	ConfigurationsStr

File If two directories contain the same files, then do not update the module's version number in the module's pom file. Otherwise, **Compare** increase version number based on the requirement shown in "Version Matrix" tag

Use Case #	Use Case Description	Directory 1 Content	Directory 2 Content	Directory Compare Result	Update version number?	Comment
		abc.doc	abc.doc			
		def.txt	def.txt			
	Same amount of files in both	config.properties	config.properties			File order does not
1	directory and file names and	help.java	help.java	Same	No	matter
	content are the same	file.jsp	file.jsp			matter.
		picture.pic	picture.pic			
		graph.jpg	graph.jpg			
			abc.doc (content			
			different from abc.doc			
	Same amount of files in both	abc.doc	in directory 1			
	directory and file names are the	def.txt	def.txt			File order does not
2	same but content is difference.	config.properties	config.properties	Different	Yes	matter
	Difference exists in non-picture	help.java	help.java			matter.
	type files.	file.jsp	file.jsp			
		picture.pic	picture.pic			
		graph.jpg	graph.jpg			
		abc.doc	abc.doc			
		def.txt	def.txt			
	Same amount of files in both	config.properties	config.properties			Currently the file
	directory and file names are the	help.java	help.java			comparator cannot
з	same but content is difference	file.jsp	file.jsp	Same	No	detect the difference
5	Difference exists in picture type		picture.pic (picture	Game	NO	in nic ing aif (nicture
			content is different			types) files
	illes.		from picture.pic in			types) mes.
		picture.pic	directory 1)			
		graph.jpg	graph.jpg			
			abcdef.doc (content			
			same as abc.doc in			
		abc.doc	directory 1)			
	Same amount of files in both	def.txt	def.txt			File order does not
4	direcotry. Two files have different	config.properties	config.properties	Different	Yes	matter
	names but the same content.	help.java	help.java			

		file.jsp	file.jsp			
		picture.pic	picture.pic			
		graph.jpg	graph.jpg			
			abcdef.doc (content			
			different from abc.doc			
		abc.doc	in directory 1)			
		def.txt	def.txt			File order dess not
		config.properties	config.properties	Different	Yes	metter
		help.java	help.java			maller.
		file.jsp	file.jsp			
		picture.pic	picture.pic			
		graph.jpg	graph.jpg			
		abc.doc	abc.jar			
	Same amount of files in both	def.txt	def.txt			
5	directory and file names and	config.properties	config.properties			
	content both different					
				Different	Voc	File order does not
				Different	165	matter.
		help.java	help.java			
		file.jsp	file.jsp			
		picture.pic	picture.pic			
		graph.jpg	graph.jpg			
		abc.doc	abc.doc (content			
			abcdef.txt (content			
			same as def.txt in			
	Same amount of files in both	def.txt	directory 1)			File order does not
6	directory and file names and	config.properties	config.properties	Different	Yes	matter
	content both different	help.java	help.java			
		file.jsp	file.jsp			
		picture.pic	picture.pic			
		graph.jpg	graph.jpg			
		abc.doc	abc.doc			
	Different amount of files in both	def.txt	def.txt			
	directory and file names and	config.properties	config.properties			
	content are the same (Directory	help.java	help.java			

		file.isp	file.isp			
	1 has more files)	picture.pic	picture.pic			
		graph.jpg	· ·			
		abc.doc	abc.doc			File order does not
7				Different	Yes	Yes matter. File type does
						not matter.
	Different amount of files in both	def.txt	def.txt			
	directory and file names and	config.properties	config.properties			
	content are the same (Directory	help.java	help.java			
	2 has more files)	file.jsp	file.jsp			
		picture.pic	picture.pic			
		graph.jpg	graph.jpg			
			test.xsl			
			abc.doc (content			
	Different amount of files in both		different from abc.doc			
	directory and file names are the	abc.doc	in directory 1)			
	directory and me names are the	def.txt	def.txt			
	Difference exists in non pisture	config.properties	config.properties			
	type files (Directory 1 has more	help.java	help.java			
	files)	file.jsp	file.jsp			
	liles)	picture.pic	picture.pic			
		graph.jpg				File order does not
8				Different	Yes	matter. File type does
Ũ			abc.doc (content	Binoroni	100	not matter
			different from abc.doc			not mation
	Different amount of files in both	abc.doc	in directory 1)			
	directory and file names and	def.txt	def.txt			
	content are the same (Directory	config.properties	config.properties			
	2 has more files)	help.java	help.java			
		file.jsp	file.jsp			
		picture.pic	picture.pic			
		graph.jpg	graph.jpg			
			test.xsl			
		abc.doc	abc.doc			
		def.txt	def.txt			
	Different amount of files in both	config.properties	config.properties			
	directory and file names are the	help.java				
	same but content is difference	file.jsp	file.jsp			

	Difference exists in picture type files. (Directory 1 has more files)	<mark>picture.pic</mark> graph.jpg	picture.pic (picture content is different from picture.pic in directory 1)			File order does not
9		abc.doc	abc.doc	Different	Yes	matter. File type does
		der.txt	der.txt			not matter.
		config.properties	config.properties			
	Different amount of files in both	neip.java	neip.java			
	directory and file names and	tile.jsp	file.jsp			
	content are the same (Directory		picture.pic (picture			
	2 has more files)		content is different			
	,		from picture.pic in			
		picture.pic	directory 1)			
		grapn.jpg	pn.jpg grapn.jpg			
			abcdef.doc (content			
	Different amount of files in both direcotry. Two files have different names but the same content.		same as abc.doc in			
		abc.doc	directory 1)			
		der.txt	der.txt			
		config.properties	config.properties			
	(Directory 1 has more files)	help.java	help.java			
	, , , , , , , , , , , , , , , , , , ,	file.jsp	file.jsp			
		picture.pic	picture.pic			
		graph.jpg		5.4	Ň	File order does not
10			abcdef.doc (content	Different	Yes	matter. File type does
			same as abc.doc in			not matter.
		abc.doc	directory 1)			
	Different amount of files in both	def.txt	def.txt			
	directory and file names and	config.properties	config.properties			
	content are the same (Directory	help.java	help.java			
	2 has more files)	file.jsp	file.jsp			
		picture.pic	picture.pic			
		graph.jpg	graph.jpg			
	_		test.xsl			
			abcdet.doc (content			
			different from abc.doc			
	Different amount of files in both	abc.doc	in directory 1)			
	directory and file names and	def.txt	def.txt			

	-			-		-
	content both different (Directory	config.properties	config.properties			
	1 bas more files)	help.java	help.java			
	Thas more mes	file.jsp	file.jsp			
		picture.pic	picture.pic			
		graph.jpg				
			abcdef.doc (content			
			different from abc.doc			
		abc.doc	in directory 1)			
	Different amount of files in both	def.txt	def.txt			
	directory and file names and	config.properties	config.properties			
	content are the same. (Directory	help.java	help.java			
	2 has more files)	file.jsp	file.jsp			File order dess pet
4		picture.pic picture.pic	File order does not			
1		graph.jpg	graph.jpg	Different	Yes	not matter.
			test.xsl			
		abc.doc	abc.jar			
	Different error errot of files in heth	def.txt	def.txt			
	Different amount of files in both	config.properties	config.properties			
	directory and file names and	help.java	help.java			
	content both different. (Directory	file.jsp	file.jsp			
	1 has more files)	picture.pic	picture.pic			
		graph.jpg				
		abc.doc	abc.jar			
		def.txt	def.txt			
	Different amount of files in both	config.properties	config.properties			
	directory and file names and	help.java	help.java			
	content are the same. (Directory	file.jsp	file.jsp			
	2 has more files)	picture.pic	picture.pic			
	,	graph.jpg	graph.jpg			
			test.xsl			
			abc.doc (content			
			different from abc.doc			
		abc.doc	in directory 1)			
			abcdef.txt (content			
	Different amount of files in both		same as def.txt in			
	airectory and file names and	def.txt	directory 1)			
	content both different	config.properties	config.properties			
		help.java	help.java			
	1				1	1

		file.jsp	file.jsp			File order does not
12		picture.pic	picture.pic	Different	Yes	matter. File type does
		graph.jpg				not matter.
			abc.doc (content different from abc.doc			
D	Different amount of files in both	abc.doc	in directory 1)			
di	irectory and file names and		abcdef.txt (content			
C	ontent are the same		same as def.txt in			
		def.txt	directory 1)			
		config.properties	config.properties			
		help.java	help.java			