Building Communities

Creating a conversation
Eclipse Today

- 120+ members, including major Java, Linux and Embedded vendors (Actuate, HP, IBM, Sybase, BEA, Borland, JBoss, SAP, RedHat, Novell, Nokia, Monta Vista, Wind River, Mentor, ENEA, QNX)
- 50+ open source projects
- 55-65% market share of Java IDE
- Downloads: 50 million++; 1 million in 40 days for last release
- 70 project leaders, PMC members
- 618 committers
- 24,426 registered Bugzilla users
Eclipse 3.1 Metrics

- Number of newsgroup posts: 32,223
- Number of mailing list posts: 7,792
- Number of Eclipse related blogs: 27
- Number of eclipse.org page views: 500,000,000
- Average Bugzilla queries per day: 600,000
- Bugzilla bug or enhancement reports entered: 21,831
- Bugzilla reports fixed: 9,871
- Most comments on a single bug: 202

An example of the community activity is the metrics around the Eclipse 3.1 release.
Of course a measure of success is that people are using us.
Lots of people think of Eclipse as one big community.
However, we think of the community in two dimensions, the first begin role based. Within the Eclipse community there are users, committers and plug-in developers. Each of these roles participate in the community in different ways and for different purposes. Therefore it is important to approach each role differently.
Eclipse is also a set of smaller communities that are developed around different technologies. They often leverage and work together but they do so from different perspectives.
So the real picture is that within the different technology communities, there exists different roles. There is also a lot of collaboration between the participants across roles and technology.
Of course the network effect starts to take off even within developer communities. The more users you have, the more plug-in developers you get and the more committers are attracted.
How do we think about building and communicating to the Eclipse community? A lot of what we do has been documented and discusses in different books and blogs. There are a few. If you would read one book, everyone needs to read The Cluetrain Manifesto.
The Cluetrain manifesto include 95 theses that describe their approach. For the purpose of this presentation, the key thing is that markets are conversations with human beings. These conversations are doing enabling powerful networks. For vendors is be part of the conversations you need to use a human voice and not use market-speak. The risk of not doing this, is that these conversations are where people are turning to to get their information.
The starting point for any developer oriented community must be developers talking to developers. Developers see through market-speak and don’t respect it. Marketers shouldn’t be fired but they need to change their tactics if they are going to be successful.
“My development manager thinks we are being out marketed, so I would like to do a press release….”

- Developers and Marketers need to share the responsibility of creating the community.
- We are all marketers!
- btww, press releases do not create communities

The quote is based on a true story of someone talking to me. The important point is that developers need to take joint responsibility on the success of building their community. It is not just a marketing role.
Developers like to talk about code and technology. Before any community can be started you need something to talk about and that is typically the code. To keep the conversation going you need to provide updates often.

The Conversation Starts with Code

18 month release cycles don’t cut it;
Try 6 weeks
The Eclipse project is scheduled to release their next major release in June 2006. It is called Eclipse 3.2. There is already 64,000 Google hits on “Eclipse 3.2”.

6 weeks after the 3.1 release, the development team had the first 3.2 M1 release and the project plan for 3.2 up on the web site. They are creating an environment for people to join the conversation.
The Eclipse 3.2 conversation started last September, 2005

9 months before the release!

But this conversation started last Fall. The bloggers were already talking about 3.2 in September. Every 6 weeks a new milestone is released and a conversation starts around that milestone.
It continues in many languages.

The conversation also occurs in many languages. Communities are multi-lingual.
The community decides what is cool; not the vendor

The community also starts to define what is cool, before the press release is written. This is a great resource for marketing types.
Joining the conversation

Make it easy to join the conversation

- Need to have low barriers to entry
- Tutorials, white papers, books
- Have experts monitor newsgroups and bug database

Standard marketing states you want to keep low barriers to entry. Building communities is no different. This allows people to join the conversation.
The Eclipse BIRT projects is a good example
Eclipse Rich Client Platform started as a community initiative
Not planned by a vendor

It is not always the vendor that creates communities or lowers the barrier to entry. The Eclipse RCP initiative was lead by the community. A number of individuals saw the potential of Eclipse to be a rich client platform. They started hacking, writing tutorials, writing white papers. In 3.0 the Eclipse project team decided to do the work to formally support Eclipse as an rich client platform.
Joining the conversation

Create an Architecture of Participation

- Enable users and other parties to extend and modify your technology
- Eclipse, Apache, Firefox
Defining the Term

“what really distinguishes open source is not just source, but an
architecture of participation” that includes low barriers to entry by
newcomers, and some mechanism for isolating the cathedral from
the bazaar. This architecture of participation allows for a real free
market of ideas, in which anyone can put forward a proposed
solution to a problem; it becomes adopted, if at all, by acclamation
and the organic spread of its usefulness.”

Tim O’Reilly
Key Points

- You need a cathedral to enable the bazaar
  - This is not about joining the project itself *per se*, but about making it easy for individuals to add their own ideas to the ecosystem

- Low barriers to entry

- New add ons are first class citizens, not second class

- Empowering individuals and small groups is key
  - Innovation comes from committed fanatics in small groups, not corporate committees

- *Laissez faire* --- never ever ever pretend you can pick a winner

- Provide the seeds for a broad ecosystem
Eclipse makes it easy for third parties to create plug-ins

The result has been one of the secrets to our success

The result for Eclipse is a strong ecosystem of over 1000 plug-ins.
I suggest you think about feeding the conversation for a community. One way of doing this is writing and publishing technical articles.
For example, an article written about the Eclipse Web Tools (WTP) project was publishing on onJava. The article by itself is great to get exposure for WTP. However, look what happens next....
This results in some of the other Eclipse portals and blogs referencing the article; extending the reach of the article into other publications.
Eclipse Web Tools now starts showing up in all sorts of places
Extending the conversation to new people

It is also picked up by other blogs that aren't usually focused on Eclipse, thus extending the awareness and education of Eclipse WTP into the communities not usually reached.
People also save it as a resource

It is also saved by individuals for future reference. Del.cios.us show 35 individuals have tagged this article.
Feeding the conversation

Nurture and promote your tribal leaders

- Persona of the community
- Speak at conferences; write books
- Often define the culture of the community

Tribal leaders is a term from the book ‘Brand Hijack’. These are the people that define the culture and pace of the community. They can be some of the core developers but also users.
Users are developers and they want to be involved in the development process of their community. To often they have been in a situation where they report a bug to a vendor and never hear back. A successful community is a transparent community. Develop in the open, exposure your bugs, everyone knows they exist.
For example, the splash screen in 3.1 was being changed. Typically this change would have been done by the marketing folks with no discussion. For Eclipse it was done as a bug and it had 81 comments and 17 people watching the bug. Marketing was not involved. A good example of getting community buy-in.
Expanding the community

Get your competitors involved

- Allows you to link into other established communities

This is more difficult for vendors as opposed to open source communities but if you can get your competitors involved you instantly start to expand your community.
For example the Web Tools project has IBM, BEA and JBoss involved. One each of the respective web site has information about WTP. This expands the reach of WTP to these vendors developer communities.
You need to think about giving back to the community. Link to blogs, give out awards, highlight other successes. Btw, this also helps with your Google results.
Eclipse is very passionate about ensuring people adopt the technology for commercial adoption. Help promote the fact that other people are using your technology and they will help promote the base technology. A lot of this is an ingredient brand strategy.
The vendor needs to be part of the community. It can’t be a command and control system. Your community will know if you are authentic. You win by letting go.
Pitfalls

- Not all developers are good communicators; or want to be
- Marketers still want to be in control
  - Remember you win by giving up control
- Too easy to fall back into the old way of doing things
- How do you know when you are successful?
What is the marketing role

- Still need to do top-down marketing
  - High level messaging for non-technical decision makers
  - Provide support for developers trying to sell up the chain

- Be part of the community
  - Developers want their community to be successful
  - Make sure you speak their language

- Create a partnership between marketing & development
Summary

- Start the conversation
- Need to make it easy to join the conversation
- Feed the conversation
- Invite others to be part of the community
- Remember to thank the contributors
- Be part of the community
Thank You