Jesse McConnell and John Casey on IRC, 31-Jan-2007

```
Jan 31 11:14:18 jdcasey so, this is sort of
the laundry list we've worked up for the
plugin/lifecycle handling so far, I think:
Jan 31 11:14:19 jdcasey
http://docs.codehaus.org/display/MAVEN/Lifec
ycle+and+Plugin+Handling
Jan 31 11:14:35 jdcasey etrade has, of
course, added their own unique twist to
things, too
Jan 31 11:15:06 jesse last editted jun 20 :)
Jan 31 11:15:27 jdcasey yup, but nothing
much has changed...I'm pulling it out of
mothballs
Jan 31 11:15:34 jdcasey there's another one
by brett too...looking
Jan 31 11:15:56 jdcasey
http://docs.codehaus.org/display/MAVEN/Plugi
n+Execution+Model+and+Lifecycle+Improvements
Jan 31 11:16:05 jdcasey you'll like the
timestamp on that one even better :)
Jan 31 11:17:36 jdcasey so, some of this has
been addressed (at least partially)...the
flexible artifact filtering, I mean
Jan 31 11:18:57 jesse ah, this problem
Jan 31 11:19:16 jesse good to be getting
unmothballed, it has a lot of ramifications,
or it can at least
Jan 31 11:19:20 jdcasey so, the big part of
that page that I want to address is phase
```

ordering and suppression
Jan 31 11:19:22 jdcasey right
Jan 31 11:20:11 jdcasey the other twist is that a client brought up the fact that you have to know the lifecycle mapping, plus all inherited plugin bindings, before you can intelligently suppress + add a replacement mojo binding
Jan 31 11:20:22 jdcasey because you have to know where to bind the replacement
Jan 31 11:21:15 jesse the whole build needs to be planned out in other words, before it can be filtered
Jan 31 11:21:41 jdcasey I consider it bad design to say "we'll solve it with tools", but one could make a plugin that would inject the bindings already present for a particular phase into the current pom, for the sake of reordering/suppressing them
Jan 31 11:21:46 jesse to come up with the true build
Jan 31 11:22:17 jdcasey well, yeah, I suppose. I think it would be valuable to have fine-grained control over ordering, to whatever degree you want it (all phases or just one, f.e.)
Jan 31 11:22:28 jesse seems to me that fundamentally this is a question of 'how maliable do we want the lifecycle to be?'
Jan 31 11:22:43 jdcasey and I don't think that adding more and more phases will solve

the problem...it just adds confusion and destroys the semantics of existing phases
Jan 31 11:22:54 jesse good good
Jan 31 11:22:58 jdcasey yeah, that's definitely true
Jan 31 11:23:12 jdcasey but I'm not sure that's the only question
Jan 31 11:23:13 jesse all this pre- post- pre-pre- post-post- stuff needs to stop
Jan 31 11:23:26 jdcasey also: how much should a build author need to know about his context?
Jan 31 11:23:37 jdcasey in terms of the inherited phase bindings, packaging, etc.
Jan 31 11:23:41 jdcasey yup, definitely
Jan 31 11:23:54 jesse well, once you decide that fundamentally the lifecycle is solid, then you decend into ording issues within phases themselves
Jan 31 11:23:57 jdcasey post-* really implies something that it cannot currently deliver anyway
Jan 31 11:24:11 jdcasey you only get the post-* phase if you run the build to or past it
Jan 31 11:24:34 jdcasey I think the lifecycle is critical. it gives a set of verbs that are universal
Jan 31 11:24:39 jdcasey that's really important
Jan 31 11:24:53 jesse totally, without it

you become ant
Jan 31 11:25:02 jdcasey yeah, basically
Jan 31 11:25:10 jdcasey maven 1 was an advanced ant
Jan 31 11:25:16 jesse right
Jan 31 11:25:42 jesse lets work within and actual phase..
Jan 31 11:25:48 jdcasey right
Jan 31 11:26:06 jesse maybe compile, imagine if the compile plugin was broken out into 3 pieces
Jan 31 11:26:14 jdcasey so, we need to be able to do three things in a phase, from what I see:
Jan 31 11:26:20 jdcasey 1. suppress a phase binding
Jan 31 11:26:20 jesse scanning, pre-processing, compiling
Jan 31 11:26:32 jdcasey 2. specify the ordering of the bindings
Jan 31 11:26:54 jdcasey 3. replace a phase binding without having to know too much about it (this last one is tricky, and I'm not entirely sure about it)
Jan 31 11:27:02 jdcasey yup, sure
Jan 31 11:27:24 jesse in my exaple you can supress pre-processing in java (mostly)
Jan 31 11:27:32 jdcasey how?
Jan 31 11:27:36 jesse but scanning sure needs to come first
Jan 31 11:27:40 jdcasey right

Jan 31 11:27:51 jesse hypotheically it can be, how is your problem :P
Jan 31 11:28:16 jdcasey hehe
Jan 31 11:28:17 jdcasey ok
Jan 31 11:28:24 jesse 1) does this need to be solved in the pom by the user?
Jan 31 11:28:25 jdcasey left as an exercise to the reader
Jan 31 11:29:02 jdcasey well, here's the thing on #1 there: users may need to specify a different scanner. how do they do that?
Jan 31 11:29:17 jesse seems to me that scanning could be declared as a requirement before compiling happens
Jan 31 11:29:55 jdcasey so the compiler declares a dependency on some sort of scanner interface or something?
Jan 31 11:30:01 jdcasey or a scanning action
Jan 31 11:30:02 jdcasey ?
Jan 31 11:30:32 jdcasey that's actually the type of thing we tried to model when we put together the concept of phases in the first place.... :)
Jan 31 11:30:41 jesse assuming scanning is some kinda plugin and compile is a seperate one...this is your state machine thing
Jan 31 11:30:51 jdcasey yeah, sure
Jan 31 11:31:01 jdcasey ok, so the compiler cannot proceed unless scanning does so first
Jan 31 11:31:03 jdcasey but...
Jan 31 11:31:18 jdcasey if I need to replace

the default scanner, how would I do that?
Jan 31 11:31:23 jesse hehe, how about this
Jan 31 11:31:27 jdcasey or, do you mean that this dependency creates an ordering?
Jan 31 11:31:32 jdcasey sort of like a DAG?
Jan 31 11:31:45 jesse each plugin declares a StateSetter and a StateRequirement
Jan 31 11:31:49 jesse exactly
Jan 31 11:32:03 jesse you can put whatever StateSetter in you want
Jan 31 11:32:19 jesse but if you want to replace it you need to duplicate the StateSetter
Jan 31 11:32:27 jdcasey that actually starts to blur the line between m2 lifecycle phases, and m1 preGoal/postGoal...if you can abstract the dependency term, you're ok
Jan 31 11:32:28 jesse you _need_ a DAG of some sort here
Jan 31 11:32:29 jdcasey hmm
Jan 31 11:32:34 jdcasey yeah, I suppose so
Jan 31 11:32:47 jesse otherwise how do you expect to reproduce it
Jan 31 11:33:00 jesse you need some ordering mechanism
Jan 31 11:33:08 jdcasey yeah, that's true...unless the addition logic never changes, which is what we have now
Jan 31 11:33:13 jdcasey right
Jan 31 11:33:42 jesse DAG = state graph of ordering inside a phase

Jan 31 11:33:47 jdcasey the ordering mechanism that we've been talking about heretofore has been some place where the user takes the stick and dlineates exactly what happens when within a phase
Jan 31 11:34:03 jdcasey your concept gets us closer to a workflow, right?
Jan 31 11:34:15 jesse ya
Jan 31 11:34:35 jdcasey it'd be interesting to see where general-purpose things like antrun or the shell plugin I wrote would fit into this
Jan 31 11:34:35 jesse that can be analyzed for holes pre-exec
Jan 31 11:34:47 jdcasey they'd have to have some dynamic way of declaring themselves as StateSetter
Jan 31 11:35:01 jesse hm..
Jan 31 11:35:18 jesse not declaring a statesetter means that the work anywhere...freeform
Jan 31 11:35:37 jesse but if the user forces a order externally then they can set it in their pom?
Jan 31 11:35:42 jdcasey yeah, I think I've been dancing around this concept of a workflow for quite awhile now...the problem is declaring an abstract dependency term...but that's sort of what we've started doing with the build context, I guess
Jan 31 11:36:38 jdcasey well, the idea was

that the user would specify a binding on the shell plugin, suppress the old make:install plugin, and specify the ordering of the package phase to push the shell plugin back up front, ahead of rpm:build
Jan 31 11:36:40 jdcasey f.e.
Jan 31 11:36:41 jesse so I write a plugin FrackItUp and it needs to run after antrun since I just wanted to use it...but both of these need to run in generate-sources since mines just twiddles with the sources...I place a stateSetter in antrun
Jan 31 11:37:11 jdcasey and the client wanted to say something like <replaces><groupId/><artifactId/>[<version/>]<executionId/></replaces>
Jan 31 11:37:28 jesse and our lifecycle plugins get an honorary awesomeness stateSetter equal to the name of the phase
Jan 31 11:37:41 jdcasey ok, so stateSetter would have to be added as some sort of markup in the plugin config
Jan 31 11:37:43 jdcasey hmm
Jan 31 11:37:52 jesse ya
Jan 31 11:37:53 jdcasey :)
Jan 31 11:38:09 jesse and it could use your build state context
Jan 31 11:38:26 jdcasey yeah, I see where you're going
Jan 31 11:38:42 jesse most people don't need to dick with this kinda stuff

Jan 31 11:38:53 jdcasey the funny thing is, the phases just become arbitrary markers in the larger lifecycle workflow in this...
Jan 31 11:39:04 jesse ya
Jan 31 11:39:12 jesse hm...
Jan 31 11:39:13 jdcasey since state consumer/producer would have to be sorted globally for the lifecycle
Jan 31 11:39:32 jdcasey well, the reason most people don't have to mess with this is that the lifecycle was designed for java dev
Jan 31 11:39:44 jesse true true
Jan 31 11:39:52 jdcasey that will change if maven finds an audience outside of java
[SNIP]
Jan 31 12:15:54 jdcasey I do like the simple elegance of saying "this execution replaces that execution" but in reality, this doesn't lessen the complexity...
Jan 31 12:16:03 jdcasey you still have to know the execId of that other execution
Jan 31 12:16:12 jdcasey which is not trivial to discover, I suppose
Jan 31 12:16:21 jdcasey unless you're looking through the debug output
Jan 31 12:16:26 jdcasey anyway, back to the topic
Jan 31 12:16:51 jesse I have a new question :)
Jan 31 12:16:53 jdcasey I like the idea that a build extension can add a new lifecycle

phase mapping, which is why keeping it in components.xml is nice
Jan 31 12:16:56 jdcasey ok :)
Jan 31 12:17:27 jesse is adding plugin ordering inside a phase tantamount to undermining the concept of a lifecycle phase?
Jan 31 12:18:01 jdcasey you mean because a phase is implicitly atomic and indivisible or something?
Jan 31 12:18:30 jesse moment
Jan 31 12:20:10 jdcasey k
[SNIP]
Jan 31 12:21:58 jesse but yes in a nutshell..
Jan 31 12:23:05 jesse isn't ordering inside of a phase somewhat rending the concept of a hard and fast phase redundent
Jan 31 12:24:39 jdcasey well, the important thing we must retain about phases is the vocabulary available to the user
Jan 31 12:24:42 jdcasey IMO
Jan 31 12:24:55 jesse fair enough
Jan 31 12:25:01 jdcasey that's a compatibility issue...if we get away from even a semblance of the lifecycle, we're in m3 territory
Jan 31 12:25:26 jdcasey beyond that, I have no problem saying that "phases" are just bookmarks in the workflow
[SNIP]

Jan 31 12:27:34 jdcasey what we could really use is a "provides"/"requires" syntax for plugins
Jan 31 12:27:54 jesse for state ya
Jan 31 12:27:59 jesse I don't think we can get awya from that
Jan 31 12:28:00 jdcasey ...then somehow overlay the phase names on top of that...if they don't jive, throw an exceptoin
Jan 31 12:28:04 jdcasey nope, agreed
Jan 31 12:28:32 jesse a DAG DOM validator :P
Jan 31 12:28:38 jdcasey that would help a lot of things...though you could argue: what if multiple mojos provide the requirement for another mojo? how does it sort out then?
Jan 31 12:28:56 jdcasey now, if only we could find an acronym for DAG-GUM :)
Jan 31 12:29:15 jesse throw an exception
Jan 31 12:29:22 jdcasey hmm
Jan 31 12:29:27 jesse hrm
Jan 31 12:29:31 jesse no
Jan 31 12:29:33 jdcasey aspectj and javac could be used together
Jan 31 12:29:38 jdcasey hmm
Jan 31 12:30:12 jdcasey I dunno, I'm almost in favor of just forgetting all of this detection and going with an explicit <ordering/> type element
Jan 31 12:30:14 jesse lemme mock something
Jan 31 12:30:16 jdcasey <lifecycleManagement/>

Jan 31 12:30:18 jdcasey ok
Jan 31 12:30:19 jesse ya
Jan 31 12:30:29 jesse _exactly_ what I was just going to mock
Jan 31 12:30:33 jdcasey the detection/magic has gotten us in trouble inthe past
Jan 31 12:30:44 jesse only problem is muliiple execution of the same plugin
Jan 31 12:31:11 jesse then you need to intrduction executionId into =it
Jan 31 12:31:26 jdcasey well, the ordering would have to have execution resolution, not just mojo resolution, IMO
Jan 31 12:31:36 jdcasey right
Jan 31 12:31:57 jdcasey and executionId all of a sudden becomes a much more important figure, where it's sort of a redheaded stepchild now
Jan 31 12:32:15 jesse just smells dirty...
Jan 31 12:32:21 jdcasey which IMO would be a good thing...it makes inheritance less quirky if people are thinking about it
Jan 31 12:32:26 jdcasey :)
Jan 31 12:32:30 jdcasey hmm
Jan 31 12:33:06 jdcasey part of the problem is that we don't have a good, consistent way of referencing a plugin/mojo/execution with a terse syntax...which means <lifecycleManagement/> could get ugly real quick
Jan 31 12:33:29 jdcasey we need to

standardize on g:a[:v]:mojo:execId, if we go to lifecycleMgmt
Jan 31 12:33:31 jdcasey IMO
Jan 31 12:33:53 jesse exactly
Jan 31 12:34:05 jdcasey and actually, g:a[:v]:mojo[:execId] where version is discovered, and execId defaults to 'default'
Jan 31 12:34:25 jesse putting <plugin><aid>gid>ve is tediuous for that
Jan 31 12:34:25 jdcasey we do something like this in the components.xml now, but it's handled in the lifecycle executor, and it's ugly
Jan 31 12:34:29 jdcasey we could make it more formal
Jan 31 12:34:36 jdcasey exactly
Jan 31 12:34:45 jdcasey I'd hate it, I can tell you thta
Jan 31 12:34:47 jdcasey that
Jan 31 12:35:08 jesse does Artifact have a toString() that makes it pretty and machine usable?
Jan 31 12:35:20 jdcasey would be nice to have shorthand, too...like: <otherBindings/> so you can do:
Jan 31 12:35:46 jdcasey <phase><binding>g:a:v:m:e</binding><otherBindings/></> to specify that one is definitely first
[SNIP]
Jan 31 12:44:13 jdcasey though I do think in

the wider context, we'll need to make execIds more prominent...if we're after suppression too...
Jan 31 12:44:37 jesse ok, so its at the execid point and not plugin
Jan 31 12:45:05 jdcasey well, if you can have compiler running for two purposes at different points, you have to have the precision to affect one and not the other
Jan 31 12:45:12 jdcasey think antrun instead of compiler if you like
Jan 31 12:45:20 jesse sure
Jan 31 12:45:31 jdcasey IMO, affecting the plugin as a whole is heavy-handed, and won't be too useful
Jan 31 12:45:36 jesse this is a shitty problem, pardon my language
Jan 31 12:45:48 jdcasey well, yeah, in a way it is
Jan 31 12:46:12 jdcasey but it's funny that we added functionality for additive lifecycle mappings, but no subtraction or replacement...don't you think?
Jan 31 12:46:20 jesse ok, another solution other then #s
Jan 31 12:46:27 jdcasey k
Jan 31 12:46:36 jesse <before> and <after> syntax on execution id's
Jan 31 12:46:46 jesse execution id's become a global ordeal
Jan 31 12:46:53 jdcasey so you say

before-some-other-execId?
Jan 31 12:46:58 jesse <ebfore>execId
Jan 31 12:47:01 jdcasey hmm
Jan 31 12:47:10 jdcasey damn, this does suck
Jan 31 12:47:12 jesse then you can build the DAG and look for collisions
Jan 31 12:47:20 jesse execID needs tobe global then though
Jan 31 12:47:25 jdcasey ick
Jan 31 12:47:37 jesse global to the phase at least
Jan 31 12:47:49 jdcasey hmm
Jan 31 12:47:49 jesse might be the simplest though
Jan 31 12:48:27 jdcasey what about <before>g:a:v:e[:m]</>, and use that instead of <phase/> ?
Jan 31 12:48:42 jdcasey we could still use phase for additions
Jan 31 12:48:56 jesse oh!
Jan 31 12:48:59 jdcasey this would be relative positioning, as opposed to normal
Jan 31 12:49:08 jesse <before> <after> <replace>
Jan 31 12:49:16 jdcasey yup
Jan 31 12:49:18 jesse all operating off of execId
Jan 31 12:49:20 jdcasey that's what I was thinking
Jan 31 12:49:30 jesse execid can be freeform like it is now

Jan 31 12:49:38 jdcasey though we should probably preserve <phase/> in there too...for don't-care conditions
Jan 31 12:49:44 jesse but we can recommend the convention of your idea up there
Jan 31 12:49:50 jesse totally
Jan 31 12:49:52 jesse acutally
Jan 31 12:49:56 jesse you still need it there
Jan 31 12:50:02 jdcasey so it'd use a global execId search, or only within a specified plugin?
Jan 31 12:50:08 jdcasey how so?
Jan 31 12:50:14 jdcasey need the phase, you mean?
Jan 31 12:50:17 jesse inside the _phase_
Jan 31 12:50:27 jdcasey so before requires phase?
Jan 31 12:50:30 jesse if there is not phase then its the plugin phase
Jan 31 12:50:41 jdcasey the one it's referencing?
Jan 31 12:50:42 jesse but the Phase concept is a req
Jan 31 12:50:46 jdcasey yup, agreed
Jan 31 12:50:51 jesse exec can have a phase in it
Jan 31 12:50:56 jdcasey so, you have two things happening here
Jan 31 12:50:56 jesse for overrideing hte plugin phase

Jan 31 12:51:11 jdcasey absolute addition, with no other plugin for reference of positioning, etc.
Jan 31 12:51:29 jdcasey and relative addition, etc. with a reference plugin for relative positioning
Jan 31 12:51:56 jdcasey and relative specifications would override the @phase just like <phase/> does
Jan 31 12:52:49 jdcasey now, for suppression, I guess you could have a <skip/> inside the execution...
Jan 31 12:52:50 jesse I would actually throw an exception on #3
Jan 31 12:52:57 jdcasey really?
Jan 31 12:53:15 jdcasey we don't on <phase/> overrides...why on <before/> ?
Jan 31 12:53:23 jesse if you claim <ebfore>foo and your not in the same phase as foo then cry foul
Jan 31 12:54:00 jesse foo wouldn't appear in the phase execId's
Jan 31 12:54:09 jesse so it wouldn't know where to put it in the graph
Jan 31 12:54:17 jesse if execi'd are global the phase
Jan 31 12:54:36 jesse i r a gud spelr
Jan 31 12:54:40 jdcasey yeah, ok...got that
Jan 31 12:55:00 jdcasey hmm, lunch time here...
Jan 31 12:55:05 jesse skip is a problem

Jan 31 12:55:11 jdcasey why is that?
Jan 31 12:55:27 jesse <skip> is absolution, supression is deterministic isn't it?
Jan 31 12:55:57 jesse ugh
Jan 31 12:56:09 jesse I need to understand supression better I think
Jan 31 12:56:21 jdcasey suppress would basically mean "take this out of the lifecycle map"
Jan 31 12:56:30 jdcasey I was using it interchangeably with skip
Jan 31 12:56:32 jesse it seems like <skip> ought to be a plugin thing
Jan 31 12:56:58 jesse cause if you want to skip and exec, just don't put it in :)
Jan 31 12:57:11 jdcasey not sure i agree...if the plugin dev fails to provide it, the user is screwed...also, it means a proliferation of skip* params
Jan 31 12:57:26 jesse no no
Jan 31 12:57:34 jesse not in the config of the plugin
Jan 31 12:57:41 jesse sibling to <version>
Jan 31 12:57:52 jesse on the plugin objecy
Jan 31 12:57:53 jdcasey ah
Jan 31 12:58:06 jdcasey makes sense to me
Jan 31 12:58:15 jdcasey knock out all related functionality
Jan 31 12:58:20 jesse yep
Jan 31 12:58:44 jesse it would be equivalent to skiping the lifecycle for lots of these

things
Jan 31 12:58:48 jdcasey though I do have a use case where I want to skip only a single mojo in the plugin...
Jan 31 12:59:00 jdcasey skip make:install, but not make:compile or make:check
Jan 31 12:59:13 jesse o.O
Jan 31 12:59:31 jesse thats one plugin? :)
Jan 31 12:59:50 jdcasey yup...it's all related functionality, and uses the same infra
Jan 31 12:59:55 jesse that plugin applies to multiple phases
Jan 31 13:00:06 jdcasey compiler plugin does too
Jan 31 13:00:06 jesse oh!
Jan 31 13:00:24 jesse <skip/> global, <skip>mojo,mojo</skip>
Jan 31 13:00:49 jdcasey that's a cool idea...
Jan 31 13:01:17 jesse lets you wack out the whole thing or a particular mojo
Jan 31 13:01:31 jdcasey all exec's, though...it's a good clean soln
Jan 31 13:01:34 jesse <skip>MakeInstallMojo</skip>
Jan 31 13:01:56 jdcasey yup
Jan 31 13:02:08 jesse <before> <after> <replace> on the exec, and <skip~/> on the plugin object
Jan 31 13:02:21 jdcasey cool. I'm going to

write this up and send it to dev@
Jan 31 13:02:25 jesse exec id's are phase global
Jan 31 13:02:26 jdcasey do you mind me quoting this?
Jan 31 13:02:31 jdcasey right
Jan 31 13:02:52 jesse go for it, you can wack out my brain dump bits, maybe just this idea part
Jan 31 13:03:02 jdcasey yup, cool
Jan 31 13:03:37 jdcasey bbiab
Jan 31 13:03:40 jesse cool
Jan 31 13:14:42 jesse can I mention this on joakim's pre and post test thread?
Jan 31 13:14:57 jesse just that john has something coming that ought to address this..:)
Jan 31 13:16:55 jdcasey yeah, that'd be cool.
Jan 31 13:17:08 jdcasey I'm going to head to campus, then I'll write it up and submit it.
Jan 31 13:56:01 * Disconnected ().
Jan 31 16:01:53 jdcasey ok, almost got this proposal written up
Jan 31 16:02:43 jdcasey I think we may need to skip executions, though...it's possible that a particular binding is the offender, not necessarily the entire functionality of the mojo in all places...does that sound reasonable?
Jan 31 16:03:31 jesse o.O

Jan 31 16:03:37 jdcasey no?
Jan 31 16:04:05 jesse what is the difference between skip on an exec and just commenting it ouy?
Jan 31 16:04:07 jesse out?
Jan 31 16:04:22 jdcasey how do you comment it out if it's in a parent pom?
Jan 31 16:04:47 jesse ack
Jan 31 16:05:01 jdcasey I'm not saying it has to be only all-plugin or by-exec...I'm saying it needs to be all-plugin, all-mojo, or by-exec
Jan 31 16:05:24 jesse <skip>execid,execid</skip> on a exec then?
Jan 31 16:05:48 jdcasey hmm, placement is tricky...I think it should still be a child of <plugin> personlly
Jan 31 16:06:08 jdcasey use <skip>g:a:v:e,...</skip>
Jan 31 16:06:22 jesse could you get this with <replace>?
Jan 31 16:06:28 jdcasey or <skip>mojo1,g:a:v:e(uses mojo2)</skip>
Jan 31 16:06:37 jdcasey you can if you have something to replace with
Jan 31 16:06:46 jdcasey and you can play around it by using a noop plugin to replace with
Jan 31 16:07:01 jesse no, that would still run it with defaults.
Jan 31 16:07:23 jdcasey sorry, not following

Jan 31 16:07:47 jesse replace implys your still in an exec and are replacing one upstream
Jan 31 16:08:10 jesse so its still a valid exec and would run with defaults, not skip it
Jan 31 16:08:27 jesse noop plugin wouldn't help, exec is in a plugin
Jan 31 16:08:56 jesse k, I am cool with your notation I guess
Jan 31 16:09:07 jdcasey I meant that the noop plugin would specify an execution that replaces the one in the other plugin
Jan 31 16:09:21 jesse oh!
Jan 31 16:09:25 jesse yes, that would work
Jan 31 16:09:43 jdcasey but, it would be more elegant to simply say 'skip that antrun execution completely'
Jan 31 16:09:47 jesse <skip> might have too much diverse functionality otherwise
Jan 31 16:09:50 jdcasey less cruft
Jan 31 16:09:54 jdcasey hmm
Jan 31 16:10:18 jesse mixing mojo1, and this other notation is not elegant imo
Jan 31 16:10:28 jdcasey agreed
Jan 31 16:10:44 jdcasey I don't like the syntax of comma-delimited lists, actually
Jan 31 16:10:46 jdcasey but that's a detail
Jan 31 16:10:47 jdcasey IMO
Jan 31 16:10:48 jesse a noop plugin requires forethouht at least...I kinda like it

```
Jan 31 16:11:02 jesse <skips><skip> :)
Jan 31 16:11:29 jdcasey
<skip><goals><goal>mojo1</></><executions><e
xecution>exec1</></></>
```

```
Jan 31 16:11:31 jdcasey maybe?
Jan 31 16:11:44 jdcasey <skip><all/></> :)
```