



# JavaOne™

[java.sun.com/javaone](http://java.sun.com/javaone)

## TEN WAYS TO DESTROY YOUR COMMUNITY

Josh Berkus, Sun Microsystems

TS-5502



# Part I: The Evil of Communities



**They mess up your  
marketing plans**  
by doing their own marketing and PR

**They mess up your  
product plans**  
with unexpected innovation

**They're never satisfied by any  
amount of quality**  
and keep wanting to improve the software

**They re-define your partner and  
customer relationships  
and confuse your salespeople**

**They require you to communicate  
constantly  
and who has time for that?**

**If Only There Were Some Way to  
Rid Yourself of the Community  
Menace ...**





**The Berkus**  
**Patented**

**The Berkus**  
**Patented**  
**Ten-Step Method**

**The Berkus**

**Patented**

**Ten-Step Method**

*To Destroy*

*Your Community*

# 1. Difficult Tools

- weird build systems
- proprietary version control systems
- limited license issue trackers
- single-platform conferencing software
- unusual & flaky CMS

## **2. Poisonous people**

**Maximize the damage they can do!**

- 1. Argue with them at length**

## **2. Poisonous people**

**Maximize the damage they can do!**

- 1. Argue with them at length**
- 2. Denounce them venemously**

## **2. Poisonous people**

**Maximize the damage they can do!**

- 1. Argue with them at length**
- 2. Denounce them venemously**
- 3. Ban them**



## **2. Poisonous people**

**Maximize the damage they can do!**

- 1. Argue with them at length**
- 2. Denounce them venemously**
- 3. Ban them**
- 4. Argue with them in other projects/sites**


## **2. Poisonous people**

**Maximize the damage they can do!**

- 1. Argue with them at length**
- 2. Denounce them venemously**
- 3. Ban them**
- 4. Argue with them in other projects/sites**
- 5. Allow them back into your project**

## 2. Poisonous people

Maximize the damage they can do!

- 
1. Argue with them at length
  2. Denounce them venemously
  3. Ban them
  4. Argue with them in other projects
  5. Allow them back into your project
  6. GOTO 1

# 3. No documentation

## DON'T

- ...document the code
- ...document the build methods
- ...document the submission process
- ...document the release process
- ...document how to install it

# 3. No documentation

## DON'T

- ...document the code
- ...document the build methods
- ...document the submission process
- ...document the release process
- ...document how to install it
- ...but always tell people RTFM!

# 4. Closed-Door Meetings

Good

Short-notice online meetings

# 4. Closed-Door Meetings

Good

Short-notice online meetings

Better

Telephone meetings

# 4. Closed-Door Meetings

**Good**

Short-notice online meetings

**Better**

Telephone meetings

**Best**

Meet in person, in your secure office



# 5. Legalese, legalese, legalese

The longer and more complex the better!

Contributor agreements

Website content licensing

Non-disclosure agreements

Trademark licensing terms

Bonus: change the documents every couple of months, without any official notice.

# 6. Bad liaison

Someone reclusive

# 6. Bad liaison

Someone reclusive

or

Someone with no time

# 6. Bad liaison

Someone reclusive

or

Someone with no time

or

Someone with no authority

# 6. Bad liaison

Someone reclusive

or

Someone with no time

or

Someone with no authority

or

Someone unfamiliar with the technology

# 6. Bad liaison

Someone reclusive

or

Someone with no time

or

Someone with no authority

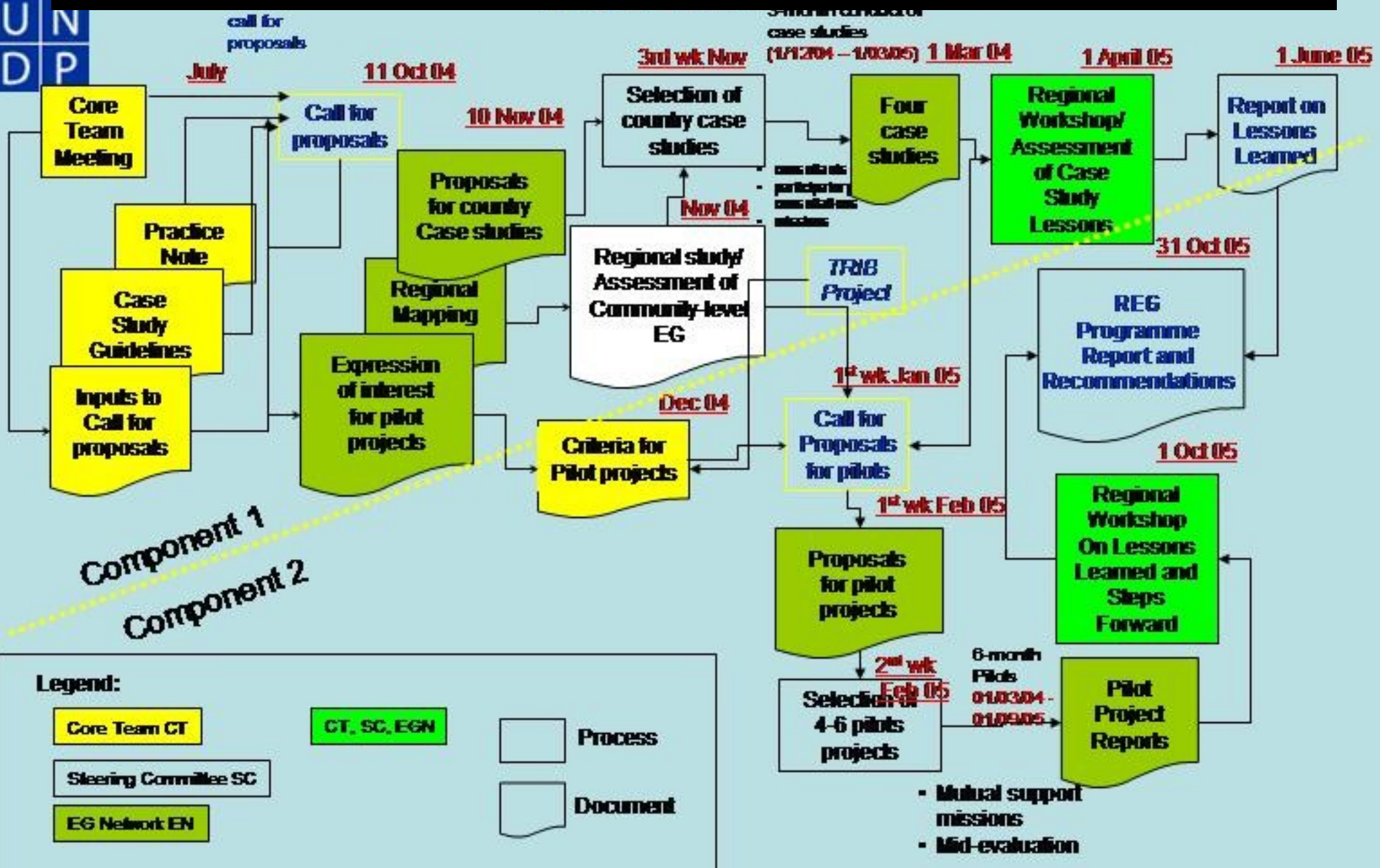
or

Someone unfamiliar with the technology

or

No liaison at all!

# 7. Governance obfuscation



# 7. Governance obfuscation

## Three Principles:

- (1) Decision making and elections should be extremely complex and lengthy;
- (2) Make it unclear what powers community officials & committees actually have;
- (3) Make governance rules nearly impossible to change.



# 8. Screw around with licenses

License  $\approx$  Identity

# 9. No outside committers

- I. No matter how much code outsiders write, only employees get to be committers.

# 9. No outside committers

- I. No matter how much code outsiders write, only employees get to be committers.
- II. If they ask why they're not promoted, be evasive!

# 9. No outside committers

- I. No matter how much code outsiders write, only employees get to be committers.
- II. If they ask why they're not promoted, be evasive!
- III. Make sure there are no written rules on who gets to be a committer, or that the the criteria are impossible to fulfill.

# 9. No outside committers

- I. No matter how much code outsiders write, only employees get to be committers.
- II. If they ask why they're not promoted, be evasive!
- III. Make sure there are no written rules on who gets to be a committer, or that the the criteria are impossible to fulfill.
- IV. Bonus: promote an employee who doesn't code to committer!

**10. Be silent**

# The Ten Ways

1. Difficult tools
2. Encourage poisonous people
3. Don't document anything
4. Closed-door meetings
5. Lots of legalese
6. Bad liason
7. Governance obfuscation
8. Screw around with licenses
9. Stop outside committers
10. Be silent

# The Ten Ways

1. Familiar Tools
2. Discourage poisonous people
3. Document everything
4. Accessible online meetings
5. Minimize legalese
6. Expert liason
7. Governance simplification
8. Treat licenses with respect
9. Promote outside committers
10. Communicate



# More Advice

- Josh Berkus: [josh.berkus@sun.com](mailto:josh.berkus@sun.com)
- Sun open source alias: [opensource@sun.com](mailto:opensource@sun.com)